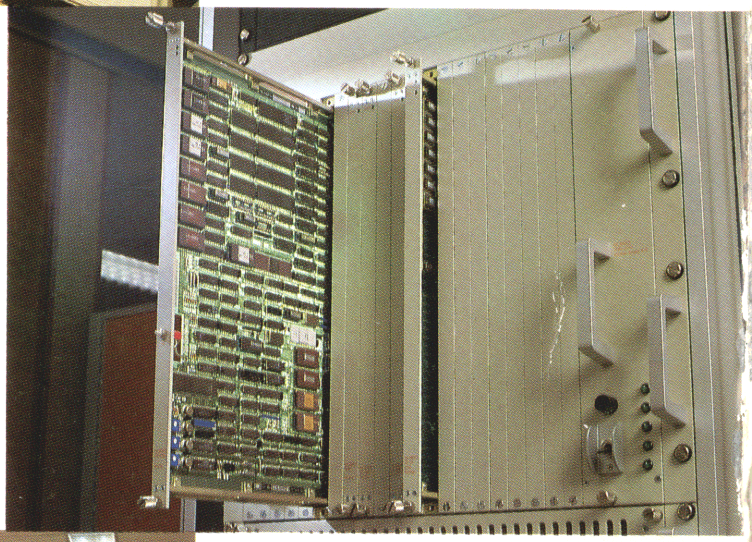
Minicomputer Handbook 82/83

CR80

The Fault Tolerant Multiprocessor CR80 FATOM is the flagship of the CR80 computer family. Here is shown one of eighteen on-line systems to be delivered for a European network stretching from Norway to Turkey. The key features important to our customer was N + 1 redundancy, NO-BREAK computing and the multilevel security of the DAMOS operating system, enabling a smooth implementation of the extensive and complex application programs.

The Processor Unit shown, is the multiprocessor building block of the CR80 computers. It can either work as a stand-alone virtual memory computer or in combination as part of the CR80 FATOM range of NO-BREAK computers. The Processor Unit is a physical entity including its own forced air-cooling and plug-in power supplies. The very low repair times, are achieved by CPU's, Memory etc. being extractable from the front without special tools or dismantling of cables.



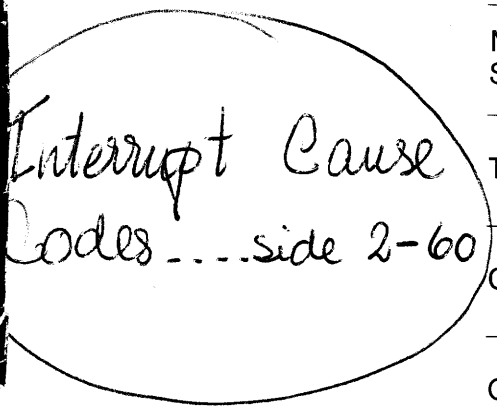
One of many CR80 TWIN Computers used in various British government networks as combined nodal packet switches and Mainframe Front Ends. The networks interfaces a large populations of several thousand terminals in to regional computing centers. The CR80 TWIN was selected on behalf of its mean time between system failure, being less than one in several years, due to dualized processing elements and unique N + 1 redundancy of the communication line interface.

Poul Erik Waldhauer

CONTENTS

CR80 MINICOMPUTER HANDBOOK

1982/83



Doc. reference: CSD/HDBK/0082
1st. edition: December 1981
Published by: Christian Rovsing A/S
Development Division
Lautrupvang 2
DK-2750 Ballerup
Denmark

PREFACE

This is your copy of the CR80M-series Minicomputer Handbook. The handbook is aimed to introducing new users to the CR80 Computer Family, as well as a handy reference for CR80 system designers and programmers. The CR80 range in processing power and cost provides the user with a software compatible choice suited for his application.

Christian Rovsing A/S has been active within computing since the Company's founding in 1962. Design and Manufacture of our first generation minicomputer started in 1972, with the second generation CR80S-series computer announced in 1975, leading to the third generation CR80M-series MINI, TWIN, MAXIM and FATOM Family of computers being introduced during 1981.

This decade of activity in all phases of computer manufacture and use, has seen the CR80 computers incorporated in a wide range of systems, including business, communication, networking industrial, and space applications.

Christian Rovsing serves the market all the way from manufacture of products to software and services. This places us at Christian Rovsing A/S in an unique position to provide you with every assistance in solving your application requirements.

ASBJORN SMITT
VICE PRESIDENT
DEVELOPMENT

1. INTRODUCTION TO THE CR80 COMPUTER FAMILIES

Several years of rapid computer technology evolution have led to the development of the CR80 computer product line at Christian Rovsing A/S. The computer families introduced in this handbook are a collection of units architecturally structured in an innovative way into powerful multiprocessor systems. Through a high degree of parallelism and redundancy, the configurations introduced herein offer nearly unlimited operating power and outstanding system reliability.

From the outset, system architects at Christian Rovsing recognized that micro-electronics was the driving force behind modern computer technology. The CR80 product line is based on the functional modularity made feasible by low-cost LSI completed with advanced distributed architecture and multiprocessing concepts. Though they appear to be minicomputers, the CR80 systems in the larger configurations, are competitive with and challenge the power of large mainframes but with far superior operational characteristics and hereto unrealizable advantages. The CR80 building-block modules allow a system configuration flexibility previously unachievable, this has led to the definition of the CR80 Computer Family depicted in summary block diagrams on the next page (figure 1.1).

Arbitrarily, the CR80 family of computers has been configured into standard computer models and given simple acronyms. The CR80 standard models are called:

- MINI
- TWIN
- MAXIM
- FATOM

The model names are simple descriptors of the characteristic features of each configuration which are listed below the block diagrams in the figure.

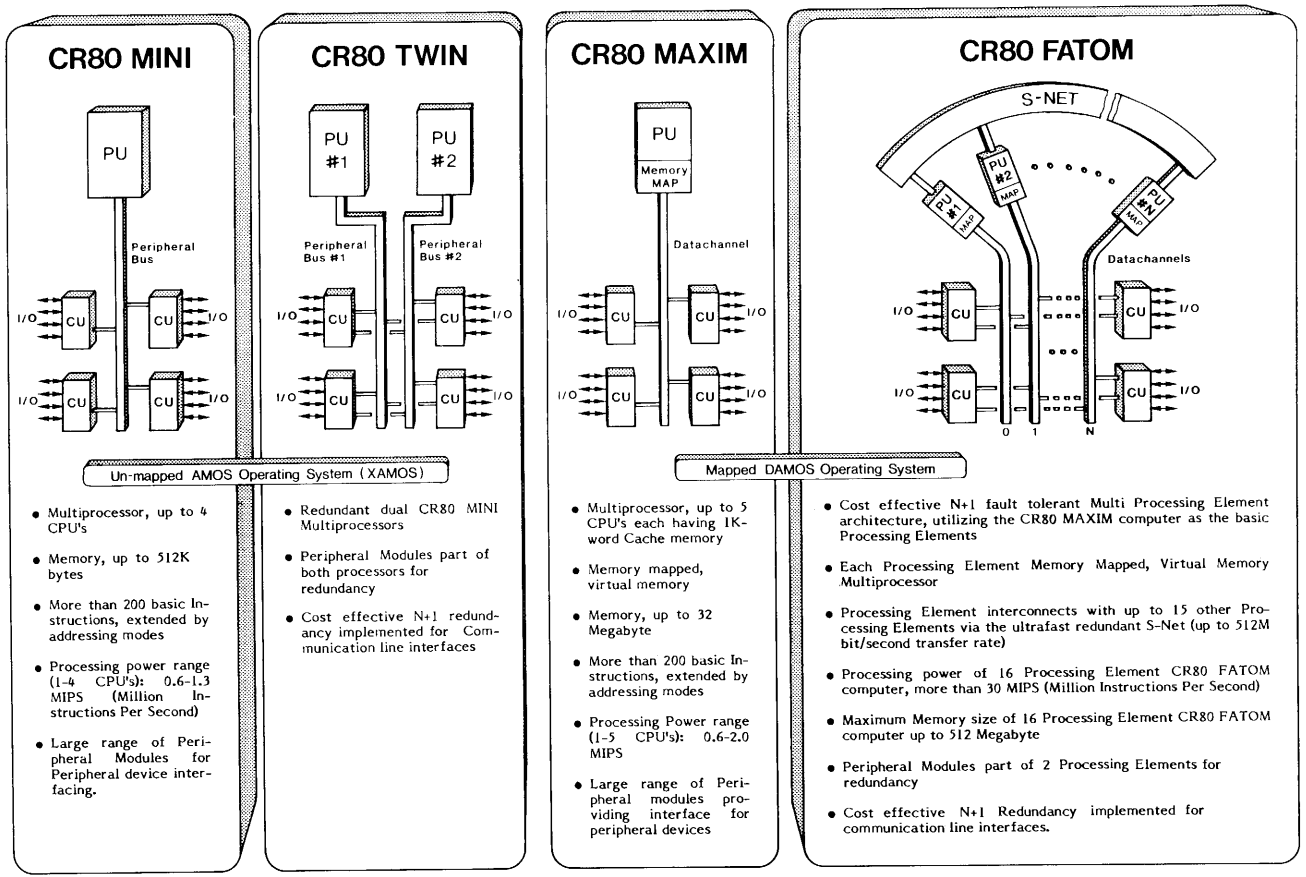


Fig.1.1 **THE CR80 FAMILY OF MINICOMPUTERS**

As qualified above, system boundaries are arbitrary and somewhat hard to define since they are truly non-existent. The CR80 product line as such probably offers the most versatile computer configurations in the industry. Nevertheless for purposes of standardization, the CR80 systems cross through 4 smooth transition levels.

The standard CR80 models are divided into two classes - unmapped and mapped - supported respectively by the AMOS and DAMOS software operating systems.

The unmapped systems included the

- CR80 MINI, a multiprocessor system with up to 4 CPU's and 256 K words of memory with an operating range of 0.6 to 1.3 million instructions/second; and the

- CR80 TWIN, a fully-dualized version of the MINI with twin multiprocessors and a dual bused peripheral subsystem.

The mapped systems include the

- CR80 MAXIM, a multiprocessor system with up to 5 CPU's and 16 megawords of memory with an operating range of 0.6 to 2.0 million instructions/second and a Data Channel with a megabyte/sec. transfer rate interfacing up to 15 channel units for control of up to 960 peripheral modules, and the

- CR80 FATOM, a fault-tolerant system comprised of as many as 16 multiprocessors interconnected through a 512 megabit message transport; each multiprocessor has the same capabilities as a CR80 MAXIM with 256 megawords of memory and an operating range up to 30 million instructions/second.

These standard configurations encompass a broad range of physical characteristics to meet the requirements of the smaller stand-alone user and those of the largest multi-installation network applications. The four models offer

- a 50:1 range in instruction execution rate varying from 0.6 mips to 30 mips.

- a 1000:1 range in memory capacity from 512 K bytes to 512 megabytes.

- a 80:1 range in processing power utilizing one CPU or up to 16 interconnected multiprocessors with a maximum of 5 CPU's each.

- a 400:1 range in connectivity through Peripheral controllers accommodating a variety of terminations with as many as 960 peripherals or up to 4096 communication lines.

Flexible variation in the size and structure of the CR80 systems are permitted by the unusual degree of hardware and software modularity. The hardware essentially consists of fast transfer buses joined to each other by adapters which allow units on one bus to access those on another. Dualization at the internal level and multiple redundancy at the system level provide a CR80 hardware architecture which is exploited by the DAMOS software operating system and programs to survive operational failure of individual components.

Reliability, which is increasingly becoming of concern in real-time and distributed network applications, is achieved in the CR80 computer systems by applying unique architectural concepts. The CR80 hardware/-software architecture treats all multiprocessors as equal elements not absolutely dedicated to a specific role. Fault tolerance and backup are achieved through an n+1 redundance scheme without preassignment of system functions to specific processors. This is in marked contrast to the more common rigid dualized configurations often encountered in dedicated applications with on-line master/slave arrangements, or off-line backup with switchover facility.

The many functional and operational features inherent in the CR80 computer system configurations presented in this handbook go beyond the mere physical size variations and expansion options. The CR80 MINI, TWIN, MAXIM and FATOM standard computer models are listed in chapter 11. Specific features of the mapped CR80 MAXIM and FATOM computers are covered in the general overview preceding the detailed technical presentation in Section 2, and those of the unmapped CR80 MINI and TWIN computers in Section 5. The corresponding software support systems DAMOS and AMOS are presented in Section 4 and 6. As a general introduction and to orient these later detailed discussions, the highlights of the CR80 Computer Family characteristics are presented here.

The following list of highlights is not exhaustive. Rather, it is meant to focus on those operational capabilities meaningful to potential users in varied applications such as private data networks, front-end processors, data concentrators, multi-terminal systems, real-time on-line systems, packet-switched networks or process control.
The last part of this section illustrates these facts by application examples.

CR80 Highlights:

- **Distributed processing throughout the CR80 computer family**
  - Multiple Central Processors
  - Multiple CPU's in Central Processors
  - Individual Microprocessors in each Peripheral Controller Module
  - Fast separate processor for Interrupt Preprocessing and Data Channel management
  - Multiple Microprocessors handling protocol and logical multiplexing of channels to S-Net and X-Net.

- **Unique multilevel SECURITY features**
  - Privileged instruction set of Central Processors, coupled with Memory Map control and boundary register, prevent unauthorized access
  - Separate SYSTEM/USER state limit data access and process changes and cause interrupt on attempted violations
  - Prevent processes from monopolizing the system resources

- 15 SYSTEM states with most sensitive part of privileged instructions only executable in the highest state
- Non assigned instructions will cause a trap
- General centralized addressing mechanism used whenever object external to a user process are referred to.

- **Fault Tolerancy**
  - NO-BREAK computing supported by numerous unique hardware, software and maintenance features to achieve mean time between system failures in the order of years.
  - Multiple Central Processor incorporation of Peripheral Controller modules providing alternative processing paths.
  - Economic N+1 Central Processor redundancy.
  - Economic N+1 Communication Interface redundancy.
  - Dual Powering of Peripheral Controller modules safeguards against single power failures.
  - Redundant Fan Units ensures sufficient cooling of equipment in the event of Fan breakdown or failure in a mains phase supply.
  - Short mean time to repair ensured by major system components exchangeable from the front with no cable detachment or special tools needed.
  - Extensive Quality assurance and control program during design and production for achieving and maintain the CR80 high level of module reliability.
  - Maintenance and Configuration Processor subsystem supervises Power Supply voltages and environmental conditions and provides reconfiguration of the computer in response to errors reported by on-line diagnostics, self-checks and status reporting.

- **Extensive use of LSI technology**
  - High equipment density achieved by use of RAM's, PROM's, CPU's, USART's, FIFO's, Programmable Logic Arrays and microprocessors.
  - Low power consumption, allowing for forced air cooling of even the largest computer configurations.
  - Very low space requirements of packed computers.
  - High speed based on Schottky-TTL technology.

- **Powerful CPU utilized**
  - Microcycle time 250 nanoseconds
  - 16 bit instructions
  - Internal pipe lining
  - Instruction prefetch
  - Comprises dual Arithmetic and Logic Units allowing up to 3 operand arithmetic operations to be executed simultaneous.
  - Extensive error checking with roll-back allowing instruction reexecution.
  - Designed for multi CPU, multiprocessor environment.
  - Non-mapped and mapped virtual memory capability.
  - Field exchangeable single unit.

- **Reduced bus contention using CACHE high-speed buffer memory**
  - Increases CPU efficiency by 40%
  - Transfer bus bandwidth utilization increased by 50%.
  - No software overhead.
  - Real time consistency with content of main memory in multi-processor environment.

- **Unique TDX/X-Net Local Area Network**
  - Addresses up to 256 devices on local network.
  - Coaxial cable pair allows up to 5000 meters between stations.
  - Multiple ports provide common services to work stations.
  - Megabit serial transfer buses provide essentially unlimited front-end throughput and connectivity.

- **On-line serviceability and extendability**
  - Computers partioned in self sustained physical subunits complete with power supply and cooling.
  - Physical subunits galvanically isolated from each other and interconnected via high speed dual or multiple redundant long distance data highways,omitting ground loops normally limiting size and on-line extension of computer systems.
  - All major modules, inclusive the power supplies and Fan Units, are insertable and exchangeable from the front without special tools.

- On-line exchange and addition of modules without power down provided by electronic power switches and bus high impedancing circuitry in the individual modules.
- Extensive individual module self test at power-on provides immediate visual indication to operator of hardware status.
- Wide range of maintenance and diagnostic programs.
- Early warning of error prone conditions and preventive fault correction made possible by the Maintenance and Configuration microcomputer monitoring power supply voltages and environmental conditions of subunits.

- **Maintenance and Configuration Processor**
  - Stand-alone system Watchdog microcomputer monitors equipment status through physical sensing.
  - Voltage variations of power supplies monitored with A/D converters.
  - Fault Tolerancy computer reconfigurations, based on accumulated on-line diagnostics, selfchecks and status reporting.
  - Distributed monitoring and control of all computer subunits through separate redundant, galvanically isolated connections.
  - Fail-safe switch-over to manual set-up of configuration in case of error in the Maintenance and Configuration Processor itself.
  - Manages the economic N+1 redundancy switch-over of communication lines.

- **DAMOS, Distributed Advanced Multiprocessor Operating System**
  - Unifies multiple mapped virtual memory multiprocessors into a high performance computer (MAXIM and FATOM).
  - Support mapped computers ranging from single Central Processor with 1 CPU and 128K word of Memory, and up to 16 Central Processors each with 5 CPU's and 16 Megaword of memory.

- High efficiency, flexibility and security in real time environment, but also supports software development and batch.
- Virtual Memory management by demand paging, but process swapping is also supported.
- Provides process management, interprocess communication, basic and high level device handling, including interactive terminals, communication lines and file structured backing storage devices.

- Comprehensive suite of software development tools and utilities, the following languages are presently available:
    - Assembler
    - SWELL, the CR80 system programming language
    - PASCAL
    - COBOL

  the following languages are announced:
    - FORTRAN 77
    - ADA.

- **AMOS, Advanced Multiprocessor Operating System**
    - Standard operating system for non-mapped CR80 multiprocessor computer (MINI and TWIN).
    - Supports unmapped computers ranging from 1 to 4 CPU's and up to 256 Kilowords of memory.
    - Aimed at real time applications, but also supports software development and batch.
    - Provides process management, interprocess communication, basic and high level device handling including interactive terminals, communication lines and file structured backing storage devices.
    - Compatible with DAMOS at the level of Input/Output, whereas process management and communication are different.
    - Wide range of software development tools and utilities, the following languages are presently available:
        - Assembler
        - SWELL, the CR80 system programming language

- PASCAL
- COBOL

the following languages are announced:

- FORTRAN 77
- ADA.

- **CR80 Multivendor and standard NETWORKING capabilities**
  - Mainframe Channel interfaces and drivers
    - IBM
    - UNIVAC
    - ICL
  - Network
    - Network Management
    - Transport Station
    - X25 level 3
  - Line Protocols
    - Start/Stop
    - HDLC
    - SDLC (IBM)
    - UDLC (UNIVAC)
    - X25 level 2
    - BSC multipoint
    - BSC point to point
    - BSC transparent
    - CO2 (ICL)
    - CO3 (ICL)
  - Device Protocols
    - TTY
    - BSC 3271 (IBM)
    - 2780/HASP (IBM)
    - 3276/3767 (IBM)
    - 3274 (IBM)
    - 7502 (ICL)
    - TC500 (Burroughs)

CR80 Application Examples

The remaining part of this chapter illustrates by four case stories the versatility of user applications in which the CR80 computer family is applied:

1. Corporate packet switched network incorporating eight CR80 MINI nodes and a CR80 FATOM Network Center with multi-vendor Mainframe support. The network combinedly supports the IBM-SNA and X25 /ISO architectures.

2. Banking network including eight CR80 TWIN based nodes interconnected via a public packet switching network.

3. A CR80 MINI based small front end for ICL computers interfacing to terminals via IBM 2780 protocol.

4. The CR80 Commercial EDP packages on CR80 MINI's for a California Supermarket chain.

Further user application examples can be found in the introductory parts of chapter 8 and chapter 14.

Corporate Packet Network

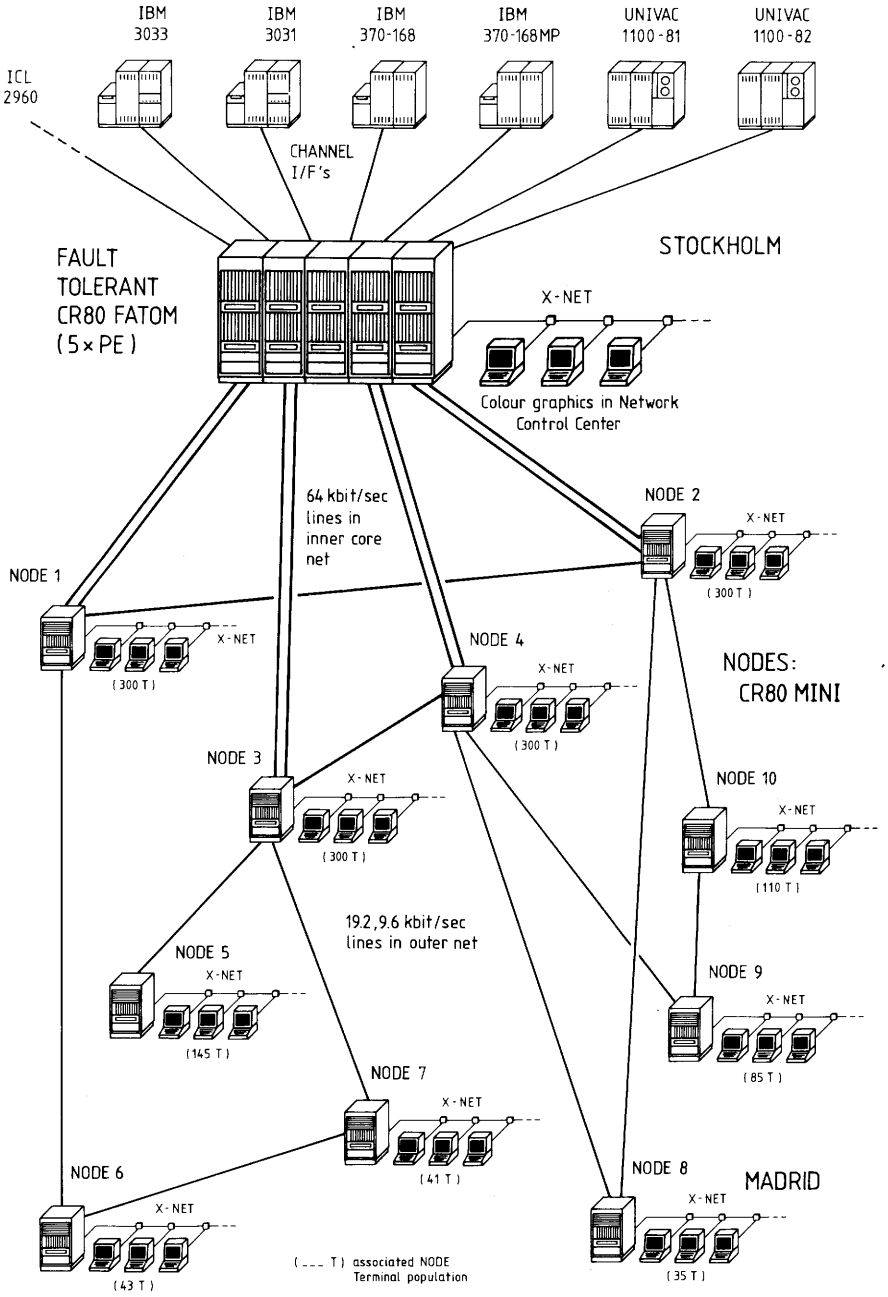
A corporate data communication network (LME-NET) is delivered for a major international Telephone and Telecommunications Company, to cover the need within the organization with regard to data communication between data centres and terminal users.

LME-NET is based on the CR80 FATOM computer and numerous CR80 MINI nodal computers, the first phase shown in figure overleaf consists of:

- a network centre,
- a fault tolerant host interface processor system for connection of IBM and UNIVAC computers,
- 10 switching nodes where traffic is collected and directed to the receiver,
- a number of leased lines between the nodes, eight of which are in Sweden, one in Copenhagen and one in Madrid.

In the later phases, the network will be enlarged with:

- more network control centres, which will enable certain distributed control parts of the network,
- more geographically distributed host interface processors, and optional interfaces to other host mainframe types (e.g. ICL),
- connection via satellite to new nodes (e.g. in Brazil).



Corporate packet switched net with multivendor support
(LME Net, phase 1)

The LME-NET architecture is based on the following concept:

- A general standardized transport facility is provided. The network follows international standards for packet switch data networks, as defined by CCITT in the recommendation X.25. This enables later connection to public networks and ensure the adaption of LME-NET to future standards.

- Existing Mainframes and terminals will be connected to the general network by means of mechanisms in the network which do not require modifications of the existing system.

The above enables the layered structure of LME-NET following recognised principles of system construction in general, and network construction in particular (acc. to ISO's seven-layer model for network: Open Systems Interconnection Reference Model).

LMENET has the following functions:

- a complete monitoring and control of the network independent of host computers connected,

- emulation of a network, complying with IBM's Systems Network Architecture (SNA) in order to establish communication between the IBM user programs and the SNA terminals and certain non SNA terminals.

- emulation of a network complying with UNIVAC's Distributed Communication Architecture (DCA) which enables a communication between UNIVAC user programs and terminals,

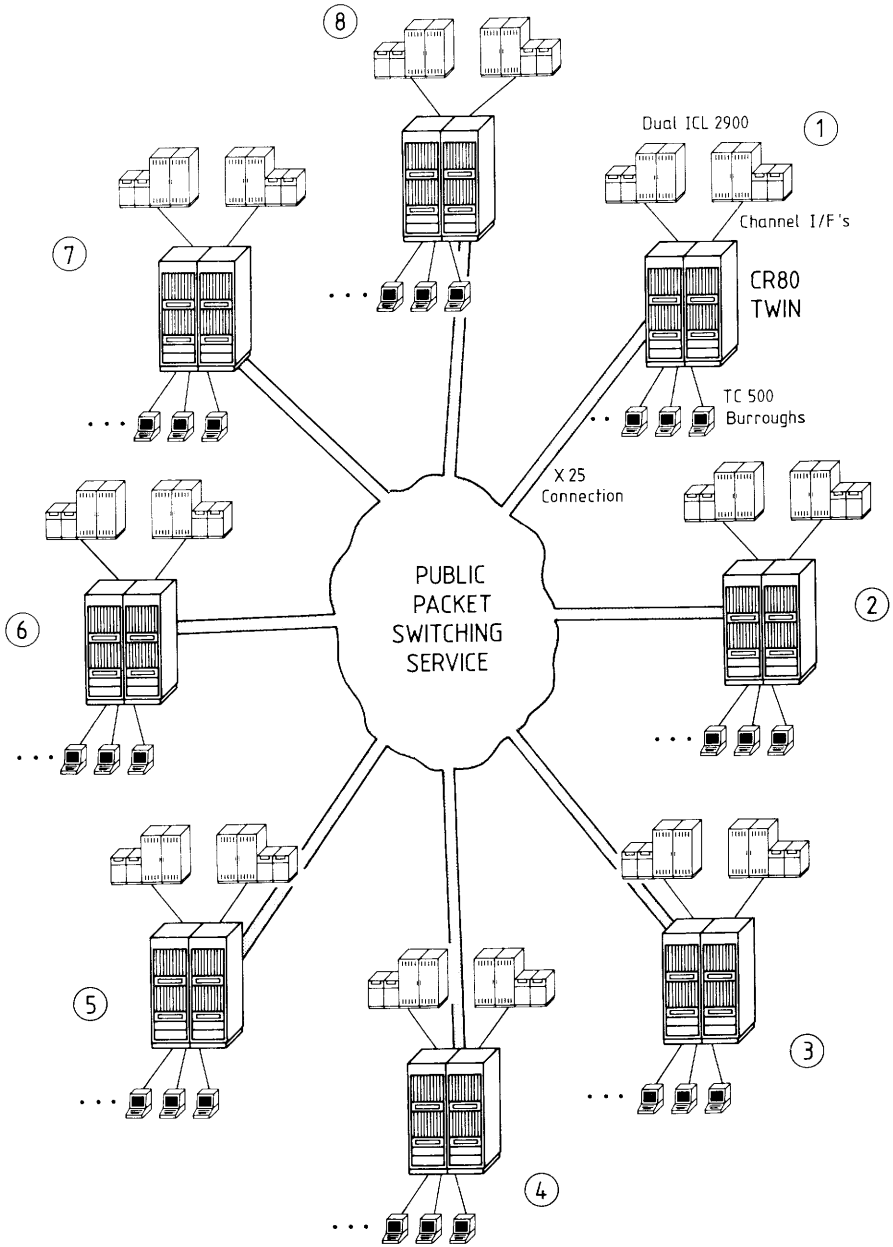
- direct program to program communication,

- different traffic types with different resource requirements,
  - dialogue/interactive traffic
  - batch traffic,
  - transparent traffic
  - terminal to terminal message traffic.

Bank Network

A bank network has been supplied to a major international bank.
This network, consisting of eight CR80 TWIN computers at geographically distributed sites and interconnected by a public packet switching network, has the following features:

- One of the CR80 TWIN computers is nominated to provide Network Management Services (NMS).
- The rest of the CR80 TWIN computers have a number of Burroughs terminals and upto two ICL 2900 mainframes attached to each.
- All the network users - the terminal operators and the mainframe applications - have access to one another according to the "open" systems architecture (ISO).
- Application programs are considered as distributed resources which are shared by the operators.
- A distributed resource sharing scheme is used to allocate the application resources.
- Closed User Groups (upto application level), priority and password schemes are used to control the use of the above mentioned resources.
- Local network control is provided to operators at each CR80 TWIN site.
- Remote network control is provided to NMS-operators.
- The NMS is designed to be the normal network operating centre, but the network operation is not dependent on it.



Major bank packet switch network

- The integrity of the data transmission across the public network is guaranteed by a data transport system utilizing a standardized transport protocol.

- Although the network level routing is performed by a public network in the given example, it is possible (optionally) to include routing algorithms and the management of routing information between the CR80 TWIN's such that they may be connected directly via leased (X75) lines and thus act as intermediate nodes.

In the following some of the redundancy features are discussed with reference to the detailed block diagram of one of the CR80 TWIN computers shown overleaf:

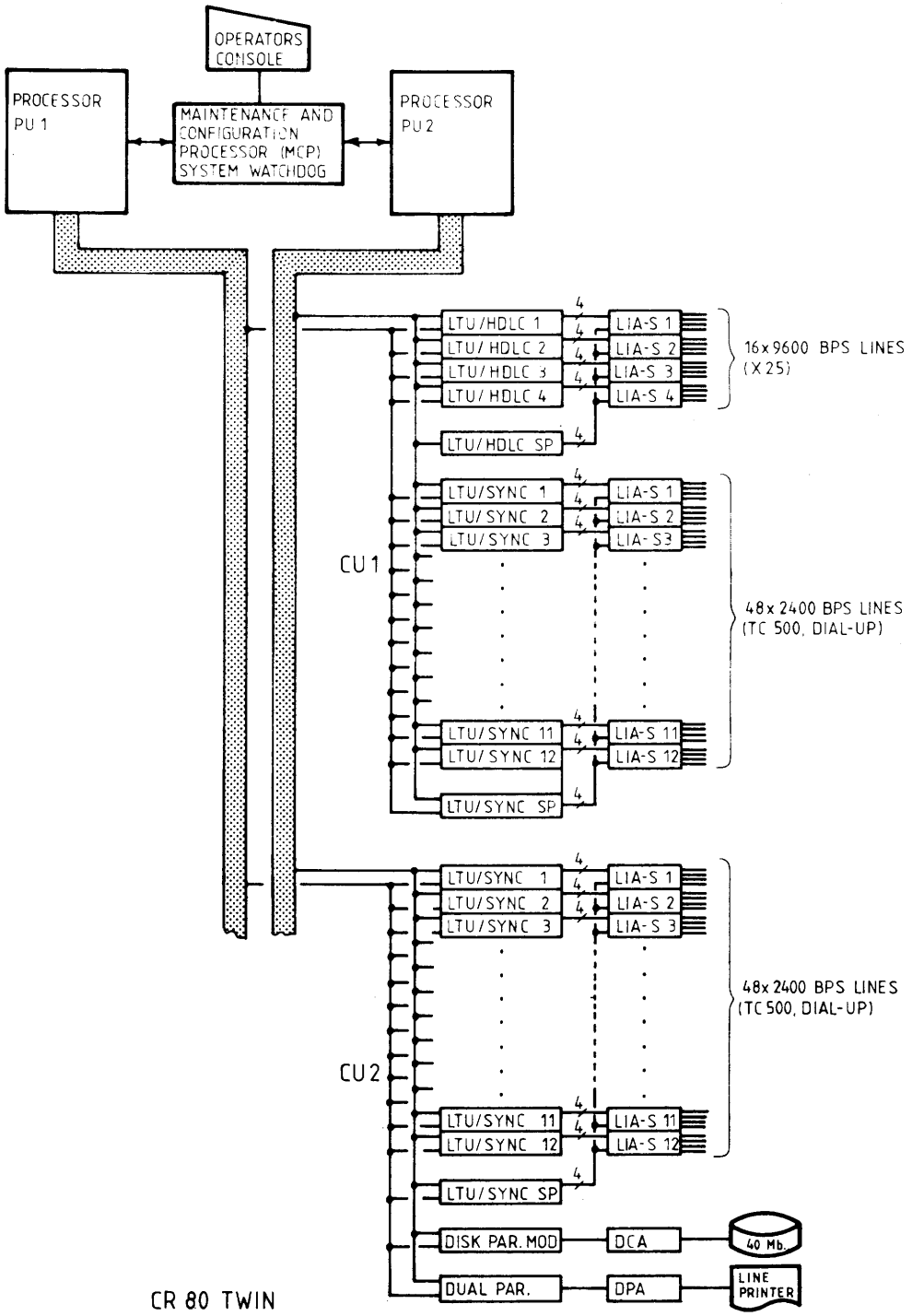
- Dualization aspects of the CR80 TWIN's allow the two processors to operate in either a load sharing mode or in active + hot standby mode. The changeover to the hot standby is without manual intervention.

- All data transfer to and from communication lines is via micro computer based Line Termination Units (LTU).

- The protocol firmware to be used on each LTU is downloaded from the CR80 TWIN.

- N+1 redundancy is provided by providing a spare LTU for each group of 8-12 LTUs. Line diagnostic software automatically detects a faulty LTU and switches in the spare LTU. The spare LTU is then automatically loaded with the correct protocol and line configuration to continue the service provided by the (now) faulty LTU.
  When the faulty LTU is repaired and replaced, the software automatically switches the repaired LTU back in action.

- Networks similar to this example have also been installed at a number of customer sites in the U.K.

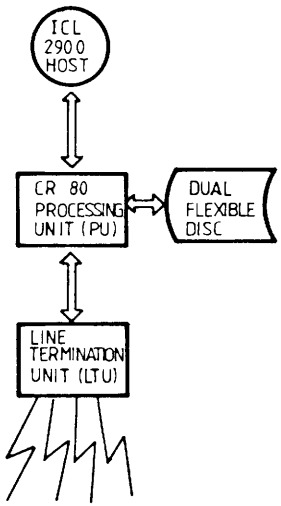


CR 80 TWIN
H/W Configuration of Front End Processor

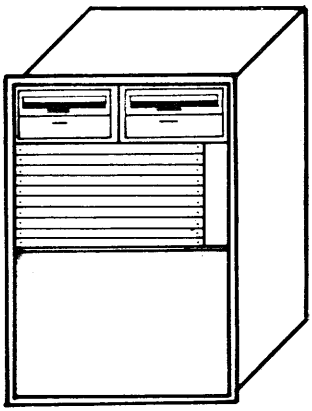
Small Front-End System:

The figure overleaf shows an example on a CR80 MINI based small front-end system, of which six have been delivered by Christian Rovsing A/S for use by a public utility company.

- The front-end has a single PU with only one CPU.
- A host interface module interfaces the system to an ICL mainframe.
- A single LTU provides access to four communication lines running the IBM 2780 protocol.
- All incoming messages on a line are routed to the mainframe and vice versa.
- There is no rerouting of messages between communication lines.
- Dependent on the complexity of the processing of the individual messages through the system, this type of system handles 5-10 transactions per second.
- Further throughput is gained by increasing the number of CPU's in the system.



MULTIDROPPED
COMMUNICATION LINES
(IBM 2780 PROTOCOL TO TERMINALS)



CR 80 MINI (model 831)

One of six small CR80 MINI Front End Computers used by a public utility company

Commercial EDP System

Overleaf is shown one of two CR80 MINI Commercial EDP Computers for use by a large California Supermarket chain. The systems are based on the standard commercial software packages (CROPS, CRMINI) available with the AMOS operating system. By adding more CPU's to the CR80 MINI the system can be expanded up to 4x300 M Byte disc capacity and 128 workstations.

The CR80 MINI, model 801 handles the following amounts of data:

- Up to 50 Supermarkets.
- Up to 10,000 reordering transactions per week for each store.
- Up to 40,000 items.
- Up to 5,000 credit customers/suppliers.
- Up to 8,000 general ledger accounts.
- Maximum 20,000 open purchase order lines at any given time.
- Maximum 100,000 main grocers invoice lines per week.
- Statistics with specification to Item Number levels for the last 4 weeks and twelve 4-week periods backwards.

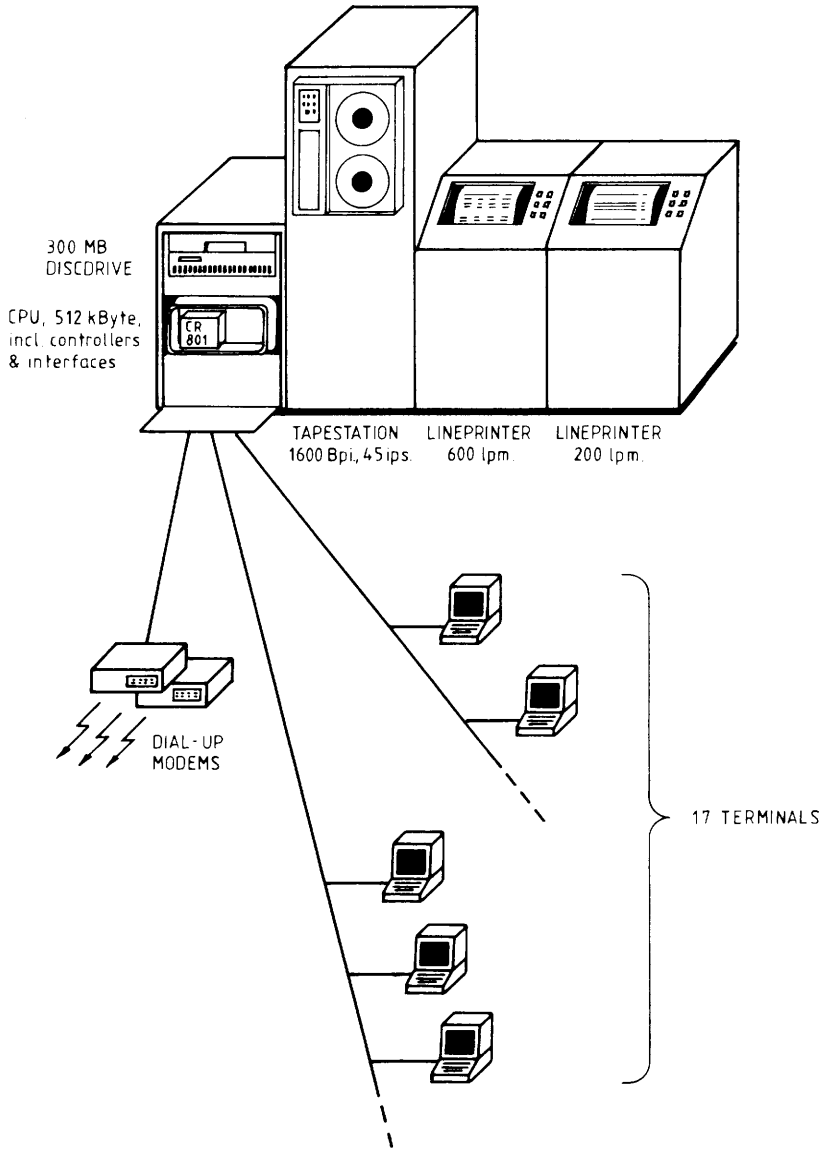
Data Security:

- The system is guarded against unauthorized use by user code and passwords. Through password codes, users have access to specific routines, but not other.

- In case of errors in the computer or in the data base, the system is protected against loss of data by the automatic recovery procedure. Recovery is based on logging all updating transactions and copies of data bases.

Program Language:

- The commercial package is coded in ANSI Cobol. The Cobol compiler is an industry efficient 2 pass compiler, fulfilling the American National Standard x3.23-1974 Level 1, with almost all Level 2 features.



CR80 MINI (model 801) in commercial EDP application
for a large U. S. Supermarket chain

The CR80 AMOS and CROPS software is an advanced general purpose multitasking transaction oriented system, which provides communication line and data base control functions in local and distributed terminal based systems, with the following features:

- Communication Lines and Data Base Control
- Local or Distributed Terminals
- General Purpose Virtual Terminals
- Low Transaction Overhead
- Index Sequential File Access
- Comprehensive Print Spool Facilities
- Extensive Recovery Procedures
- Data Security
- Broadcast Facility
- Send Transaction Direct
- Send Transaction Indirect
- Development Tools
  - TOS, Interactive Terminal Environment
  - CMI, Command Interpreter
  - Text Editor
  - COBOL Compiler
  - SWELL Compiler
  - PASCAL Compiler
  - Linkers
  - Debugger
  - Print Spool
  - File Utilities: Create, Delete, Copy, Dump etc.
  - Screen Picture Generator

The CR MINI Commercial package contains the following modules:

- Order Entry
- Invoicing
- Direct Deliveries
- Purchase Management
- Inventory Management
- Stock Taking in Stores
- Stock Taking in Warehouse
- Sales/Cash Figures
- Invoice Control System (Main Grocers)
- Accounts Payable
- Retail System
- Accounts Payable
- Fixed Assets Depreciation
- Payroll
- General Ledger
- Report Generator/Statistics

The modules are described in more detail in the following:

- Order Entry

  Orders from the stores are recorded on and transmitted by handheld terminals to the central computer. Here the orders are optimized by item and split by store.

- Invoicing

  After items have been picked at the warehouse, possible corrections to volume are entered directly into CRT and an internal invoice produced. This can also accompany the shipment to the store.

  Alternatively, the system will produce a weekly invoice for each store covering all the supplies delivered that week. This includes own products and those from direct vendors and main grocers.

- Direct Deliveries

  Invoices from vendors who have delivered directly to the store are controlled by the head office through a simple but effective validation process.

  On request, the CRT will display an open order of items and prices from a specific vendor. Only the number of items will be entered after which the system generates an invoice copy that is matched against the original invoice. Deviations are printed out and sent to the purchasing department for approval.

  Validated invoices are sent automatically to accounts payable. The store's account and vendor account are immediately updated.

- Purchase Management

  Orders for delivery to the warehouse are registered in the purchase management system as open orders. In this way the buying department has a clear picture of items in stock and in transit.

  When merchandise is received at the warehouse, it is registered in the CRT which automatically removes it from the open order file and updates warehouse stocks. Vendor invoices are registered on receipt without affecting the status of the warehouse since this is updated on physical receipt of goods.

- Inventory Management

  Inventory management control checks quantities and prices against the original purchase orders. The system ensures that you are paying for what was received and at the right price, and that merchandise received is in accordance with original orders, or other criteria, depending on source of delivery.

For warehouses, there is an automatic inventory control with automatic re-ordering levels, inventory status list and price lists. An item inquiry request or price correction can be done at any time in an on-line mode. with reference to certified grocers items, an automatic price change control routine will be built into the system and produce a deviation report.

- Stock Count in Stores

  Stock counts are recorded on handheld terminals and transmitted on-line to the central computer. The system then produces an overall stock-taking value report by store/department/category which is then transferred to the retail system and general ledger.

- Stock Count in Warehouses

  Stock counts are recorded on handheld terminals registering quantity per item. The computer then produces stock count reports and itemized deviation lists. Inventory can then be adjusted with new in-stock figures.

- Sales/Cash Figures

  Sales and cash figures are transmitted by hand terminals or a number of p.o.s. terminals to the central computer. For stores with scanners a scan movement catalogue will be produced once a week.

- Invoice Control System (Main Grocers)

  The open order file for main grocer orders is adjusted against receipt of delivery notes, and then controlled automatically against grocers invoice tape. A deviation list is produced for controlling prices and quantities.

- Accounts Payable

  The accounts payable system facilitates cash flow management and control. Via the purchase system cash forecasts can be made. The system keeps track of all invoices, payment terms, and check lists.

  The system will write checks, print statements, labels, open posting lists, age analysis and movement lists of vendors on-line.

- Retail System

  The retail system calculates retail prices and profits by store/department based on cost price of delivered items, refunds, and shrink percentage.

  The retail system also calculates stock values by store for entry into the general ledger until actual stock count figures are entered to replace stock values.

- Accounts Receivable

  Accounts receivable covers two functions in the system. One is to give a statement by store/department for purchases during a period. This gives management the opportunity of monitoring stores daily or periodically.

  The second function is to keep track of regular credit customers and their payments. The system will calculate any penalty interests, produce dunning lists, and age analysis reports.

- Fixed Asset Depreciations

  This module is in accordance with the American practice and standards. Depreciations are input into the general ledger.

- Payroll

  Payroll system with Input to general ledger.

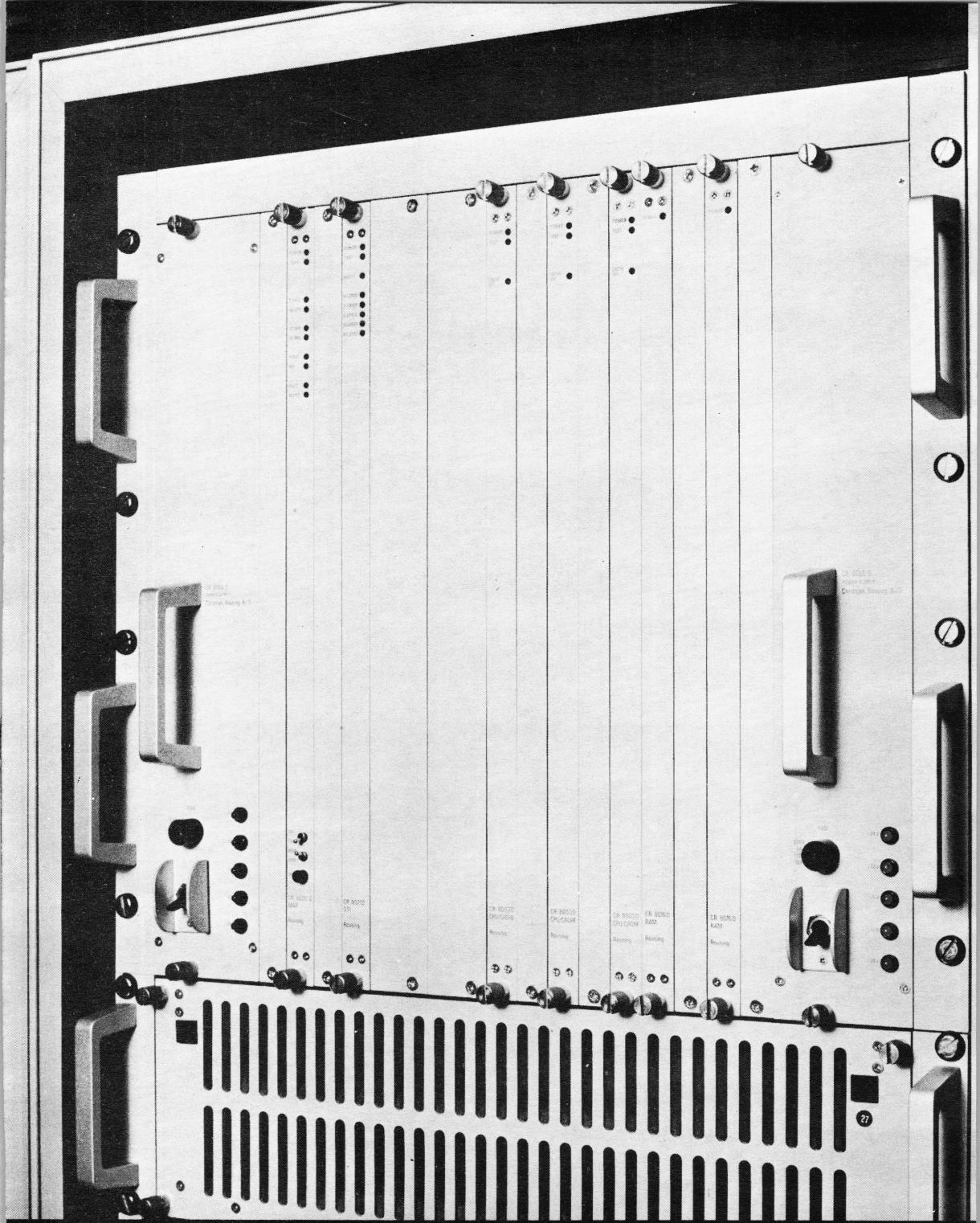
- General Ledger

  The information flow from all the other modules is available to the general ledger. The system can produce status reports for top management. These include trial balances, profit and loss accounts, cost allocation reports, collation of results and budgets this month/year, and last month/year. The system gives information on stores as profit centers, in totals, or by numerous criteria in key areas.

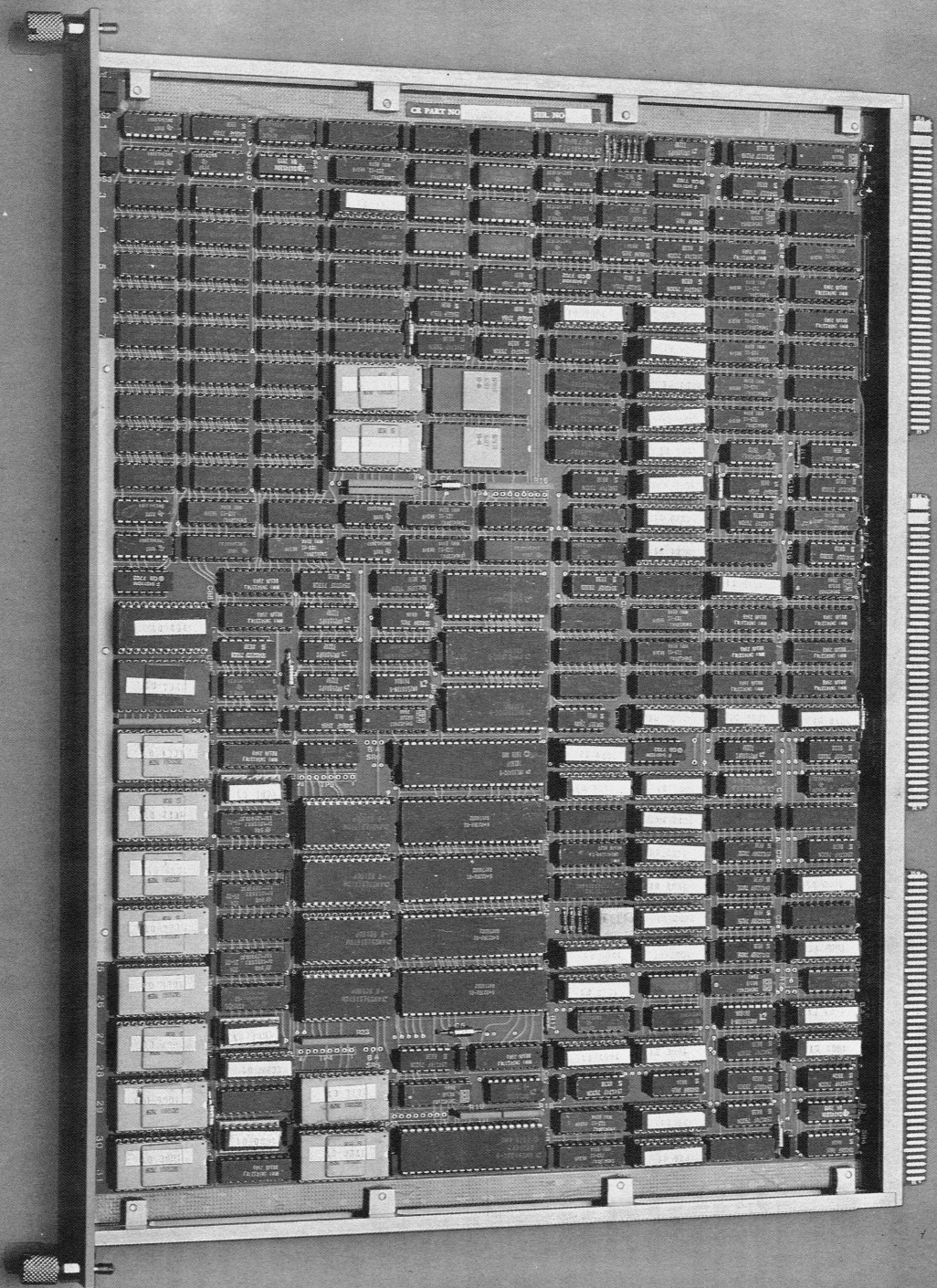
- Report Generator/Statistics

  Data from sales order entry and purchase management are stored in the statistics file under a number of customer/supplier and item criteria.

  Presented in 5 pre-defined print layouts, information can be retrieved when required from the statistic file. Sorting, printing, and totalling is fully user driven.



R80 FATOM Processor Unit containing 3 CPU/CACHE's, 0.5 Megabyte Memory, AP and S-NET interface (STI). The PU is extendable by plugging in further CPU/ ACHE's and Memory.



The CPU/CACHE plug-in module on a single multilayer printed circuit boar includes a high performance bit slice Schottky-TTL Central Processor Unit, kilobyte fast associative cache buffer memory, dual bus interface, and circuitry fc extensive self-test.

2. CR80 MAXIM and FATOM Computers

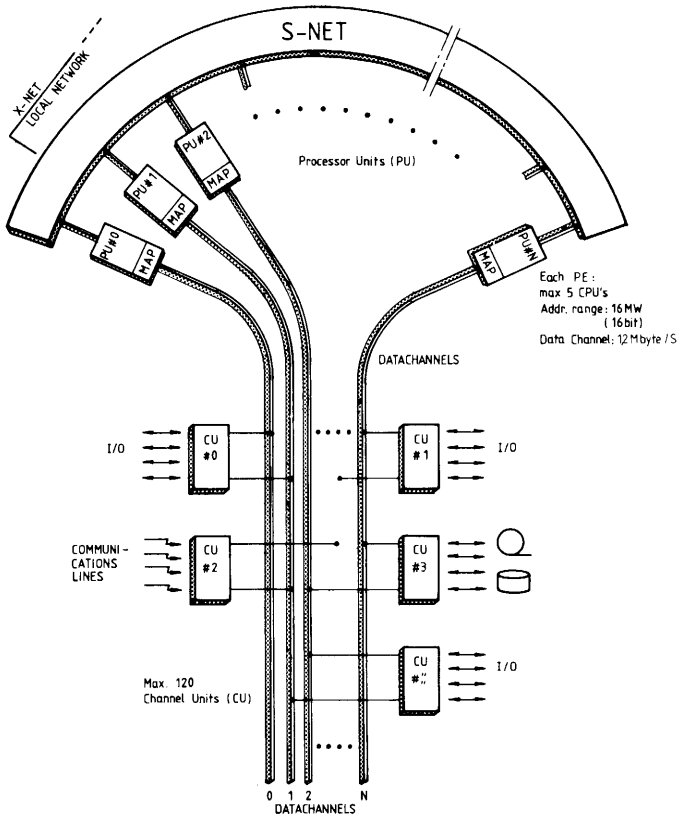
2.1 General

Christian Rovsing A/S with the CR80 MAXIM and FATOM virtual machines has introduced a new and powerful architecture for ultra-reliable, easy to maintain and modular fail safe computers. The high speed memory mapped multiprocessor computers have been designed to provide modular growth in processing power and memory requirements to cope economically with the requirements of:

- General purpose computer systems
- Front end processors
- Packet switches
- Concentrators
- Process control
- On-line systems
- Terminal systems

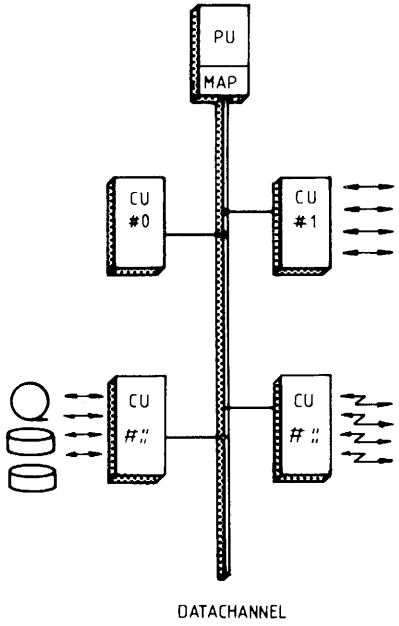
The illustration overleaf shows that the CR80 FATOM computer tightly couples Processing Elements (Multiprocessors) together via the S-NET, the Processing Element (PE) being constituted by a Processor Unit (PU) and elements of Channel Units (CU's) attached to the interconnecting Data Channel. Each peripheral connects to two Processing Elements (PE's), one PE being the active processor for a connected peripheral, the other the back-up processor. Also it is seen that the CR80 MAXIM (Memory mapped Maxi-computer) is the single Processing Element (PE), non-redundant subset of the CR80 FATOM (Fault Tolerant multiprocessor), otherwise they have identical high performance characteristics.

The CR80 FATOM fault tolerant computer differs from other computers (large, medium or small) in that it, based on a unique distribution of its memory providing nearby unlimited processing power, up to more than 30 Million instructions per second (MIPS) in a 16 PE configuration, together with minimum added hardware to achieve its "self repair" features and 256 Mega word maximum memory size. Extensive hardware checks have been incorporated throughout the CR80 architecture, supporting integrity and security in execution of both application and system programs, ensuring that erroneous interaction among users, as well as with the system software is prevented.



CR80 FATOM

FAult TOlerant Multiprocessor



CR80 MAXIM

This is extremely important during software maintenance and development, once a fault tolerant system has been brought operational, as well as facilitating the initial software development and debugging.

The CR80 architecture and DAMOS system software supports modularily the total spectrum of virtual memory machines, from the 0.6-2.0 MIPS MAXIM multiprocessor computer with one or more CPUs, up to the more than 30 MIPS, N+1 redundant FATOM computer, incorporating the cost effective approach of only having 1 single spare unit, capable of backing up for any of N working units. The CR80 can be upgraded in the field, often without stopping operational use, due to its on-line maintenability and unique galvanic isolation between system elements at the card-magazine level.

A CR80 Processing Element (PE) constitutes either a uni- or multiprocessor computer with from 1 to 5 CPUs (.6 to 2 MIPS). The CR80 FATOM connecting 16 Processing Elements (PE's) together via the extremely fast S-NET (up to 512 Mbit/sec.) into a tightly coupled multicomputer with more than 30 MIPS capability. In addition all lower levels of input/output processing is distributed to Peripheral Processors in the Channel Units (CU), this further enhances the CR80 above the simple accumulated processing power of the CPUs.

The Peripheral Processors communicates with PE's through one port of a triple ported memory, the two other ports allowing for this memory being part of the address space of two Processing Elements (PE's), which ensure an alternative Processing Element, in case of a Processing Element (PE). failure. The Peripheral Processor and associated three ported memory is physically located in a Peripheral Module housed in a CU.

The CR80 computers also gain their strength from high speed intelligent multiplexed Direct Memory Access (DMA) channels between the distributed memory in PUs and CUs. The imbedded channel processors (S-NET & INTRA MEMORY) with minimum interruption of the CPUs autonomeously handle and ensure the integrity of hundreds of simultaneous active logical channels between processes.

The CR80 FATOM basic system philosophy is to achieve N+1 redundancy on all levels, both for Processor Elements and Peripheral Interfaces. A unified system approach to software in a redundant system, relieving application software as far as possible of mechanisms and functions necessary for fault tolerance, moving these to the system S/W. Thus the CR80 FATOM Computer is designed to have no single points of failure on a system basis, this includes all parts of the system: Processors, busses, I/O devices, power supply, cooling and software in order to achieve a continously available no-break computer. The on-line maintenance features, allows any failed module to be exchanged and tested, without interrupting system operation.

Furthermore the CR80 modular packaging and integration system, ensures the capability for expansion of a CR80 FATOM Computer to virtually any physical size, using only a few standard types of modules and cables, as well as achieves the cost efficiency of both the single and fault tolerant CR80 Computers.

2.2 System Organisation

2.2.1 System Overview

The CR80-memory mapped computers are of modular construction, allowing the system architect to configurate Computer Systems with a performance ranging from a single (MAXIM) computer with from 1 to 5 CPUs and few peripherals up to a complex failure tolerant (FATOM) computer system with up to 80 or more CPU's and nearly unlimited amount of peripheral equipment and communication lines, by use of few standard items.

In the CR80 Functional Overview shown overleaf (Fig. 2.2.1-1), the basic elements are readily identified as S-NET, TDX (X-Net), Processing Elements (PE's) and Peripheral Processors (PER.PROC.), Data Channels and two types of basic units, Processor Units (PU) and Channel Units (CU).

Each Processing Element (PE) looks to the user as a multiprogrammable multiprocessor (up to 5 CPUs) virtual memory (16 Megawords) and demand paging. Processing Elements (PE) send messages to other Processing Elements (PE) Memory and receive messages in own Memory via the S-NET. As all data transfers via the S-NET is through both the Memory Map of the source PE and the destination PE, full hardware protection against unintended interference between PE's is ensured.

A Processing element (e.g. PE no. 1) is physically implemented in two types of crates (card cages), the Processor Unit (PU) containing all address sourcing devices (CPU's and DMA's) and the first Megaword of the PE-Memory, and a number of Channel Units (CU's) containing additionally up to 15 Megaword of PE-Memory. the Channel Units (CU's) furthermore, contains the Peripheral processors interfacing peripherals (e.g. Disc, Tape, terminals, commnication lines etc.) to the Processing Element. The Memory Bus of the PE is divided into three parts (P, C and D) with the Memory Map centrally placed, in order to optimize processing and access to Memory.

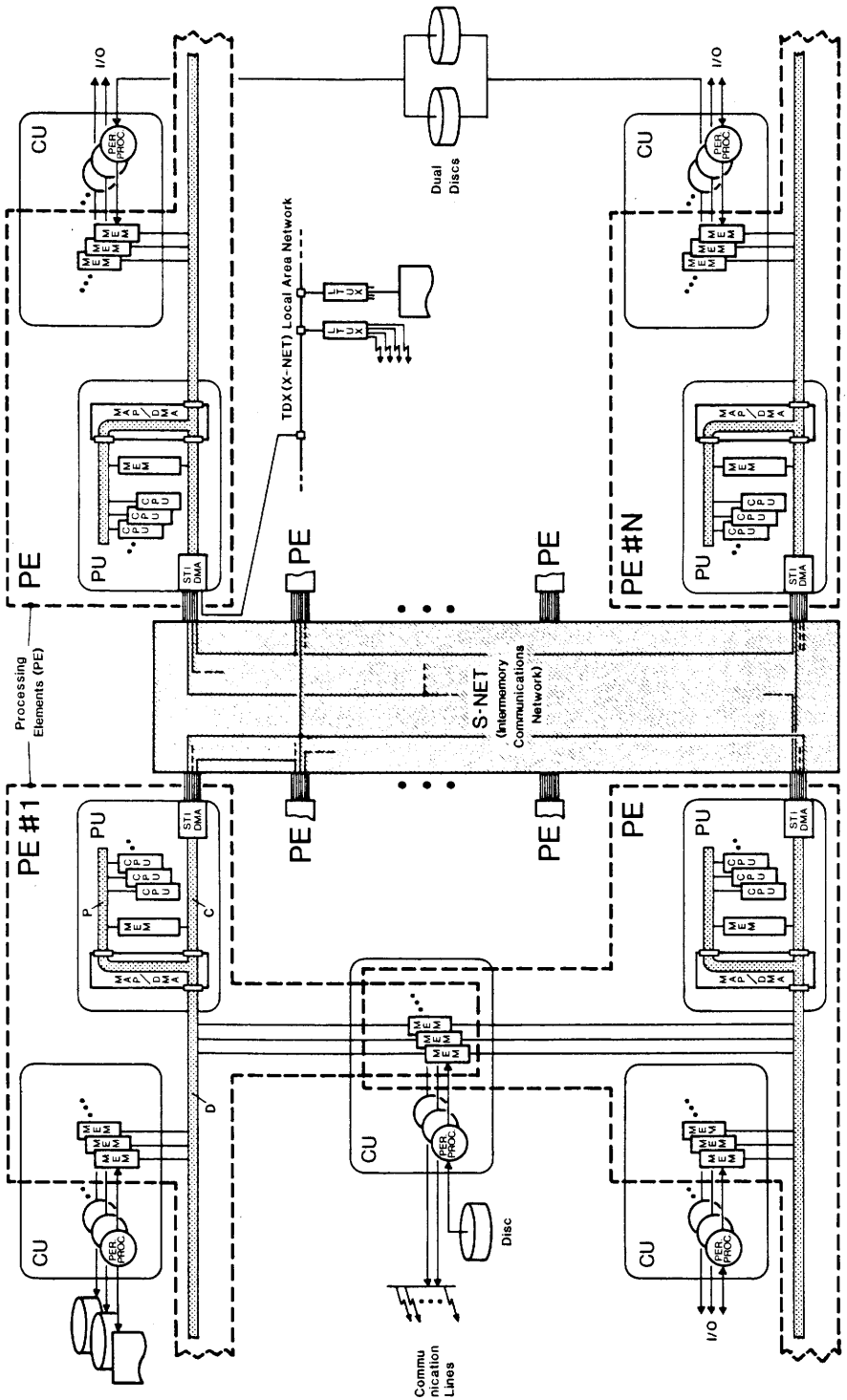


Fig. 2.2.1-1 FATOM Functional Overview

The P-Bus (P) is reserved for the Central Processing Units (CPU's). The C-Bus (C) is utilized by Direct Memory Access (DMA) devices, e.g. the S-Net/TDX Interface (STI) DMA's and the INTRA MEMORY DMA positioned in the MAP module. The memory extension bus (D), commonly named the DATA CHANNEL, is shared by CPU's (on P-Bus) and DMA's (on C-Bus) for access to PE-Memory positioned in CU's. The memory extension bus (D) gets its channel like characteristics, due to the intelligent C-Bus DMA's and their associated embedded processors. Only passive PE-Memory is attached on the bus (D) itself. The P-Bus and C-Bus operates independently of each other (although multiplexing the use of the Memory Map). This allows processing in the PE by the CPU's to continue without interference during bulk data transfers with the S-Net, TDX (X-Net) or within the PE-Memory.

Peripherals (Disc, Terminals, communication lines etc.) are attached by Peripheral Processors (PER. PROC.), that performs distributedly the I/O processing associated with the specific types of attached peripheral devices, and directly communicates Data and status/control messages to PE-Memory. The parts of PE-Memory accessible by Peripheral Processors is compartmentalized. A Peripheral Processor can only access its own compartment, and not that of another Peripheral Processor or parts of PE-Memory allocated for general processing. The combination of a Peripheral Processor and compartmentalized memory is defined as a Peripheral Module. This insures integrity and security of Input/output, as well as each Peripheral Processor having its own path to PE-Memory, results in omission of the restrictions and speed degradation of the commonly used multiplexed I/O access to memory. It is seen from the foregoing that three distinct levels of multiprocessing are found in the CR80:

- Processor Elements (PE level)
- CPU's within a Processor Element (CPU level)
- Peripheral Processors (PER. PROC. level)

Physical Memory can be incorporated into one or two Processing elements as shown in the CR80 MAXIM and FATOM Memory layout overleaf (Fig. 2.2.1-2). This allows for Bulk Memory or Compartmentalized Memory associated with Peripheral Processors, to be part of the memory space of two PE's. As both PE's can process data found in the common part of their memory, this provides for the Fault Tolerance of the CR80 computer in case of failure of a PE. It enables other PE's to take over processing associated with common parts of their Memory (Bulk or Compartmentalized).

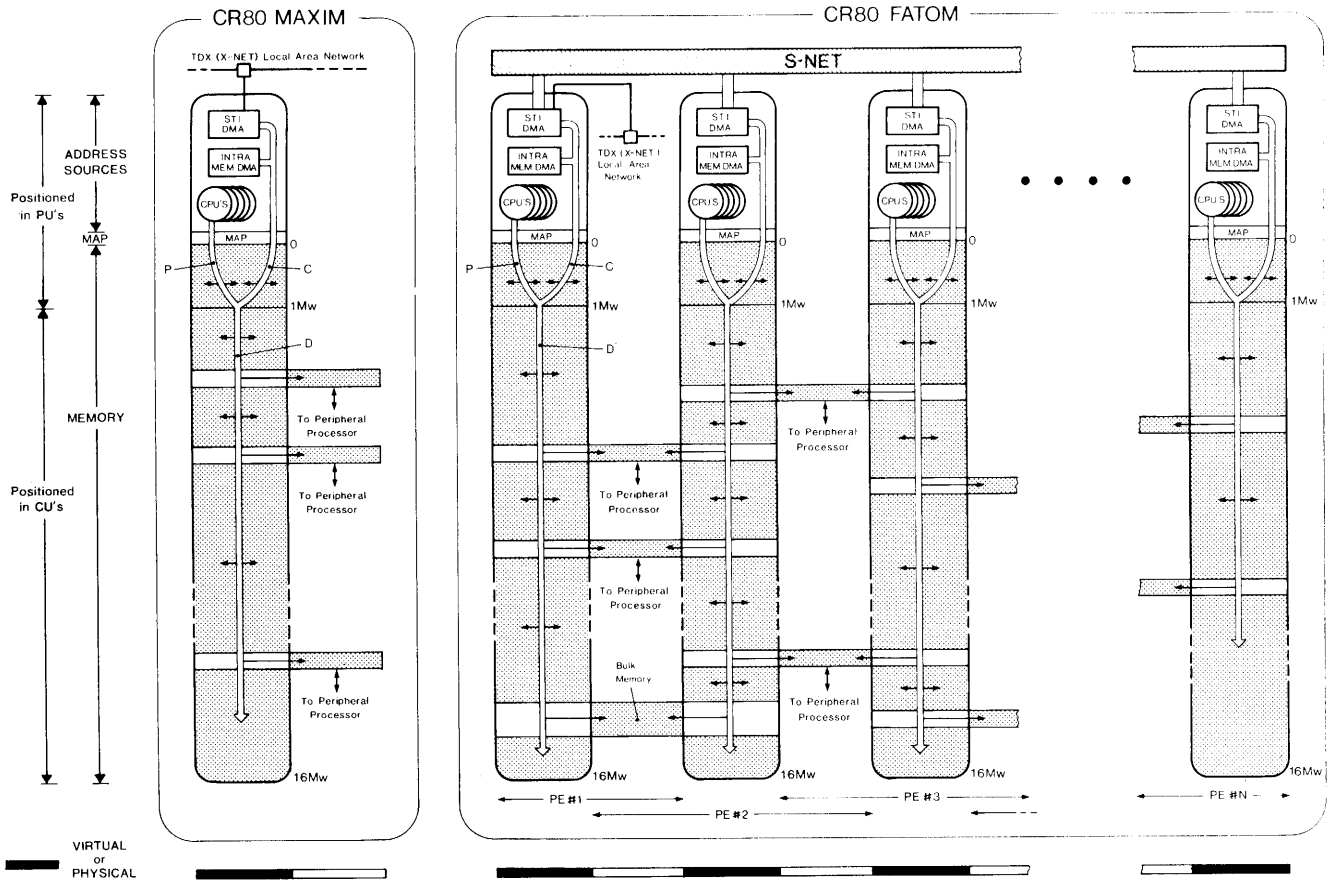


Fig 2.2.1-2 Memory Layout of CR80 MAXIM & FATOM Processor Elements (PE's)

In systems where a Peripheral Module group in a CU, can not tolerate a failure, the N+1 redundancy principle is implemented by having a spare Peripheral Module available in the CU. This spare Peripheral Module takes over the operation of any failed Peripheral Module by switching of the physical peripheral device interface to the Peripheral Module. This is controlled by the Maintenance and Control Processor (MCP) described in a later section.

The S-Net (Intermemory Communication Network) provides high-speed transport of data between Memory of Processing Elements. Each Processing element interfaces to the S-Net with from 1 to 32 coaxial twisted pair cables (SUPRA-BUSES). Galvanic isolation via transformer interface to the SUPRA/-TDX Interface (STI) DMA's, avoids ground loops between Processing Elements. The information transfer is multiplexed on the twisted-pair cables, each carrying 16 Megabits serial transmission under packet protocol protection which ensures error free transmission. The Processing Element interface to the S-Net thus is modularly expandable, by adding SUPRA BUSES, providing a port to up to 512 Megabits of S-Net traffic (32 x 16 Megabit). The S-Net achieve high system reliability and provides multiple redundancy, in that traffic on a failed SUPRA BUS automatically by the protocol is distributed to the other SUPRA BUSES. A Processing Element can communicate with up to 15 other Processing Element via the S-Net. The S-Net/TDX Interface (STI) DMA modules alternatively, or mixed with SUPRA BUSES, provide interface to the TDX (X-Net) Local Area Network (LAN) for connecting to Peripheral devices (Terminals, Printers, Process Control, Communication Lines etc.) and other Processing Elements, within an area of up to several square Kilometers. The TDX (X-Net) Local Area Network is described in detail in chapter 8 and 14 of this handbook.

Interrupt handling within a Processing Element is done centrally by the Interrupt Preprocessor found in the MAP module. The Interrupt Preprocessor receives interrupts transmitted serially from C-Bus DMA's, Timers, Peripheral Processors etc., queue the Interrupts and compare them with interrupt masks and CPU priorities. The CPU is only notified CPU via a direct notification line, when an actual context switching is to take place, thereby relieving the CPU's of tedious and time consuming Interrupt handling.

2.2.2 Processor Unit and Channel Units

The following sections describes PU and CU packaging and organisation.

2.2.2.1 CR80 Modular Packaging

As for the processing system design, great emphasis has been put on Failure Tolerance and modularity of the packaging, cooling and Power Supply sub-systems.

The CR80 modular fault tolerant computer system is assembled using standard modules (printed circuit cards) housed in Processor Units and Channel Units (Card Cages). The Units are interfaced by galvanically isolated transfer buses, structured as shown below (figure 2.2.2.1-1) and described in the following.

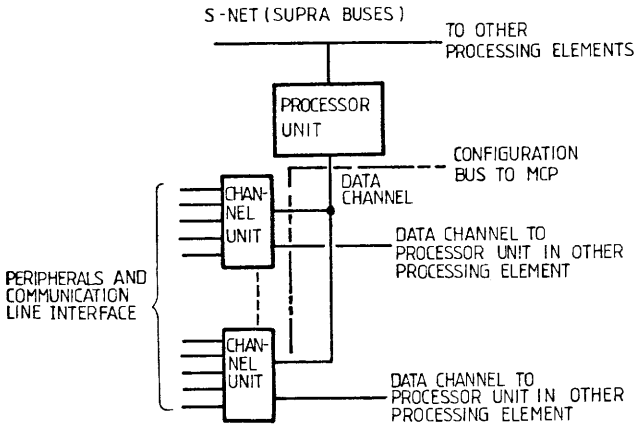


Fig. 2.2.2.1-1
Processing Element, Units and Buses

Units are housed in 19" Crates (Card Magazines) for installation in standard 19" Racks, as shown overleaf (Figure 2.2.2.1-2). A Crate contains a 25 slot Front Magazine for insertion of up to 17 Printed circuit card modules and 2 Power Supply modules, the two upper rows of connectors is each interconnected by multilayer printed circuit buses, while the lowest row of connectors are connected individually via flatcables to corresponding connectors in the Rear Magazine. The 19 slot Rear Magazine, which can be pivoted down for access to Crate internal cabling, holds Adapter modules providing the interface and cabling to external devices (S-Net, Peripherals etc.) for their corresponding Front Module. Also a number of slots is provided outside the Rear Magazine, at the rear of the Crate for insertion of bus termination cards and interface cards to the Data Channel bus. Keeping all external cabling at the rear of the Crate, allows all front modules (CPU, RAM, Peripheral Modules etc.), inclusive the plug-in Power Supplies, to be exchanged quickly without use of special tools.

Below each crate (PU or CU) in the CR80 system is installed an exchangeable FAN Unit, which by forced air cools the modules in the crate. To ensure continuous air flow, the FAN unit is redundantly constructed with the airstream being provided by two sets of blowers, each being powered from different Mains phases, and each with a capacity sufficient for cooling the entire crate over a prolonged period of time. This ensures the failure tolerance of the FAN unit, both against a Mains phase falling out and mechanical breakdown of a blower.

One, or two power supply modules operating in parallel, are installed, in each PU crate dependent of the required power consumption. A power supply failure in the PU will cause the PE to stop processing, but it will not influence the system operation, as processing of the failed PE will be taken over by the remaining operating PE's.

In each CU crate two Power Supplies are installed, each backing up for the other in supplying the modules installed in the crate - distributing power via separate Buses. This power scheme ensures that a single power supply can fail without influencing the operation of the modules in the CU crate due to the special Power Supply ORing-circuit in each of the modules. The power ORing-circuit contains a current limiter which ensures that a short in a module will not draw excess power from the power supplies, and thereby interrupt the

CR80M PROCESSOR UNIT & CHANNEL UNIT

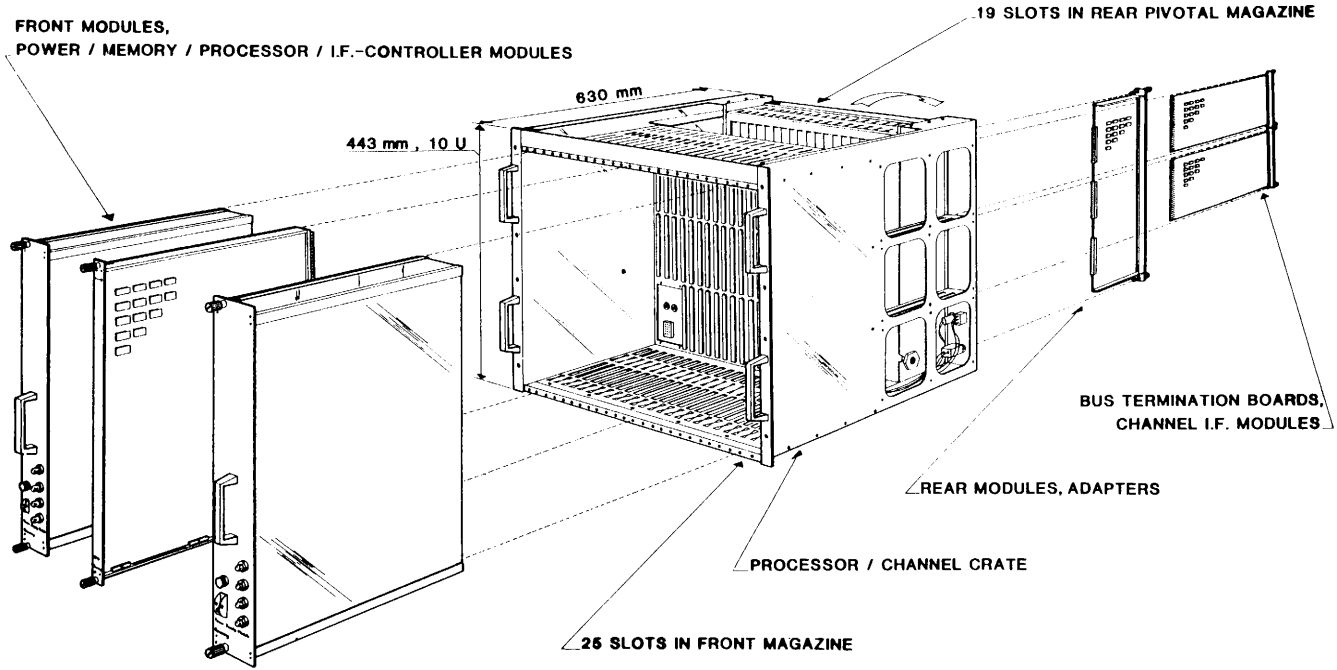


Fig. 2.2.2.1-2

operation of other modules in the crate.

A second function of the Power ORing-circuit is, in combination with a slightly shorter pin in the interface connector between the Peripheral Module and the busses, to allow on-line replacement of a module in an operating CU-crate. The shortened pin will disconnect first and connect last, when a module is removed or inserted, as this pin (via an integrating circuit) controls the current limiter in the Power ORing-circuit, power to the module is therefore removed, or applied without spikes on crate-busses during module exchange. Also because of the special bus driver/receivers used, which have high impedance against the busses when the power is removed, no interruption occurs in operation of the Data busses during module exchange.

BIT (Built In Test) are found in most CR80 modules. The test starts automatically when power is applied to the module and lights the red TEST LED on the front plate. When the internal test cycle, which lasts a few seconds, has been run through successfully, the TEST LED is estinguished to indicate this, otherwise it will remain on.

The red color is reserved for the BIT function, giving the CR80 computer its characteristic appearence when power is applied, the modules in the system will then light red and after a few seconds estinguish only leaving an eventually failed one still on, easily identifyable for exchange.

Other built in test functions, which are not destructive of the normal module function, are used for error detection by the CR80 on-line diagnostics, during actual operation of the computer.

2.2.2.2 Processor Unit Organisation

Installation of modules into the PU-Crate is shown overleaf (Fig. 2.2.2.2-1).

As previously described, interconnection of the PU modules is performed by means of two parallel transfer buses, the P-Bus and the C-Bus implemented as two backplane printed circuit boards. The buses have identical electrical and timing specifications with the following characteristics: transfer rate up to 4 megaword/second (16 bits + 2 parity bits), addressing of 1 megaword as word or byte. The P-Bus is the transfer bus for the Central Processor Units

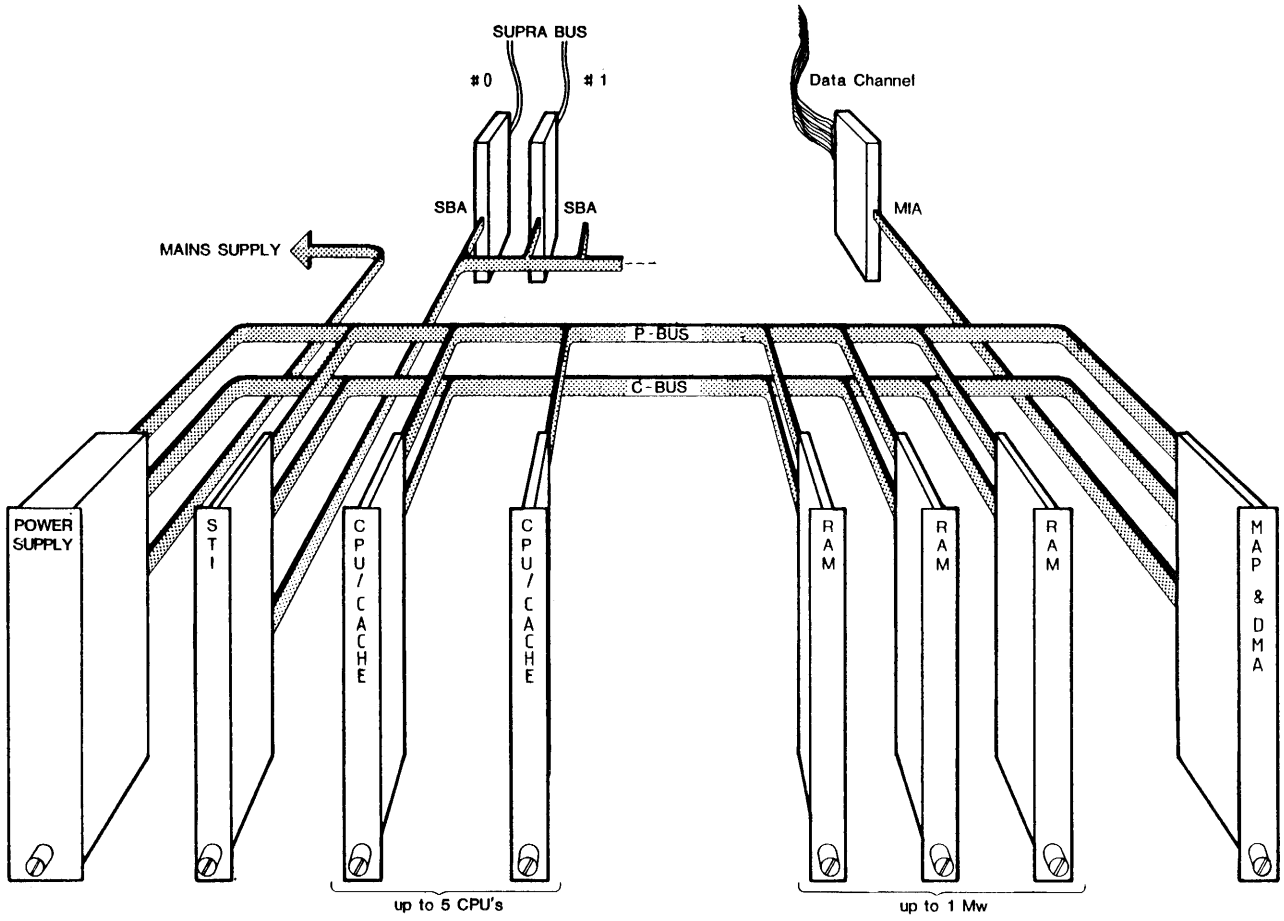


Fig. 2.2.2.2-1 Processor Unit (PU)

(CPUs), while the C-Bus is used as transfer bus by the C-Bus modules (DMAs).

The central processor units, CPUs, are general purpose processor units with a word length of 16 bits and the ability to address 64Kword of instruction and 64Kword of data. All data/instruction transfer performed by the CPU are via the P-Bus and the memory MAP to the memory. Referring to Fig. 2.2.2.2-2 overleaf, physically, the CPUs and the memory MAP are connected to the same P-Bus, but logically the CPUs recognize the MAP as being located between the memory and the CPU.

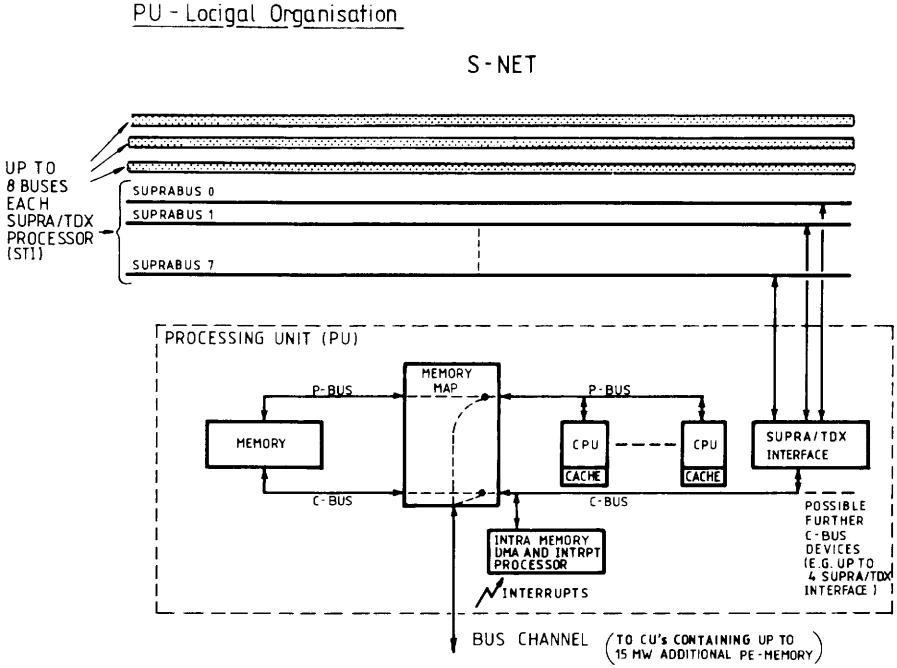
Each CPU includes a private 1Kword Cache Memory. This diminishes the load on the P-Bus, as CPU's often will find addressed memory locations in the Cache Memory.
The function performed by the memory MAP is to expand the addressable memory area to 16 megaword of which 1 megaword can be located in the PU as dual ported Memory, while the remaining 15 megaword can be located in CU's on the Data Channel Bus (accessed via the MAP Interface Adapter, MIA). Besides the address translation, the MAP also provides memory read/write protection, with protection performed individually for each 1K page of the memory.

The functions performed by the MAP on the P-Bus transfers are also performed on all C-Bus transfers. This means that the C-Bus Modules can access the complete 16- megaword memory area, subject to memory read/write protection.

Beside the address translation described above, the MAP module also includes the INTRA MEMORY DMA function, interrupt preprocessing and Data Channel Bus interface. The DMA is used for block transfer within the complete 16 megaword of Processing Element memory. The DMA is under control of the system software.

The interrupt preprocessing performed ensures that only interrupts with sufficient priority will cause a context switch in one of the CPUs, while all other interrupts will be queued by the MAP, until the status of a CPU allows service of them.

PU - Locigal Organisation



PU - Actual Organisation

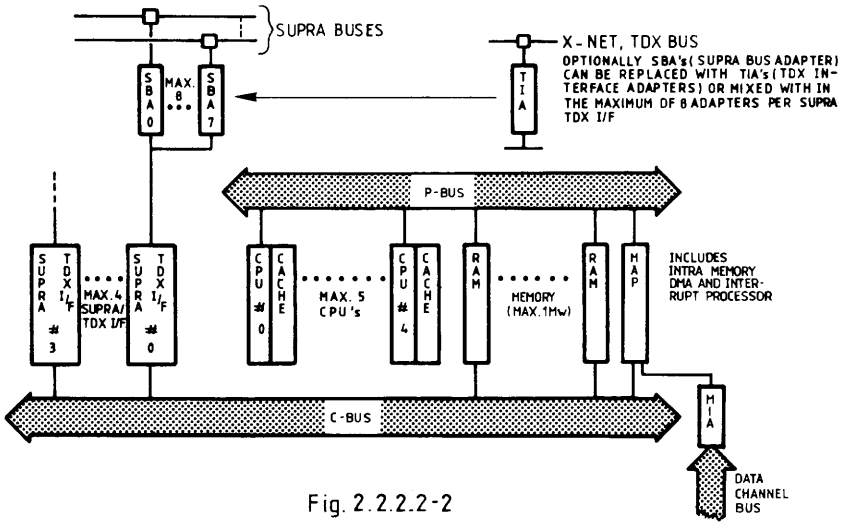


Fig. 2.2.2.2-2

Transfer on the Data Channel Bus will be performed by the memory MAP (via the MIA module) when the addressed location is not within the PU Memory addressing space (1 Mword).
The STI module contains DMA and associated processors for autonomeous multiplexed transfer of datablocks to/from the total 16 Mword PE-Memory. It interfaces via up to 8 SUPRABUS Adapter modules to the S-Net. Each SUPRABUS Adapter contains receive and transmit buffer and transformer interface to one 16 Mbit SUPRABUS. Up to four STI modules can be installed in a PU, allowing for interfacing a total of 32 SUPRABUSES (512 Megabit/sec.) to a Processing Element.

2.2.2.3 Channel Unit Organization

The organisation of Channel Units is shown below (Figure 2.2.2.3-1) and installation of modules into the CU-crate is shown overleaf (Figure 2.2.2.3-2).

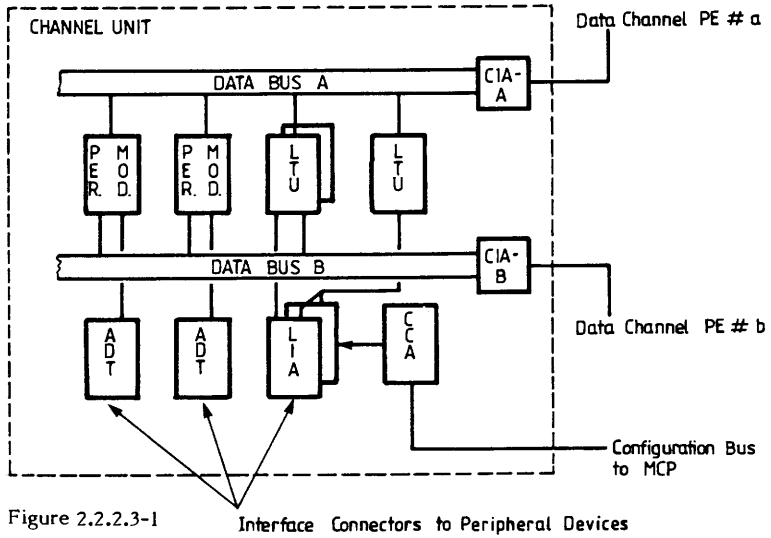


Figure 2.2.2.3-1

Interconnection of the CU modules is performed by means of two parallel transfer buses, Data Bus A and Data Bus B, implemented as two backplane printed circuit boards. The buses have identical electrical and timing specifications with the following characteristics: transfer rate up to 4

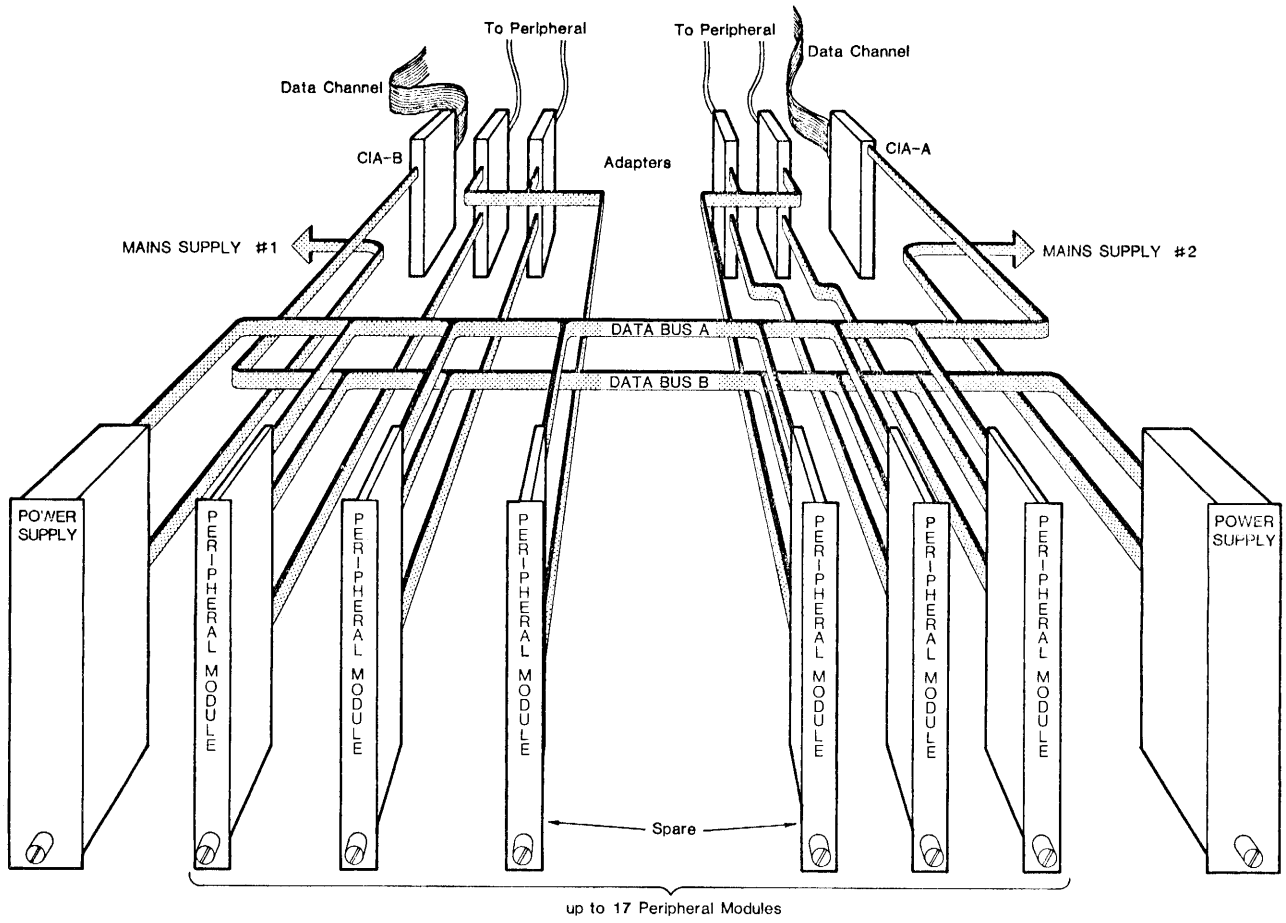


Fig. 2.2.2.3-2 Channel Unit (CU)

megawords/second (16 bits + 2 parity bits), addressing of 1 megaword as word or byte.
Data Bus A interfaces to the Data Channel of one Processing Element, and Data Bus B to the Data Channel of another Processing Element via Crate Interface Adapters (CIA-A and CIA-B respectively).
The Channel Units contains Memory Modules and Peripheral Modules. The dual transfer bus structure ensures that a single point failure will not stop operation of more than one Peripheral Module.

The physical interface to the peripherals, communication lines etc. is an Adapter module located at the rear of the CU Crate. For interfacing to communication lines, a special Adapter module (LIA-S) is available. This module is able to select a spare LTU module to be used instead of a failing LTU module. This selection is under control of the Maintenance and Configuration Processor MCP. The spare LTU can be back-up for a number of active LTU's (N+1 redundancy). As the internal bus structure is dualized, the power input is taken from two separate sources to ensure that a failure in one power source can not stop the CU operation.

2.2.3 The Data Channel

The Data Channel Bus is a 1.2 Megabyte/sec., byte serial, twisted wire bus. Through the combined address space and physical length it jointly extends the P-Bus and C-Bus in the Processor Unit (PU) to the attached daisy chained Channel Units (CUs). Only address destination modules (memory) can be attached on the Data Channel Bus. All access is from the PU through the memory MAP; The MAP automatically routes accesses to physical memory addresses above 1 Mword out on the Data Channel Bus. At the hardware level the Data Channel is viewed by the CPU's as a physical memory extension to the P-Bus in the PU. At the system level, However, the Data Channel is viewed by the CPUs as more intelligent due to the C-Bus DMA processors. The C-Bus Processors (INTRA MEMORY DMA/Interrupt processor and SUPRA/TDX processors) attached to the C-Bus in the PU, concurrently with the CPUs and under their control, can execute high level programs for moving data in the total memory of the CR80. Data can be moved within the Processing Element by the INTRA MEMORY DMA; and between Processing Elements via the S-Net. Data is moved as single data words, data blocks or as block multiplexed data streams, where up to several hundred simultaneous logical connections are handled autonomously by the C-Bus processors, requiring only high level interaction with the CPUs.

2.2.4 Peripheral System Architecture

Each Peripheral Device (Disc, tape, terminals, communication lines etc.) or a group of Peripheral Devices are attached to the CR80 by a Peripheral Module. A Peripheral Module, see fig. 2.2.4-1 overleaf, contains Compartmentalized Memory and a Peripheral Processor.

The Peripheral Processor (Microcomputer or bit-slice (CPU) handles the Peripheral Device interface, as well as lower level I/O processing (part of device handlers, communication protocols etc.) and store/fetches data in the Compartmentalized Memory via the Peripheral Module I/O bus. The Compartmentalized Memory (typically 16-64 Kilobyte) is part of the Memory space of one or two Processing Elements.

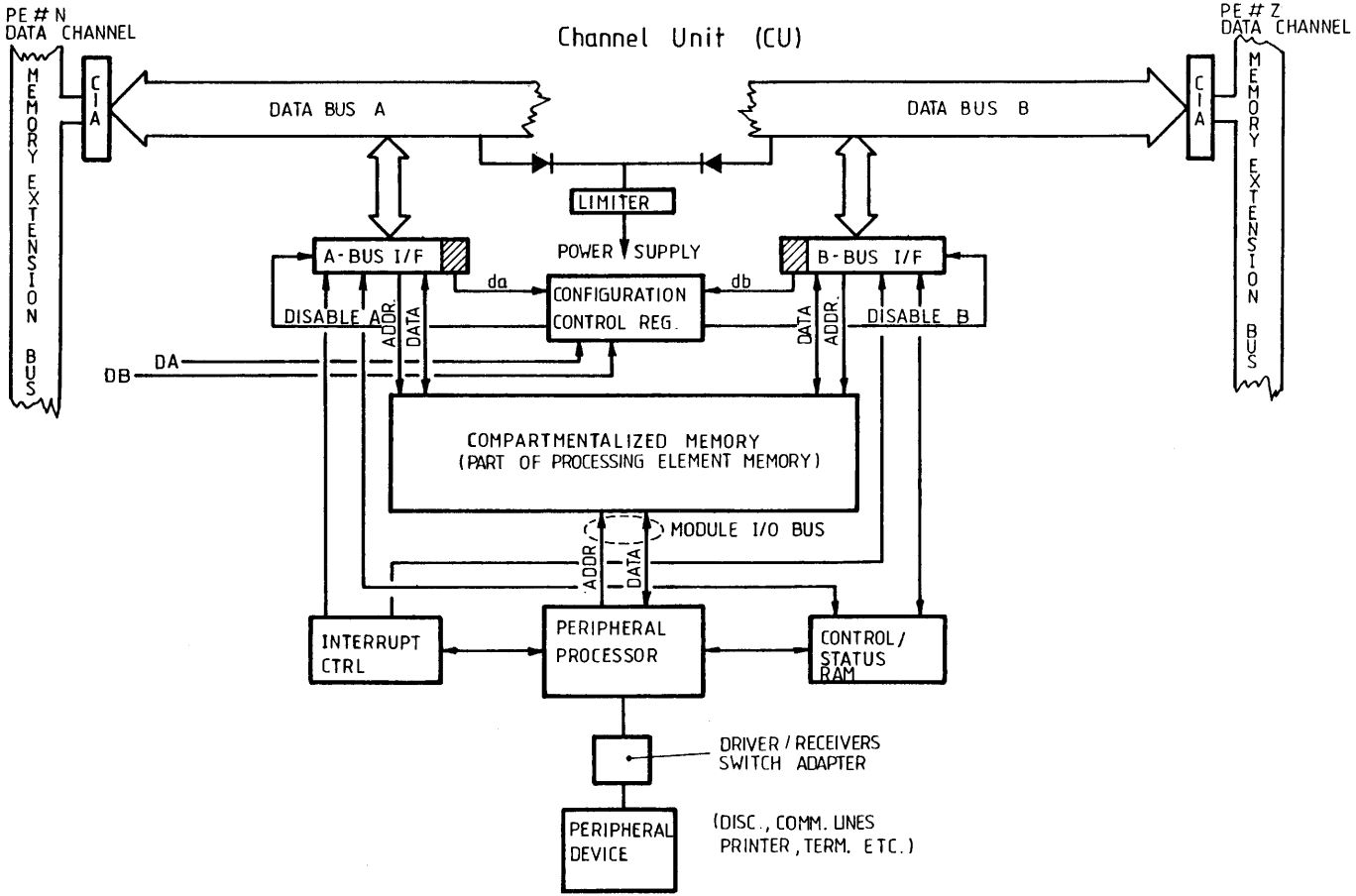


Fig. 2.24-1 Peripheral Module, Embedding Compartmentalized Memory and Peripheral Processor

The Processing Security is thus enhanced, in that Processing Element Memory is Compartmentalized in Peripheral Modules so that a memory fraction is handed over to only one associated Peripheral Processor. Distributing I/O processing to Peripheral Processors provides a separate level of multi-processing in the CR80 architecture, relieving the CPU's of real-time, device or character dependent processing , this together with each Peripheral Processor having its private I/O Bus to Processing Element Memory, thereby omitting Multiplexed I/O Channels, provides for the remarkable peripheral connectivity and throughput of the CR80. Also as Compartmentalized Memory is simply a part of Processing Element Memory, CPU's can directly access and process data without further movement.
The program executed by a Peripheral Processor, is either contained in PROM or in RAM, in the latter case this provides for loading the program from the Processing Element via the Compartmentalized Memory to the Peripheral Processor RAM. This allows standardization of Peripheral Module hardware by providing software adaption of a module to different peripheral devices. An example is the CR80 Peripheral Module for Communication Line Interfacing (Line Termination Unit, LTU), the standard LTU dependent on program loaded when initialized, copes with level 1 and 2 of most Asynchroneous, Synchroneous and Bit-synchroneous protocols (V24, BSC, HDLC, SDLC etc.) as well as variations in the line interface (control line options, different clock sources etc.). The soft-load feature is furthermore important, as discussed later, for N+1 redundancy between Peripheral Modules, in that the common spare Peripheral Processor can be loaded with adaptation software corresponding to that of the faulty Peripheral Processor it replaces.
Protection against un-availability of Peripheral Devices in case of failure of a Processing Element, is ensured by Peripheral Module Compartmentalized Memory being part of Memory Space of two Procesing Elements.

Referring to figure 2.2.4-1 showing peripheral module, Data Bus A, Data Bus B and Crate Interface Adapters, already described in previous sections. The Compartmentalized Memory is seen to be connected to the Data Bus A and the Data Bus B via bus interfaces A and B respectively. Further is shown a configuration control register, a current limiter a peripheral device, an interrupt controller, control/status register RAM and Power OR-ing diodes.

Main control lines and data control lines are shown and the signal directions indicated by arrows. Signals DA and DB is provided via separate external lines

from a not shown Maintenance and Configuration Processor (MCP), supervising the total CR80 computer. Attention should first be drawn to the configuration control register, this register being controlled in different ways to disable either of the Data Bus A and Data Bus B. If the Maintenance and Configuration Processor (MCP) detects a failure in the Processing element incorporating Data Bus A, it will issue a disable A signal DA to the configuration control register causing the register to disconnect the A-Bus Interface from the Data Bus A, apart from the hatchet fraction of the interface, the hatchet fraction and a similar fraction of the B-Bus Interface also being connected to inputs of the configuration control register via lines da and db respectively. Besides disabling the A-Bus interface the DA signal also oppresses the da signal.

If not being overruled by the signals DA and DB, the signals da and db respectively controls the configuration control register to enable or disable bus interface A or B. That is the bus interface may be controlled by the Processing Elements themselves. It is seen from Figure 2.2.4-1 that interrupt signals from the interrupt controller and data transfer to and from control/-status register RAM neither are transmitted via a disabled bus interface. Thus, it is possible that either of two Processing Elements are having access to the Compartmentalized Memory, controlled by the Processing elements themselves or by a supervisor system. The actual situation is reflected in the control/status register RAM, that is a Processing Element being knowledged about the operation of the other Processing Element, of actions of the Maintenance and Configuration Processor and further about the operation of the Peripheral Processor (e.g. the control/status RAM keeping the result of a self-checking routine in the Peripheral Processor). If the switch adapter has switched the Peripheral Device over from the peripheral processor to another Peripheral Processor the control/status register RAM of these Peripheral Processors will store the switching conditions so that the Processing Elements knows where to fetch/convey data.

To further enhance the reliability, the dual power supplies are connected as shown in Figure 2.2.4-1. Dual power supplies are often connected to dissipating devices via the power OR-ing diodes, but without the current limiter, whereby a short circuit in one module cuts the power to all modules supplied from the Power Supplies (draws down both power supplies through the diodes).

This is obviated by means of the current limiter standardly implemented in CR80 Peripheral Modules. The Current Limiter also gracefully turns power on/off the Peripheral Module as it is controlled by an integrating circuit connected to a special pin in the edge connector. This shortened pin is the last to connect, and the first to disconnect when a Peripheral module is inserted or withdrawn from a CU-crate. Also, special bus driver/receivers are high impedanced against the buses. Thus, if voltages in a module falls below nominal, the module can be serviced and exchanged in an operational CU-crate without introducing power supply spikes or noise on buses which might disturb the operation of other Peripheral Modules.

The N+1 redundancy of Peripheral Modules is now described with reference to Figure 2.2.4-2 overleaf, showing a peripheral subsystem of the CR80, typically incorporated in one CU-crate (printed circuit card cage) being interfaced to Processing Element Data Channels via Crate Interface Adapters (CIA). The CU-crate containing N+1 Peripheral Modules comprising peripheral processors and associated Compartmentalized Memory (RAM) and switch adaptors (LIA-S). The CU-crate is dual powered by Power Supplies A and B.

Also shown is a Crate Configuration Adapter (CCA), which is connected via the configuration bus to a Maintenance and Configuration Control Processor (not shown). The Maintenance and Configuration Processor supervising the overall CR80 system, receives status information from various modules and by way of example monitors the Power Supplies A and B via lines PA and PB respectively, the respective Crate Configuration Adapter and the configuration bus. The Maintenance and Configuration Processor also receives information from the Processing Elements, this information being by way of example a message concerning data missing or being incorrect from Peripheral Module no. N. Such a message is mostly created by way of the application software or on-line diagnostics. The Maintenance and Configuration Processor, will cause select spare signal to be transmitted to Switch Adapter no. N (LIA-S) via the configuration bus. The Switch Adapter comprises solid-state switches arranged in each buswire to disconnect the telecommunication lines from Peripheral Processor no. N and connect these lines to the common spare Peripheral Processor no. N+1 when the select spare signal is received.

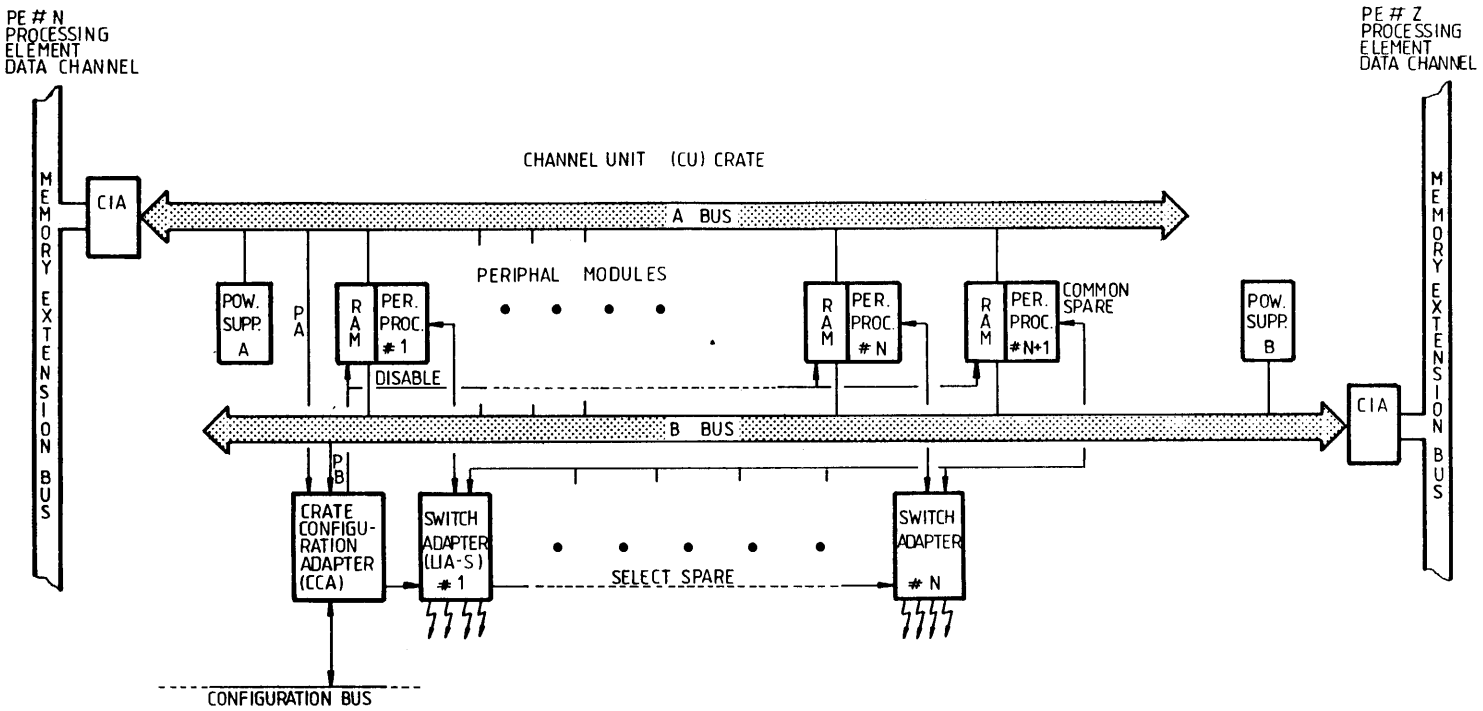


FIG. 2.2.4-2 REDUNDANCY OF PERIPHERAL MODULES AND CONFIGURATION CONTROL

The CR80 N+1 redundancy of Peripheral Modules combined with the dual incorporation of Compartmentalized Memory into two Processing Elements, yields great improvements over previously used techniques, as to connectivity and Fault Tolerance.

2.2.5 Maintenance and Configuration Processor (MCP) System

MCP system shown below (Figure 2.2.5) consists of standard modules specially suited for monitoring and control of CR80 Computers. The elements of the MCP system is described in detail in chapter 7 of this handbook, therefore only a brief overview is given here.

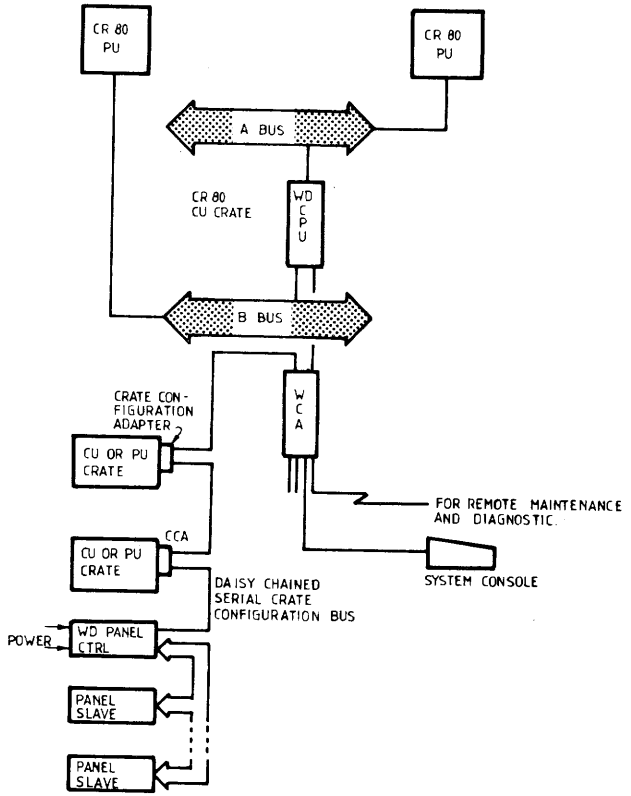


Fig 2.2.5
MCP System

The WD-CPU, positioned as a standard Peripheral Module in the CU-Crate, is the central Maintenance and Configuration Processor receiving status and control messages from the CR80 Processing Elements through its dual interface to two PE's of the CR80 system.

The WCA (Watchdog CPU Adapter) constitutes the interface between the WD CPU and the configuration Bus and the four available V24 communication ports. The V24 ports are used for connection of one or two system consoles and for connection to a communication port for remote maintenance and diagnostics of the CR80 system.

The Daisy Chained Configuration Bus is a dualized serial communication path between the WCA and the connected CCA's (Crate Configuration Adapters). The CCA is a standard CR80 adapter module designed for monitoring and control of the PU and CU Crates. The functions available are: monitoring of the DC voltages, switching of LIA-S modules (switching a spare LTU to the lines instead of a defect module), and monitoring of digital and analogue inputs, and control of digital outputs.

The WD CPU and the WD Panel Controller utilize alternative paths of the serial configuration bus for control and monitoring of the attached crates and associated modules. The serial configuration bus therefore is redundant with different parts of it being used in AUTO and MANUAL mode.

A fail safe circuit is implemented between the WD CPU and the WD Panel Controller, which performs automatic switching to the manual settings of the WD Panel in case of WD CPU failure or service. Similarly, replacement of the WD Panel Controller can be done without off-lining the system when under control of the WD CPU (AUTO MODE).

Crates under control of the MCP system is galvanically isolated by optocouplers from the serial configuration bus and can be removed from the operational configuration bus without electrical interference with the remaining part of the system.

2.3 Accessing and Moving Data in the CR80

The CR80 access and tranfer techniques are described in this section. In particular security provisions through state control, mapped data transfers and privileged read/write instructions are covered.

2.3.1 Security

The inherent logical and physical separation of programs and data in the CR80 architecture is well-suited for preventing unauthorized access to data and programs and for preventing non-intended modification of programs.

The objectives of the protection mechanisms in the CR80 are:

- to protect data belonging to a process against unauthorized modification by other processes and against not intended reading;
- to protect programs against modifications;
- to prevent unauthorized execution of programs;
- to prevent processes from monopolizing the system resources.

Security is supported by means of memory access protection and division of instructions into three privilege classes.

The CPU has 16 states of which one (state 0) is USER STATE and 15 (states 1 through 15) are SYSTEM STATES. Higher states have more privileges than lower states. In USER STATE only non-privileged instructions may be executed. Medium privileged instructions can be executed at all SYSTEM STATES while the most privileged instructions are reserved for execution at SYSTEM STATE 15 (system high).

Attempt to illegally execute a privileged instruction in USER STATE or SYSTEM STATES 1 through 14 causes a local interrupt, upon which the CPU automatically invokes a supervisor routine.

The CPU state is changed by means of the MON_instruction which is used to activate system procedures.

In addition to the memory protection provided in USER STATE by the MEMORY MAP, each of the system states has its own memory bound register. Only data memory locations below or equal to this boundary value may be modified while all data memory locations available might be read in SYSTEM STATE.

The Memory Map protection mechanism which is active in User State is implemented by means of two access control bits for each 1 K word page in memory. The protection values are:

00 Page absent
01 Full access
10 Read only
11 No access

As will be seen in the following all non-privileged (USER STATE) memory accesses (both from CPU's and DMA's) go through the Memory Map, and are checked by hardware not to violate the protection value. In the System States full access (read or write) is granted irrespective of the protection value.

If a not allowed access is attempted, the transfer is terminated without sending the physical address to the memory, and a transfer error is signalled from the Memory Map.

The "Page absent" condition is used to invoke the demand paging feature of DAMOS. It indicates that the accessed page is not resident in main memory (or not mapped in), and will lead to suspension of the process until the page has been loaded into memory or relocated.

2.3.2 Mapped Data Transfers

The CR80 adds to its unique processing strength by the CPU's being offloaded from doing bulk data transfers and low level I/O processing. This is performed by distributed, intelligent DMA (Direct Memory Access) processors and peripheral modules (Peripheral Processors).

Security is ensured by all accesses having to go through the Memory Map.

The DMA processors are all attached to the C-bus in order to leave the P-bus completely available for CPU activity. The INTRA MEMORY DMA (attached to C-bus, but physically part of the MAP module) performs block multiplexed DMA transfers within the total Processing Element (PE) Memory (within PE-memory in PU, within PE-Memory in CU's (Memory and Peripheral Modules) or between PE-Memory in PU and CU's).

The SUPRA/TDX I/F DMA performs block multiplexed DMA transfers between the S-NET (or X-NET) receive/transmit buffers and the PE-Memory (PE-Memory in PU or PE-Memory in CU's (Memory and Peripheral Modules)). The transport over S-NET (or X-NET) between PE's of the receive/transmit buffer content is performed by other means described elsewhere in this handbook.

The C-Bus DMA's (INTRA MEMORY DMA or SUPRA/TDX I/F) are multiplexed by their associated DMA processors, having control blocks for outstanding DMA transfers (up to several hundreds) in their private control memory.

All Memory accesses, whether under control of the CPU's or DMA's, are performed through the Memory Map (as transfers in logical memory space), thereby ensuring that the memory protection features, previously described, are all in operation.

Access to the Data channel is interleaved between CPU's and DMA's on a word by word basis. Figure 2.3.2-1 overleaf shows the source and destination of the various types of datatransfer within the CR80 architecture, as described below:

A: CPU	addresses PE-Memory in the PU
B: CPU	addresses PE-Memory on the DATA-CHANNEL
C: INTRA MEMORY DMA	addresses PE-Memory in the PU as Data source and Data destination.
D: INTRA MEMORY DMA	addresses PE-Memory on the DATA-CHANNEL as Data source and Data destination.
E: INTRA MEMORY DMA	addresses PE-Memory in the PU as data source (or data destination), and PE-Memory on the DATA CHANNEL as data destination (or data source).
F: SUPRA/TDX I/F	addresses PE-Memory in the PU as data source (or data destination), and SUPRA BUS ADAPTER transmit buffer as data destination (or receive buffer as data source).
G: SUPRA/TDX I/F	addresses PE-Memory on the DATA CHANNEL as data source (or data destination), and SUPRA BUS ADAPTER transmit buffer as data destination (or receive buffer as data source).

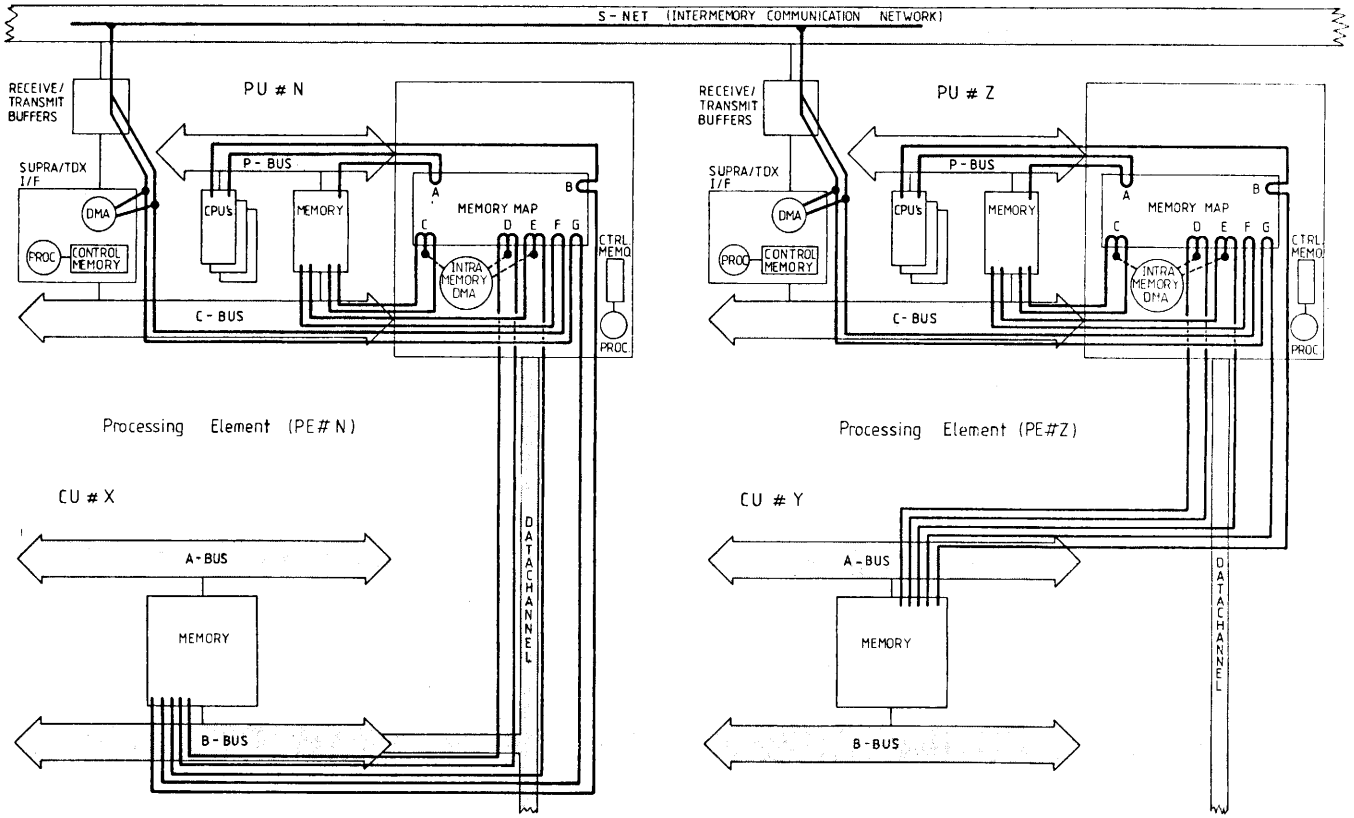


Fig. 2.3.2.-1
CR 80 Mapped Datatransfers

Some interesting and quite unique features of the CR80 architecture can be seen from above:

- Secure PE to PE transfers via S-NET (G-G, F-F or F-G), as these go through the Memory Map of both the sourcing PE and destination PE. As these Memory Maps are solely controlled by the respective PE's, it is not possible for the destination PE to gain access to data in the source PE not intended for it, as well as the source PE can not write data in the destination PE outside the area specified by the destination PE.

- Data transfer directly between Memory of one PE to/from Peripheral Module belonging on datachannel of another PE via S-NET (F-G). After initial set-up of the Peripheral Module by the owning PE (initiated by PE to PE messages), data (e.g. to/from a disc controller) can be transferred via the S-BUS to the requesting PE, without interrupting operation of the owning PE or intermediate buffering.

- Data transfer directly between Peripheral Modules belonging on data channels of different PE's via S-NET (G-G). After initial set up of the Peripheral Module, by their respective owning PE's (initiated by PE to PE messages), data (e.g. between communication interfaces (LTU's)) can be exchanged via the S-BUS, without interrupting operation of the owning PE's or intermediate buffering. This is a very useful feature in large communications or packet switches.

The above features are supported by the CR80 approach to Peripheral Modules, as being autonomous peripheral processors, distributedly performing all low level I/O processing and communicating directly into the memory of Processing Elements.

2.3.3 Privileged read/writes

Privileged (non-mapped) read/writes are used by the CPU's (only allowed when in SYSTEM STATE) for loading the Mapping memory (Translation Tables) and Translation Table Registers (TTR), as well as for communicating with control memory of C-BUS DMA Processors and Peripheral Modules.

C-BUS DMA processors (SUPRA/TDX I/F DMA and INTRA MEMORY DMA) utilize privileged (non-mapped) read/write for loading their dedicated Translation Table pointers (TTR pair) and associated part of the mapping memory, with a memory view (Translation Tables), as specified in the control block for a particular DMA transfer. Thus the C-BUS DMA processors, by being able to control their subset of the memory map, can intelligently multiplex the use of their DMA, between up to several hundred control blocks (or pointers to) contained in their control memory, each specifying a different transfer request (DMA) and associated memory view.

System security is ensured by the fact that only CPU's, and only in SYSTEM state, can load DMA control blocks and memory views into the control memory of a DMA processor.

The drawing overleaf (figure 2.3.3-1) shows the source and destination of the various privilged read/writes as described below:

A: CPU privileged read/write in MAP translation RAM (address translation tables) or TTR RAM.

B: CPU privileged read/write in MAP Processor control memory.

C: CPU privileged read/write in Direct Memory Access Processor (STI) control memory.

D: CPU privileged read/write in control memory of a Peripheral Module on the DATA CHANNEL.

E: MAP Processor writes in MAP translation RAM or TTR RAM for setting up logical to physical address translation for an INTRA MEMORY DMA transfer.

F: Direct Memory Access Controller (STI) processor writes in MAP translation RAM or TTR RAM for setting up a logical to physical address translation for a DMA transfer.

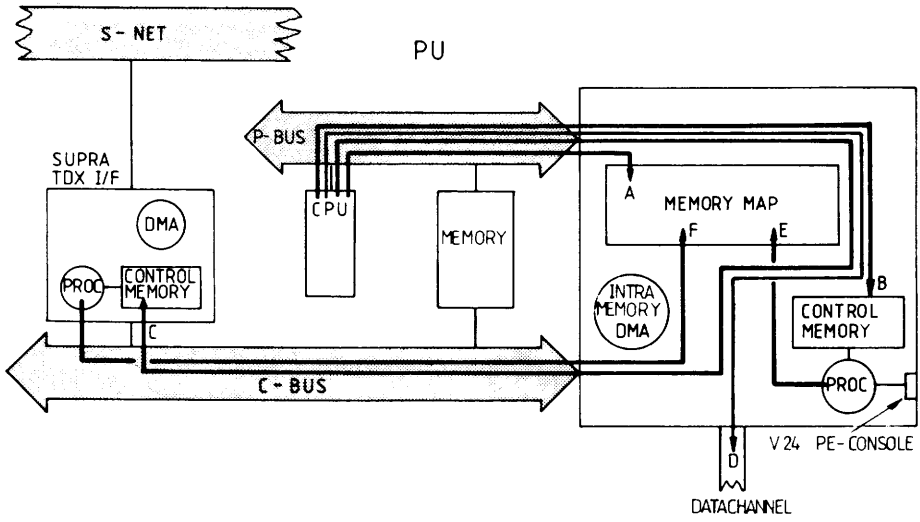


Fig. 2.3.3-1 Privileged Read / Writes
(CPU in SYSTEM STATES only , or C Bus DMA processor privileged Memory Map Control)

2.4 Central Processors (CPU/CACHE)

2.4.1 Introduction

The CR80 CACHE/CPU modules used with Memory Mapped CR80 Computers are divided into the CPU and the CACHE memory parts as shown overleaf:

The CPU part of the CPU/CACHE module is a general purpose processing unit with a 16-bit word length and capable of addressing 2 x 64K words of memory. The CPU has sufficient microprogram space to support an optional alternative instruction set. The standard instruction repertoire has more than 220 basic arithmetic, logic, transfer and special instructions including bit, byte, word and multiple word manipulations.

The standard instruction set is implemented in approximately 2K words of a 4K x 64 bits Microprogram PROM.

An alternative instruction set may contain:

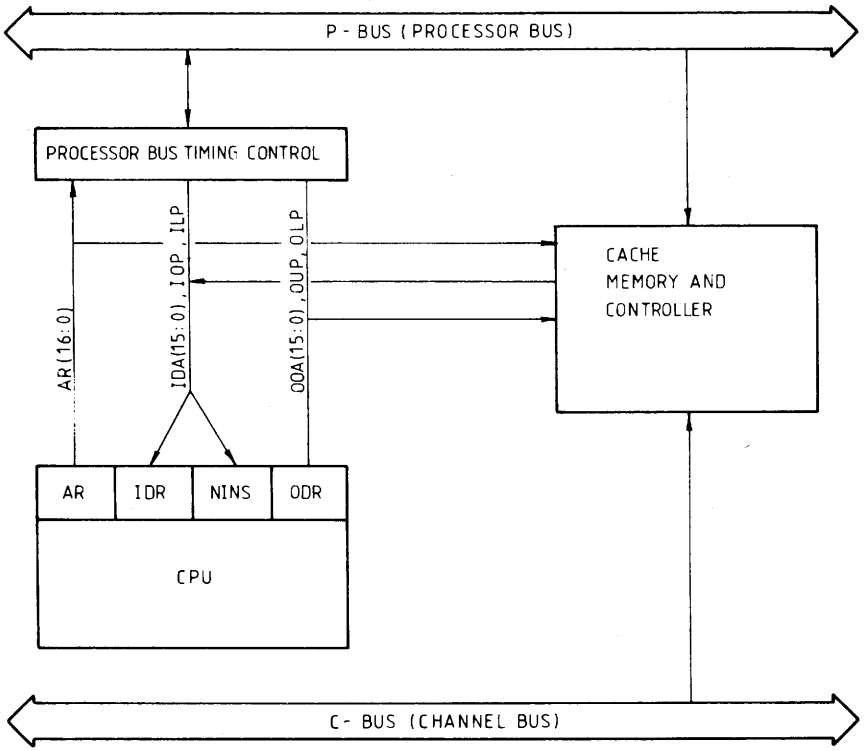
- Decimal Arithmetic instructions
- Instructions supporting high level languages
- System routines
- User-defined instructions

Internal context registers include 8 general purpose accumulator or index registers (R0-R7), a base register (BASE), a program base register (PROG), a program counter (PRPC), a timer register (TIMER), a process status word (PSW) and a modify register (MODIFY), a execution register (EXR), a bound register (BOUND) and a Cache Error Register (CER).

Instructions are provided to save and load all registers minimizing overhead during context switching.

Instructions can be addressed directly or indexed, relative to PROG, indexed relative to PC. Data is always addressed relatively to the base register BASE.

Instructions are addressed in 16-bit words. Data may be addressed in multiples of bits, bytes or words.



AR: ADDRESS REGISTER
IDR: INPUT DATA REGISTER
NINS: NEXT INSTRUCTION REGISTER
ODR: OUTPUT DATA REGISTER

Fig. 2.4.1-1
CPU/CACHE Functional Block Diagram

Two ALUs (Arithmetic Logic Units) are included, allowing address and data manipulations to be performed simultaneously, or allowing address calculations involving three elements to be performed in one cycle when employing base register addressing.

The micro-instruction cycle time is 250 ns.

The CPU may execute in two states:

- USER state, and
- SYSTEM state

In the system state the CPU will further be at one of 15 system levels which are numbered 1 through 15. In user state, the level is 0.

In USER state the CPU is not allowed to execute any privileged instructions and it is subject to the memory access protection mechanism of the MAP module.

In SYSTEM state the CPU is not subject to the memory access protection mechanism of the MAP-module.

For those of the instructions which may be executed on levels 1 to 14, a memory write protection mechanism based on a BOUND register applies: The CPU only allows write operations to locations with effective logical addresses lower than or equal to the current value of the BOUND register. This mechanism is implemented in the CPU alone (not involving the MAP module).

Instructions which may only be executed on level 15 (highest level) are subjected neither to the MAP module nor to the BOUND register memory protection mechanisms.

In the CR80 system, four types of vectored interrupts exist:

- Module Interrupts
- CPU Interrupts
- Power Failure Interrupts
- Real-Time Interrupts

And two types unvectored interrupts:

- Fast Timer Interrupt
- Termination (Error) Interrupts

The interrupts are resolved by hardware within the Memory MAP module.

Two Mask bits, one for the vectored interrupt types and one for the Fast timer interrupt are available in the CPU process status word (PSW).

Vectored Interrupts are grouped on 16 priority levels. Only interrupts with priority higher than the priority of the process presently executing on the CPU are serviced.

Fast Timer Interrupts (each 250 us) are serviced by CPU micro-program. When a timer interrupt is received, the CPU TIMER register is decremented and tested. If the TIMER register is negative and the PSW mask bit for timer interrupts is not set, a local interrupt is generated. This feature is used with multiprogramming of the CPU. A process is loaded with a timeslice inserted into the TIMER register; when the timeslice runs out (local interrupt) the next scheduled process to execute is loaded (with a new time slice).

The CACHE Memory part of the CPU/CACHE consists of a 1K x 38-bit static memory directly connected and accessible from the CPU without using the P-bus.

Fundamentally, the CACHE memory provides a small (compared to main memory) high speed buffer containing copies of most recently accessed memory locations in the PU part of Processing Element memory.

The CACHE memory algorithm used is the "write through algorithm", which stores all of a CPU's "reads" from Main Memory via the P-Bus in it's CACHE memory and refers all the CPU's "writes" directly to Main Memory (via the P-Bus) with its CACHE memory updating itself in parallel if it contains the specified address.

The CACHE memory also monitors the "write" accesses done by other CPU's on the P-Bus and by DMA's on the C-Bus, and in case it contains the specified address, it is erased in the CACHE memory.

The above together with the "write through algorithm" ensures that the contents of addresses stored in PE-Memory and the CACHE memories of CPU's are always identical. When context switching on the CPU takes place, normally associated with change of CPU logical memory view, the complete CACHE memory content is erased.

The efficiency of the CACHE memory relies on the basic principle that, in the step by step execution of a program, a CPU-originated read operation at an address has a high probability of being repeated, and therefore can be found in the CACHE memory after the first access to it.

Two advantages when using CACHE memories are obtained:

- Loading of the P-bus (Processor Bus) is reduced, allowing for the use of more CPUs before bus contention occurs. The max number of CPU/-CACHEs to be connected without risk of continous bus contention has been raised to 5, compared to 2 if CPUs without CACHE memory had been used.
- Average access time when performing memory read operations is reduced, as reads from CACHE is faster than from Main Memory, thus improving the speed of each CPU.

Generally, the presence of the CACHE memory is only visible to the programmer in terms of improved execution speed, as no CACHE handling software is required.

2.4.2 PE Multiprocessing Performance Enhancement, Utilizing CPU/CACHE

The calculated benefits from using the CPUs with CACHE memory in the PE are as follows and shown (figure 2.4.2-1) overleaf:

Based on analysis and measurements on the distribution of 12147 instructions of DAMOS Kernel programs it was found that the average instruction has 1.98 memory references.

With one CPU/CACHE in a PU, and assuming a negligible frequency of memory-write-operations initiated by other address sourcing modules than the CPU's, the overall CPU efficiency (80% HIT RATE) will improve 40%, due to faster memory access, relative to a CPU without CACHE memory. The CPU/CACHE, on average (80% HIT RATE), uses 28% of the P-Bus bandwidth.

Thus, some improvement of the throughput will result with a single CPU/-CACHE in a PU, but the main advantage is that a greater number of CPUs can be used in a single PU without heavy P-Bus contention.

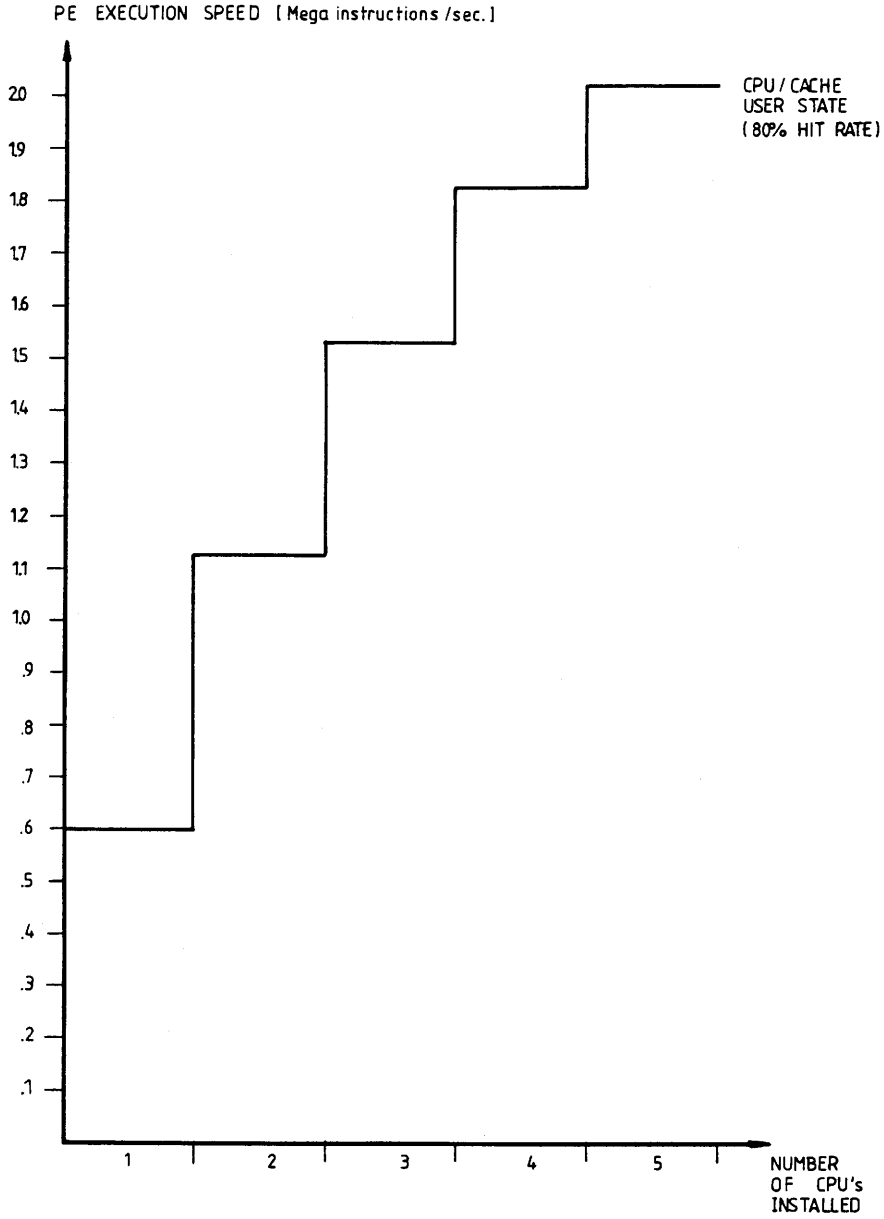
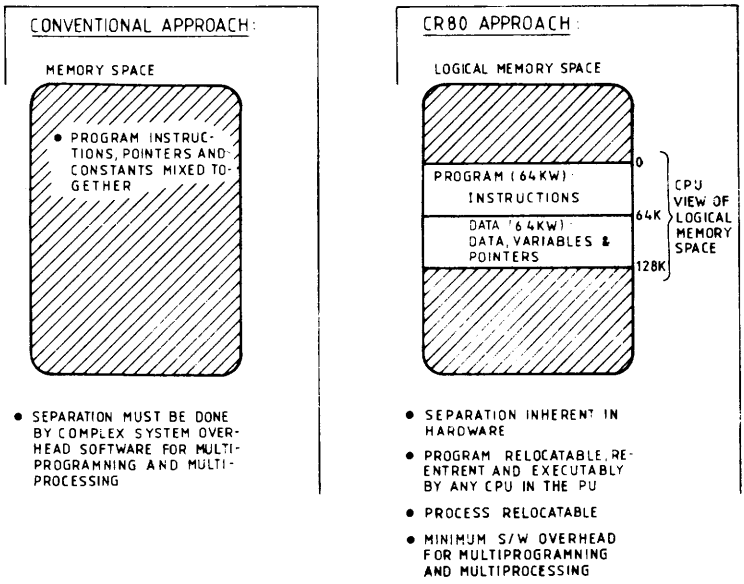


Fig. 2.4.2-1
PE Multiprocessing performance using CPU / CACHE 's
(CR 8003, 80% HIT RATE 0,6 MIPS)

2.4.3 PROGRAM and DATA

The key to the CR80's success in safe reliable multiprogramming and multiprocessing is the separation in both hardware and software of the computer operations into a program part and a data part of each process.

The program part is the static list of machine instructions to be executed, and the data part, which is separated totally from the program part in the memory, is the variable part of a process.



The program is addressed relatively to the program register. The standard instruction set includes a number of instructions which operate on Program data. However, all these instructions can only read program locations, not write into or modify the contents of program locations.

The Data part, besides containing all variables used by the program at execution time, also has a context for complete images of the CPU registers including the location counter. This area is called the Process Parameter Block. It is contained in Data page 63.

When a process is preempted, the CPU registers are pushed on the Context Stack in the Process Parameter Block by hardware. A process can thus be preempted and resumed at any point in execution time. This allows for easy timesharing of CPU time between processes. Each process contains a complete state description of the execution of a program. The only information necessary in order to resume a particular process is the physical address of its Process Parameter Block. Further the separation into separate process and program parts allows different processes to share the same single copy of a program. The processes may independently of each other execute different parts of the same program. If the system contains more than one CPU, processes may even execute the same program in true multiprocessing mode.

2.4.4 CPU Registers

The CR80 CPU shown overleaf (Figure 2.4.4-1) contains 8 general purpose registers and 8 special registers.

General: R0, R1, R2, R3, R4, R5, R6, R7

Special:		
	PSW	Process Status Word
	EXR	Execution Register
	BASE	Base for Data Addresses
	MOD	Modify Register
	PROG	Program Base Register
	PRPC	Program Counter
	TIMER	Timer Register
	BOUND	Bound Register
	CER	Cache Error Register

General Purpose Registers

The 8 general purpose registers may be used as

- Accumulators
- Intermediate Storage
- Index Registers

Process Status Word (PSW)

The PSW register contains masks and flags concerning interrupts, state of the arithmetic/logic unit, and the CPU state. The layout of the PSW is shown in Table 2.4.4.2a where the meaning of the flags is explained.

Fig.2.4.4.-1
CPU/CACHE
Accessible Registers ALU & Shift Control when using Standard Instruction Set

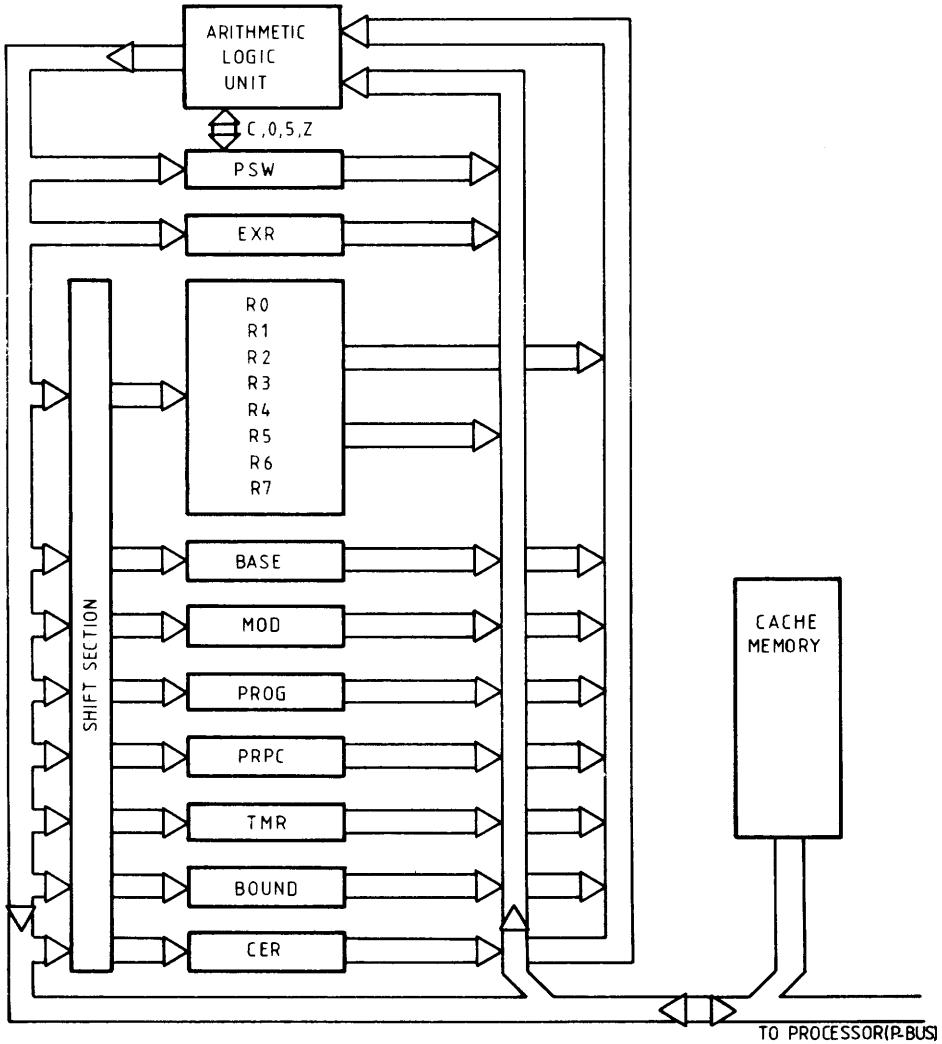


Table 2.4.4.2.a CR80 CPU/CACHE PROCESS STATUS WORD (PSW):

BIT NMB	NAME	SET BY	DESCRIPTION
15	timer mask	SW	interrupts caused by a timer notification are disabled if set
14	notific. mask	SW	vectored interrupts are disabled if set
13			not used
NOT USED 12	alternative instruction set	SW	instructions are decoded according to an alternative instruction set if the flag is set
11	CPU state	SW	set means SYSTEM (supervisor) state, reset means USER (application process) state
10	CPU number	HW	
9	CPU number	HW	physical CPU number
8	CPU number	HW	
7	Z	HW	an all zero result of an arithmetic op.
6	S	HW	the sign bit from an arithemtic op.
5	V	HW	a not overflowing arithmetic op.
4	C	HW	the carry from an arithmetic op.
3			not used
2			not used
1	trace	SW	set indicates trace mode
0	modify	SW	set when an instruction modification condition exists

- PSW, Interrupt Masks

  The bits 15 and 14 control the ability of the fast timer, other CPU's and I/O devices to interrupt the executing process. A detailed description is given in the section on INTERRUPTS.

- PSW, Alternative Instruction Set

  Bit 12 indicates whether the CPU is interpreting instruction codes according to the standard instruction set or according to the optional alternative instruction set (Standard Instruction Set, when zero).

- PSW, CPU State

  Bit 11 defines the current CPU state - USER or SYSTEM. If bit 11 is set the CPU is in SYSTEM state otherwise USER state.

- PSW, CPU Number

  The bits 10-8 contain the physical device No. of the CPU on the P-Bus. These bits are controlled by three switches on the CPU circuit board.

- PSW, Arithmetic Flags

  The bits 7-4 contain status flags from the arithmetic/logic unit. The flags are updated by arithmetic instructions only. After an integer arithmetic operation the meaning of the flags is:

  The Z-flag is set if the result of the operation was zero and cleared if the result is non-zero

  the S-flag is the sign-bit of an arithmetic operation.

The V-flag is set if an overflow condition did not occur.

The C-flag is set if the arithmetic operation produced a carry and cleared if not.

- PSW, Trace Flag

If the trace flag is set, the CPU will generate a trace interrupt after execution of each instruction in user state.

- PSW, Modify Flag

Bit 0 indicates whether a modify is active (active when set).

- Execution Register (EXR)

The EXR contains four fields: PRIO, LEVEL, PREV_LEVEL and VIEW.

- PRIO

This field is contained in bits 8, 9, 10 and 11 of the EXR. PRIO contains the current execution priority of the CPU, and it is a copy of the priority used by the MAP for the CPU when deciding whether to interrupt the CPU or not. PRIO is affected by interrupts and by the CALL and RETURN, and STP instructions.

- LEVEL

This field is contained in bits 4, 5, 6 and 7 of the EXR. When the CPU is in SYSTEM state, LEVEL defines the current system level of the CPU. LEVEL is affected by interrupts and by the MON, RTM, CALL and RETURN instructions. With the CPU in USER state, LEVEL is 0.

- PREV LEVEL

This field is contained in bits 12, 13, 14 and 15 of the EXR. It contains the level from which the current level was called.

- VIEW

This field is contained in bits 0, 1, 2 and 3 of the EXR. The VIEW defines the current view (cf. section 2.4.5.4) used by the CPU. VIEW is affected by interrupts and by the CALL and RETURN instructions.

- MODIFY REGISTER (MOD)

  The effective address of most CR80 instructions can be modified when preceded by a MODify instruction.

  The MODify instructions add or subtract some value to or from the MOD register.

  Several types of MODifications are possible:

  - Base relative memory address modification
  - Program relative memory address modification
  - Location (PRPC) relative memory address modification
  - Module address modification
  - Parameter modification such as immediate value, shift count etc.

- BASE REGISTER

  The BASE register provides the baseline for all addressing of data locations.

- PROGRAM REGISTER (PROG)

  A program can be placed anywhere in the logical program address space (64Kw) of the CPU. The PROG register points to the start of the program being executed by the CPU. All program relative addresses are relative to the PROG register.

- LOCATION COUNTER (PRPC)

  The PRPC register contains the absolute logical address of the next instruction to be executed by the CPU.

- TIMER REGISTER

  This register is used for tying a process to TIME. The register can be preset by the Load Timer instruction (LDT). The TIMER register is decremented by one every time the fast timer notification occurs. The frequency of the fast timer is determined by the MAP module to 250us.

  If after decrement of the TIMER register, its value is negative and if the TIMER mask (PSW bit 15) is reset, an interrupt is generated.

- BOUND REGISTER

  The BOUND register provides basis for the memory write protection mechanism applied in SYSTEM state. Memory write addresses are checked to fulfil the restriction:

  0 <= effective write address <= BOUND

  where the BOUND register is interpreted as an unsigned integer in the range 0-65535.

  The BOUND register is a function of the current state (level 0 -15) BOUND is affected by the MON, RTM, and RETURN instructions.

  Instructions, which may only be executed on LEVEL 15 are not subject to the BOUND - Protection mechanism.

- CACHE ERROR REGISTER (CER)

The CR80 CPU/CACHE is equipped with a cache memory providing a significant performance enhancement.

The "health" of the cache memory is constantly monitored by the CPU, incrementing the CACHE ERROR REGISTER upon cache memory parity errors. The contents of the CER may be transferred to a general purpose register.

The most significant bit of the CER is a software setable bit enabling/-disabling the cache memory (CACHE enabled when set).

2.4.5 CPU Handling of Interrupts

2.4.5.1 General

- An interrupt is defined as:

  1) an event which is, either, unrelated to the current program being executed, and beyond the control of that program, or which is related to a fault condition caused by the program.

  2) which preempts the current process, and

  3) transfers control to the operating system kernel.

- Interrupts can be caused by:

  A) notifications by Peripheral Modules, C-Bus DMAs and CPU's, in common called vectored interrupts.

  B) notifications by the fast timer, which cause the timer register to become negative (timer interrupts).

  C) a set trace flag (trace interrupt).

  D) a page fault (absent page) interrupt

  E) a number of error conditions (error interrupts):

    - attempted execution of an illegal instruction (unassigned instruction code)

    - execution of a TRAP instruction.

    - Parity error in memory

    - timeout from memory or from a device

    - attempted violation of memory protection in USER state

- attempted violation of BOUND memory protection in SYSTEM state

- attempted execution of a privileged instruction in USER state (privilege violation, USER state)

- attempted execution of a privileged instruction in SYSTEM state (privilege violation, SYSTEM state)

- context stack overflow

- context stack underflow

- error encountered during interrupt processing or error encountered during context stack operation, destroying context stack (emergency interrupt).

Interrupts stemming from the first two causes (A and B) may be individually masked by the interrupt masks (bits 14 and 15) in the PSW. Interrupts are only generated by the third cause (C) if the CPU is in USER state. The interrupts originating from the last causes (D and E) cannot be masked.

2.4.5.2 Processing of Interrupts

The processing of interrupts by the CPU includes the following steps:

1) Except for trace interrupts, a view is fetched from locations defined below in program page 0. The corresponding physical program and data Translation Table Registers (TTR) are determinated.

Interrupt cause		View from location
CPU interrupts	(A)	32
Error interrupts	(E)	33
Page interrupts	(D)	34
Timer interrupts	(B)	35
Module interrupts	(A)	36

The program and data Translation Table Registers are loaded by those values defined by the view. The new logical data page 63 (the Process Parameter Page) is changed. If the interrupt is a Module Interrupt (i.e. interrupt vector greater than 31), the page address is taken from location no.FFEO in the former data space (cf section 2.4.5.4 Process Parameter Block), otherwise the page address of the former logical data page 63 is used (fetched from the former Translation Table). For trace interrupts, the program and data Translation Table Registers are not changed.

2) The current context is pushed onto the context stack except in the case where the cause for the interrupt is a context stack overflow or underflow.

The value of PRPC pushed onto the context stack upon an interrupt depends on the type of interrupt:

- at interrupts of types A, B and C, the stacked value of PRPC will point at the instruction succeeding the last instruction processed.

- at all other interrupts involving context stacking, the stacked value of PRPC will point at the instruction under execution, when the interrupt was detected. It should be noted that all instructions are implemented allowing - to some extent - reexecution, if the instruction is interrupted by an error.

- no instruction will change the MODIFY FLAG (PSW bit 0) or the MODIFY register before it has been determined that no errors can arise.

- no instruction will commence memory write operations before it has been determined, that all memory locations to be accessed by the instruction are accessable.

3) A primary and secondary cause code are generated as shown in table 2.4.5.2a overleaf (ALA is an abbreviation for Absolute Logical Address). A tertiary cause code is generated at interrupts with primary cause codes equal to 3, 4 and 8 through 18. The tertiary cause code is the contents of the MAP status register.

Interrupts with primary cause codes greater than or equal to 7 are error interrupts.

The primary and secondary cause codes are loaded into registers 7 and 0 respectively. The tertiary cause code is loaded into register 1 at interrupts with primary cause codes equal to 3, 4 and 8 through 18.

The interrupt masks are set and the CPU state is set to system state, and the system level is set to the highest level.

The PROG and BASE register is set to zero; BOUND is set to 65535 (no. FFFF) and MODIFY is clerared.

Table 2.4.5.2.a Interrupt Cause Codes

Interrupt Cause	Primary	Secondary Cause
unassigned	0	
CPU interrupt	1	interrupt vector
Module interrupt	2	interrupt vector
Page fault in program space	3	ALA of absent location
Page fault in data space	4	ALA of absent location
Timer interrupt	5	Value of TIMER register
Trace interrupt	6	ALA of next instruction

Table 2.4.5.2.a Interrupt Cause Codes (continued)

Interrupt Cause	Primary	Secondary Cause
Illegal instruction	7	ALA of illegal instruction
Parity error in program space	8	ALA of location with parity error
Parity error in data space	9	ALA of location with parity error
Parity error on privileged read/ writes	10	Device Address
Parity error on MAP access	11	MAP Address
Parity Error in MAP Translation table	12	ALA of addressed location

Table 2.4.5.2.a Interrupt Cause Codes (continued)

Interrupt Cause	Primary	Secondary Cause
Timeout from program memory	13	ALA of not responding location
Timeout from data memory	14	ALA of not responding location
Timeout from Privileged read/ writes	15	Device address
Timeout from MAP	16	MAP address
Protection Violation in program space	17	ALA of addressed location
Protection Violation in data space	18	ALA of addressed location
Bound Violation	19	ALA of addressed location

Table 2.4.5.2.a Interrupt Cause Codes (continued)

Interrupt Cause	Primary	Secondary Cause
Privilege violation 'p' instruction	20	ALA of privileged instruction
Privilege violation 'pp' instruction	21	ALA of privileged instruction
Context stack overflow	22	ALA of failing instruction
Underflow	23	ALA of failing instruction
Emergency	24	ALA of failing instruction
Inconsistency	25	Description of inconsistency

2.4.5.3 Further actions on Interrupts

The further actions taken, in addition to those described in the previous section (2.4.5.2), depend on the type of interrupt

- Not Vectored Interrupts (and Vectored Interrupts with Vector ≤ 31).

  A branch is performed to the logical program location defined in the following logical program locations:

Interrupt type	Location
CPU interrupt	55
Emergency interrupt	56
Trace interrupt	57
Error interrupt	58
Page interrupt	59
Timer interrupt	60
Context stack overflow	61
Context stack underflow	62

- Vectored Module Interrupts (Interrupt Vectors ≥ 32)

  The interrupt vector is used as an index into an array of 1024 interrupt descriptors. The address of the first location is taken from program location 37. The interrupt descriptor defines a device handler index and a view:

  The view defines a program and a data Translation Table Register (TTR) to be used during the interrupt handling. The program and data Translation Table Registers are loaded by those values defined by the view, and the former logical data page 63 is mapped in as the new logical data page 63.

The device handler index which is a number in the range 0 to 1023 is used as an index into a device table. The address of the first location in the device table is taken from program location 38. This table defines for each device handler index a pair of a device handler procedure and a device control block by means of two absolute logical addresses contained in two consecutive machine words.

The PRPC register is loaded with the address of the device handler procedure and register 4 is loaded with the address of the device control block.
The priority field in EXR is updated with the value defined for the CPU in the MAP module (i.e. the priority associated with the interrupt vector).

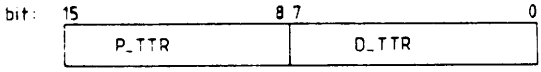
2.4.5.4 PROCESS PARAMETER BLOCK

The Process Parameter Block (PPB) contains process related variables which are used by the CPU.
The PPB is always placed in the highest logical data addresses and is confined to a single page (1024 words).

The PPB variables used by the CPU hardware/firmware are:

CPU Translation Register Map (TRM)

The CPU Translation Register Map is used by the CPU to translate logical views (i.e. combinations of program and data spaces) into sets of translation registers. The TRM contains 16 entries corresponding to 16 views (0 through 15). Each entry consists of two bytes:



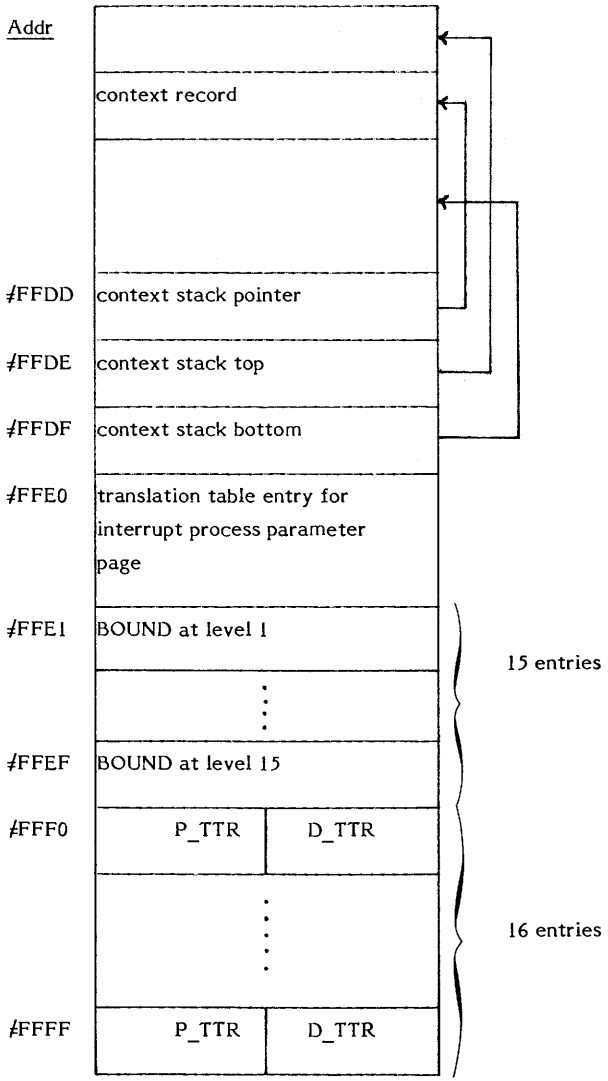
P_TTR and D_TTR defines the values of the program and data Translation Table Registers corresponding to the view. These values must be in the range 0...63.

BOUND Register Map (BRM)

The BRM defines BOUND as a function of the SYSTEM level.

The BRM is an array of 15 words contained in absolute locations FFE1 through FFEF. The BRM is indexed by the SYSTEM level (field LEVEL in the EXR register). The BRM is used by the CPU when executing MON, RTM and RETurn instructions.

Layout of Process Parameter block

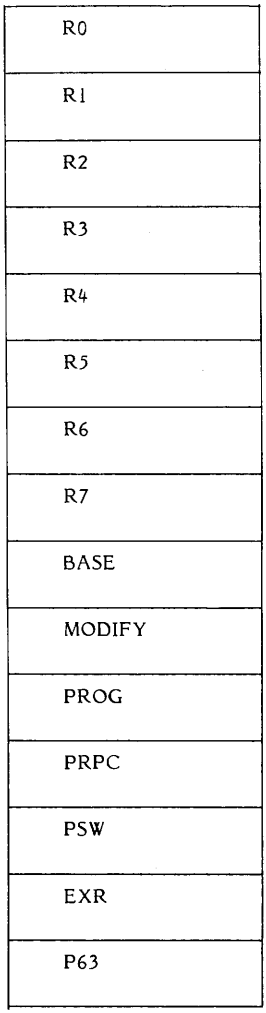


2.4.5.5 Context Stack

the Context Stack is used by the CPU for stacking of CPU Context Records. The Context Stack is used by the instructions MON, RTM, CALL and RETurn.
The Context Stack is administered by a Context Stack Control Block which consists of:

- Context Stack Pointer (CSP)
- Context Stack Top (CST)
- Context Stack Bottom (CSB)

A Context Record contains saved CPU registers. The layout of a Context Record is:



physical address of logical data page 63

2.4.6 The CACHE Memory

2.4.6.1 General

As previously said, the CACHE memory concept relies on the basic principle that, in the step execution of a program, a CPU originated read operation at an address has a high average probability of being accessed shortly again. Fundamentally, the CACHE memory provides a small high speed buffer memory containing copies of previously accessed memory locations in PU part of Processing Element Memory. Addresses and corresponding contents accessed by the CPU are stored in the CACHE memory, which reduces accesses to slower main memory, increasing the overall system speed.

Two advantages from using the CACHE memory are obtained:

- The processor bus loading is reduced, making use of a greater number of CPU's possible before bus contention occurs. A maximum of 5 CPU/CACHE modules may be used in one Processing Element without continuous P-Bus contention.

- Average access time of memory read operations is reduced, thus improving the throughput of each CPU.

2.4.6.2 Cache Memory Organization and Maintenance

The CACHE memory functional diagram is shown (fig. 2.4.6.2-1) overleaf.

The CACHE memory contains 1K locations which are updated on a "one word at a time" - basis. Each location is capable of storing a copy of one main memory location, tagged with both the logical and physical address information identifying the main memory location copied.

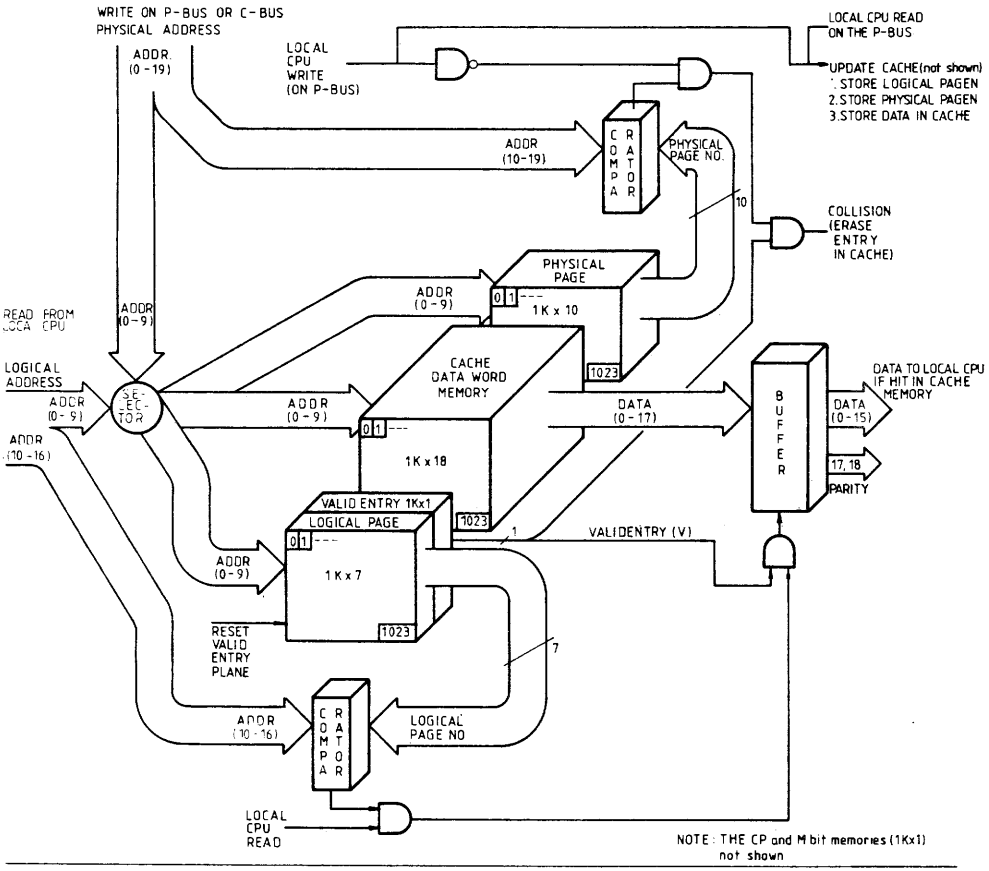


Fig. 2.4.6.2-1 CACHE, FUNCTIONAL OVERVIEW

The CACHE is addressed by the 10 least significant bits, which are common to logical and physical addresses for any given memory location.

At memory read operations initiated by the local CPU, the CACHE memory is read prior to accessing the main memory. If a copy of the main memory location is present, the data are presented to the CPU, not involving main memory access. If data are absent in the CACHE, the main memory is read, and the data presented to the CPU and copied by the CACHE. Only successful reads to the main memory in the PU cause CACHE updating.

Memory Write operations initiated by the local CPU to a memory location already copied in the CACHE, cause the main memory and the CACHE locations to be updated.

Memory Write operations initiated by other modules (CPUs or C-Bus DMAs) to locations copied in the CACHE cause deletion of the copies.

Semaphore Protected memory operations, Privileged operations and operations to PE-memory outside the PU never involve the CACHE.

2.4.6.3 Software Overhead

Generally, the CACHE memory requires no software overhead. However, the software design can be optimized with respect to CACHE memory operations if the following characteristics of the CACHE are observed:

- Whenever the Working Set (logic to physical address MAP) of a process implemented by a CPU is changed, the CACHE memory local to the CPU is cleared by CPU firmware. This applies in all cases of changing the Working Set except when doing this by writing to the MAP module using CIO-intructions (System State level 15 only). When CIO-instructions are used for this purpose, the Cache-memory should be cleared by surrounding the CIO-instructions by Cache-Disable/Cache-Enable instructions:

  - CAD (Cache Disable)
  - Change Working set using CIO-instructions
  - CAE (Cache Enable) (if required)

Consequently, if very frequent changes of the Working Set (frequency < 1/5ms) takes place, disabling of the CACHE during this period of time is an advantage.

- The benefit from using the CACHE memory is highly dependent upon the nature of programs executed, favouring programs containing loop structures not exceeding 1K Words of program space.

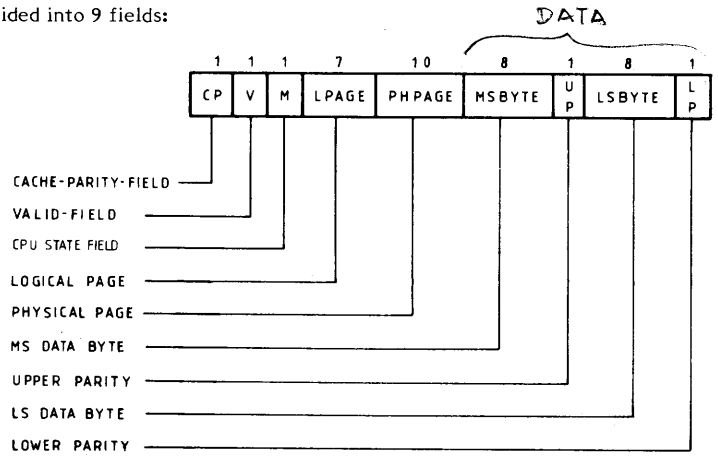
- The CACHE memory is constantly monitored by the CPU, detecting parity errors. In the CPU a Cache Error Register (CER) is provided, which is incremented upon detection of CACHE parity errors. Instructions for:

  - reading CER
  - enabling CACHE memory
  - disabling CACHE memory
  - clearing CACHE memory
  - testing CACHE memory

are provided in the standard instruction set of the CPU.

2.4.6.4 CACHE-Word Format

The CACHE memory consists of a 1K word by 38 bits RAM. Each word is divided into 9 fields:



- The CACHE-Parity-field contains the parity bit necessary to obtain even parity in the CPU STATE- and LPAGE-field (only correct parity if VALIDFIELD is set).

- The VALIDFIELD indicates whether a CACHE location contains valid data or not, "1" corresponding to "valid".

- The CPU STATE FIELD indicates the CPU STATE (SYSTEM/USER) at the time the CACHE-location was loaded, "1" corresponding to SYSTEM mode.

- The LOGICAL- and PHYSICAL-PAGE-fields are address information fields, identifying the page of the address in Local Main memory location copied in the MS and LS DATA BYTE fields.

- The MS and LS DATA BYTE fields contain the actual content of the main memory location copied. The UP and LP-fields contain the corresponding parity bits.

2.4.6.5 CACHE Memory Addressing

The CACHE memory is addressed by using one of the three address sources:

- the local CPU, providing a logical address as CACHE entry,

- another "address sourcing module" in the same PE, using the P-Bus and through the MAP providing a physical address, or

- another "address sourcing module" in the same PE using the C-Bus and through the MAP providing a physical address.

In all 3 cases, the address is (logically) divided into two fields:

XX=19, when physical address
XX=16, when logical address

The Low Order Address (10 bits) designate which of the 1K CACHE locations is to be accessed, while the PAGE-field (physical or logical, depending on the address source as described above) is compared to the corresponding page-field in the CACHE word, in order to detect whether a copy of the addressed Memory location is present in the CACHE.

Thus, each of the 1K CACHE locations is shared between PU part of PE-Memory locations with identical LOW ORDER ADDRESS, providing a unique mapping function from any given Memory address in the PU to a CACHE location, and a one-to-many mapping from a given CACHE location to PU part of PE-Memory. Always retaining in the CACHE for a specific LOW ORDER ADDRESS (0 -1023) the contents of the corresponding address in the most recently accessed page in Memory.

The many-to-one correspondence principle is shown in (figure 2.4.6.5-1) overleaf.

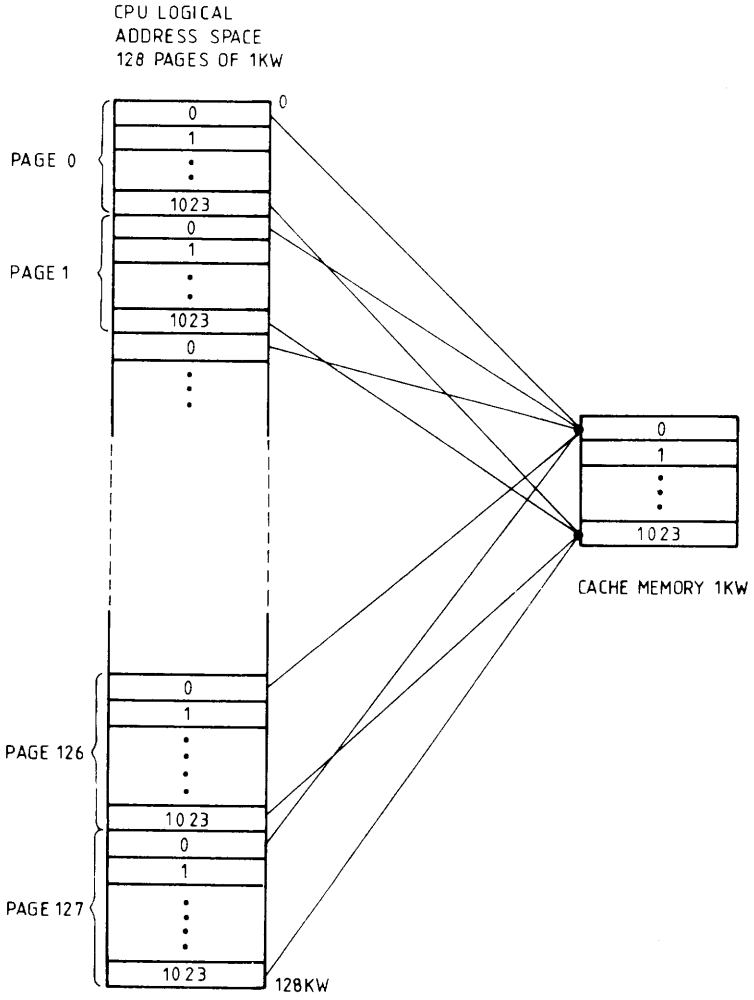


Fig 2.4.6.5-1
Many to one correspondence principle of CACHE Memory

2.4.6.6 CACHE-Operations

- Non-Semaphore-Protected Read Operations

When the CPU initiates a non-semaphore-protected read operation, the 10 LOW ORDER ADDRESS bits from the CPU (AR(9:0)) are used for selecting a word in the CACHE. The word is read, and the contents of the data fields presented to the CPU, provided:

- the valid indication is set
- the CPU state equals the STATE_FIELD stored with the accessed word in the CACHE or the CPU is in SYSTEM state
- The logical page from the CPU (AR(16:10)) is equal to the logical page stored in the LOGICAL PAGE field.
- No CACHE parity error is detected.

If all of these conditions are met, a HIT is said to take place, enabling the CPU to continue operation at maximum speed, without requiring P-Bus access.

If anyone of the tests fails, a P-Bus read operation is initiated. During the bus cycle all data, parity, address and CPU-mode information is assembled by the CACHE-controller and stored in the CACHE, setting the VALID-bit.

In case of an unsuccessful processor bus operation (timeout, parity, "no-access" or "page-fault"- error) the CACHE ignores the bus-operation, not updating the CACHE.

- Semaphore Protected Read Operations

  Semaphore-Protected CPU read Operations always require main memory access, in order to be subjected to the control of the "P-Bus Authority Control"-unit. Data accessed by the CPU using a semaphore protected read-instruction are copied by CACHE.

- Write-Operations

  When the CPU issues a write-operations, a P-Bus cycle is always initiated. The CACHE-controller detects this situation and performs a read operation in the CACHE location specified by the LOW ORDER ADDRESS (AR(9:0)). Concurrently, data, parity and CPU-mode are assembled, and the new data and parity written into the CACHE-location, provided:

  - The VALID-bit was set
  - the CPU state equals the STATE_FIELD stored with the accessed word in the CACHE or the CPU is in SYSTEM state
  - the contents of the LPAGE-field equals the logical page supplied by the CPU.
  - no faults occur during the P-Bus operation.
  - no CACHE-parity error is detected.

  If any of these conditions are not met, no CACHE updating will take place.

  If a CACHE parity error is detected or if a fault during the bus operation occurs, the contents of the CACHE location is deleted.

Thus, only successful CPU-write-operations involving main memory locations previously copied by the CACHE will result in CACHE-updating.

CACHE memory content may be deleted in individual locations or in all locations.

- Deletion of Single Location

Deletion of a single location takes place whenever a write operation initiated by an "address sourcing module" other than the CPU connected to the CACHE, and involving main memory locations copied in the CACHE, takes place. The CACHE controller continuously monitors the P-Bus and the C-Bus, detecting successful write operations, comparing the physical addresses to those contained in the CACHE and deleting CACHE-locations corresponding to PU part of PE-Memory locations having been updated (by resetting the valid-bits).

- Deletion of all Locations

Whenever the logical to physical address mapping function for a CPU is changed by the CPU, the complete content of the CPU's CACHE memory is deleted (by resetting all valid bits), in order to maintain the uniqueness of the mapping.

2.4.6.7 CACHE Timing

the CACHE controller/memory monitors all main memory operations in addition to servicing the local CPU, and may therefore have difficulty in servicing (updating/deleting/loading) the CPU activity, if a certain frequency of activity is exceeded. This limit cannot be determined exactly, as it depends on the distribution of the activity.

It must be noted, however, that exceeding the limit does not cause a system malfunction, but only a degradation of the benefit from using the CACHE memory (provided the absolute timing requirements are kept).

Timing characteristics:

- Shortest permissible delay between 2 successful write operations on each of the P-Bus/C-Bus; 250 ns, measured between two subsequent positive edges of TRQ.

- CACHE memory access time with HIT (from CPU initiating read operation to data loaded into CPU):

  MIN.: 125 ns
  TYP.: 125 ns
  MAX.: dependent on P-Bus and C-Bus activities.

- Access time prolongation when MISS (from CPU initiating read operation to data loaded into CPU), compared to access time without CACHE memory:

  MIN.: 125 ns
  TYP.: 125 ns
  MAX.: dependent on P-Bus and C-Bus activities.

- Prolongation of write operation duration caused by CACHE memory: 125 ns.

2.4.6.8 CACHE/CPU Failure Detection Facilities

The CACHE section is equipped with a number of facilities for fault detection/prevention:

1) With the CPU performing a main memory read operation, the CACHE memory will normally be loaded with data, parity, address, etc. information. In order to prevent faulty data from being loaded into the CACHE memory, a parity check is performed on the data received from the P-Bus prior to storing these in the CACHE memory. Revealing a parity error, the CACHE controller aborts the CACHE write operation.

2) All CACHE memory locations contain a parity bit (the CACHE parity field) which is loaded concurrently with loading/modifying the remaining fields in the location.

   When the CACHE controller performs a CPU read operation in the CACHE memory, the parity is checked.

   Detecting a parity fault, the event is signalled to the CPU, and a main memory read operation is initiated, causing the CACHE memory to be loaded with, and supplying the CPU with the correct data.

3) When ordered to do so by the CPU, the CACHE controller performs a CACHE memory test sequence. Once initiated, the sequence is carried out independently of the CPU and delivering the test result to the CPU in the IDR register. The CACHE test sequence is destructive, i.e. deletes the contents of the CACHE memory.

4) The CPU may at any time disable the CACHE memory and turn on the "CACHE OFF-LED". This feature enables the CPU to monitor the CACHE memory "health", disabling the CACHE if unacceptable high error rates during normal operation are detected, or if the result of the CACHE test sequence revealed an error. Immediately after disabling, the CACHE memory (and registers in the CACHE section) are cleared, prepared for future enabling.

5) After disabling, the CACHE section may be enabled by the CPU. A standard instruction in the CPU instruction set is included for this purpose.

The above mentioned hardware facilities provide the necessary means for controlling and monitoring CACHE memory operation by means of CPU firmware, transparent to or under control of software as required.

2.4.7 CPU Micro Architecture

A short overview of the CPU Micro Architecture is given in this section in order to assist the user in evaluating the feasability of microprogramming his own instructions for inclusion in the CPU alternative instruction set.

2.4.7.1 CPU/CACHE Outline

The CPU/CACHE consists of 6 main sections, which communicate using a set of data and control lines, as outline in fig. 2.4.7.1-1 overleaf.

Data/address buses are drawn using unbroken lines, while control paths are dotted lines.

The CPU performs all the traditional duties of a CPU, possessing the "initiative" of the module. All data processing takes place in this section. Data in/out to/from the CPU are transferred through the four 16-bit registers:

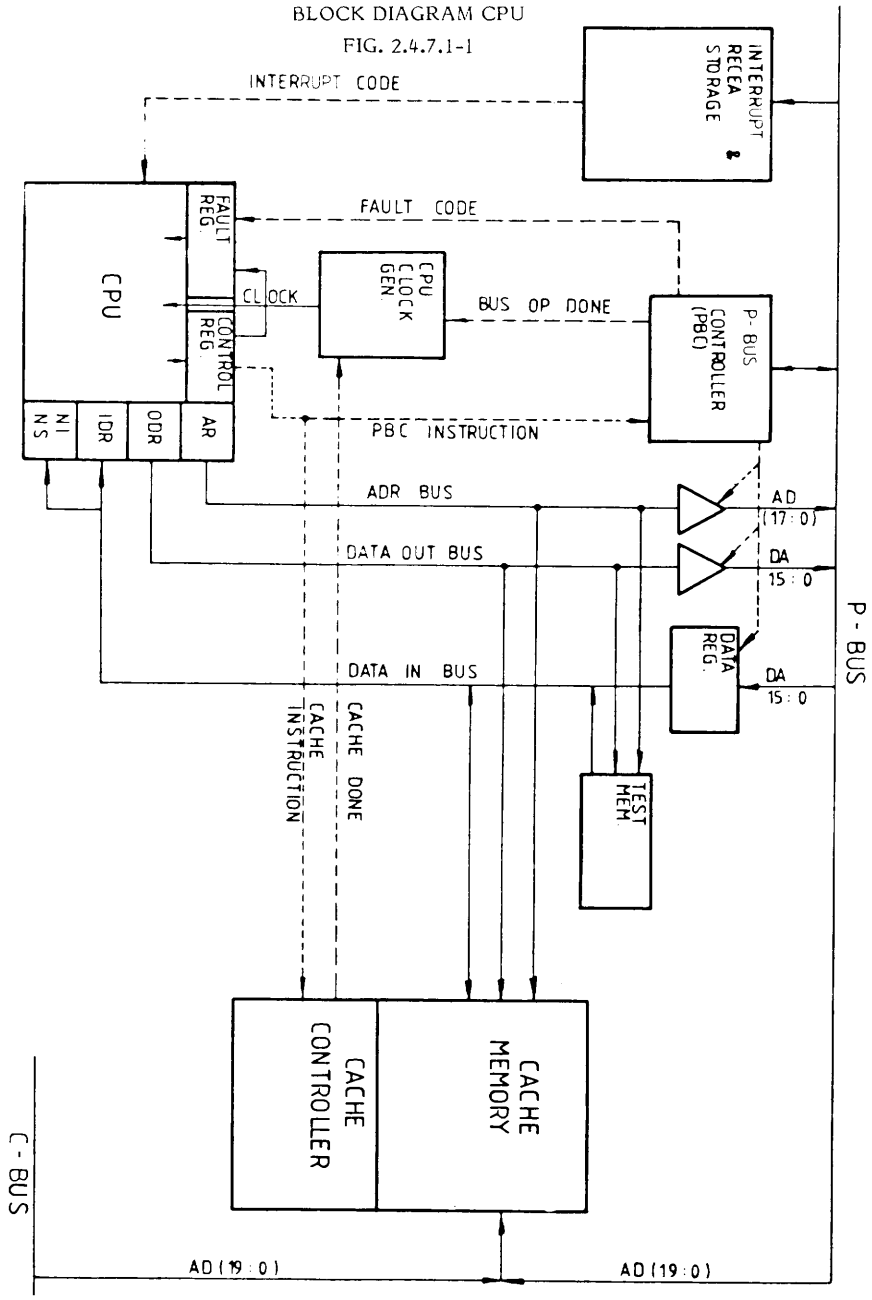
AR: Address Register,
ODR: Output Data Register,
IDR: Input Data Register,
NINS: Next Instruction Register.

All the registers are unidirectional, writing in the AR and ODR being controlled by the CPU, while writing in the IDR/NINS is controlled by the CACHE Controller, or the PBTC (P-Bus Timing Control), depending on which of these at any given instant is sourcing the DATA IN BUS.

The CPU transmits instructions to the PBTC and the CACHE Controller using the control register, the contents of which is continuously monitored by the PBTC and CACHE. Having loaded the relevant Data/Address registers, the CPU loads the Control register, thereby signalling to the PBTC and CACHE

BLOCK DIAGRAM CPU

FIG. 2.4.7.1-1



Controller, which of these are to process the contents of the data/address registers, and informing the CPU Clock generator that an operation external to the CPU is in progress.

While the PBTC and/or CACHE operation is executed, the CPU Clock generator monitors the CPU to detect premature CPU access to those of the registers (AR, ODR, IDR, NINS) being used as buffers, halting the CPU if such an access is attempted.

Consequently, communication between the CPU and the PBTC/CACHE is fully handshaked, allowing the CPU firmware to be designed neglecting the timing characteristics of the above mentioned sections.

The PBTC (P-Bus Timing Control) section competes with other "address sourcing modules" to obtain P-Bus access, and when granted, performs the actual data transfer.

In addition to this, the PBTC detects fault conditions, which may arise during P-Bus operation; loading the FAULT REGISTER with information identifying the fault cause, and forcing the CPU to jump (in firmware) to a fault handling routine.

The CACHE memory/controller is not described, as it is not available for user microprogramming.

The Interrupt Reception and Storage section (IRS) monitors the Interrupt/-Notification lines; detecting, storing and masking interrupts, to produce an interrupt code identifying the interrupt with the highest priority of relevance to the CPU, which is not masked by the Process Status Word (PSW).

The interrupt code may be used by the CPU for u-Program branching. An interrupt being serviced by the CPU is cleared in the IRS.

The test memory may contain a CPU test routine, which is executed by the CPU at Master Clear. The test routine is executed with all connections to the P-Bus disabled, only enabling these (and disabling the test memory) if the test is terminated successfully.

Consequently, the CPU/CACHE may be thoroughly tested upon Master Clear, before the module is allowed to use the P-Bus, thereby minimizing the risk of CPU malfunction resulting in a P-Bus lock.

2.4.7.2 The CPU

2.4.7.2.1 CPU Outline

Functionally, the CPU is divided into two main sections:

- the CPU control section

and

- the ALU/data Path/register section

These two main sections will be briefly described in two subsequent chapters.

2.4.7.2.2 The CPU Control Section

Generates the micro instruction to be executed in the following clock cycle, i.e. when loaded into the microinstruction register and the four registers:

- IBUS microinstruction register,
- A address select and application control register,
- B address select register and
- MCU instruction register

The microprogram address MIA (11:0) is generated by the MCU (Micro program Control Unit) and by the "feed-back control circuit", controlled by the microinstruction fields FB (11:0), MCI (3:0) and MCJ (1:0).

The CPU control section provides the following main facilities for micro-program address generation:

- Microprogram subroutine facilities:
  In the MCU (AM 2910) a 5-level LIFO stack is included, in which the microprogram return address may be saved, when a subroutine call is executed. A microprogram subroutine call may be executed conditionally, with the contents of the "arithmetic flag status register" (which constitutes part of the Process Status Word) or with the instruction code register (OPREG) bit 3 as the condition.

- Jump to any of the 2K microprogram locations in one clock cycle.

- Multiway jump facilities, controlled by:

  - 9 instruction code bits (OPREG (15:7))
  - PSW11 (the system/user state bit),
  - PSW12 (alternative instruction set) and
  - a bit from the micro program register, FB1.

This jump facility enables a fast identification of the instruction code presently contained in the OPREG. Note that PSW11 is used as a parameter in the multiway branch. In the CR80, PSW11 specifies the state of the process being executed by the CPU as one of the two: System/User State.

The action to be taken by the CPU when decoding and executing an instruction depends on the state of the CPU. This is why the state is incorporated as a control parameter in the instruction decoding.

The multiway-jump facility is implemented using a 4K x 8 PROM, generating the lower 8 bits of the microprogram address.

- The interrupt code from the "interrupt reception and storage" section of the module may be used as the lower 3 bits of the micro program address. The "Feed-Back Controller" is designed to ensure that no time is lost testing whether interrupts are pending or not.

This is done by using a composite microinstruction for:

- testing interrupts and
- performing the previously mentioned multiway branch on the opcode, etc.

The instruction performs a test (in hardware) on the interrupt code. If interrupts are present, the interrupt code is used for generating the next micro program address. If no interrupts are pending, the multiway branch on the opcode is executed.

- Facilities for testing selected fields of the instruction code.

- Facilities for testing the LS bit of the Q-register in the 2901 ALU slices. This is used for integer multiplication, making it possible to shift and test the mutiplier, controlling the subsequent shift and additions of the multiplicand.

- Facilities for testing the CPU state PSW11 separately.

- Facilities for testing the contents of the fault register. This is used upon unsuccessfull PBTC P-Bus operations, which unconditionally force the MCU to microprogram address Ø. Due to the fact that no microprogram return address is saved in this situation, information identifying the type of the faulty processor bus operation cannot implicitly be held in the microprogram, which is why a separate register (the fault register) for

- saving information identifying the type and parameters of the bus operation

and

- information identifying the fault cause

is necessary.

2.4.7.2.3 The ALU/Data Path/Register Section

Consists of the following subsections (all 16 bit wide):

- two ALUs:
  the ALS and the AMS ALUs.
  Both are able to execute a number of logic and arithmetic functions.

- two 16 x 16 bit register files:
  the ALS and the AMS register files.
  The ALS file is a two-port register file, allowing simultaneous read in two different locations and write in one of these, i.e. operations of the type:

  (B) := (A) * (B)

  The file addresses may be sourced by the microprogram directly, or by the OPREG, i.e. from the instruction code.

  The AMS register file is primarily intended as the physical storage space for those of the CPU registers visible to the programmer that are involved in main memory address calculation (BASE, PROG, etc.).

- The NINS register:
  This is intended to be used as an intermediate register on the instruction code path from the memory to the OPREG, allowing the instruction code of the next instruction to be fetched, while the current instruction code is held in the OPREG during execution.

- The mask PROM register file:
  This register file is only readable and, as the name implies, is intended for masking operands supplied through the OBUS to the AMS ALU with one of 32 pre-programmed masks.

  The register file has separate input and output ports and an addressing scheme allowing reads and writes in two different locations (with some limitations when selecting the pair of registers).

  The register file is writeable concurrently with loading the AR register, a feature to be used for maintaining a copy of AR contents. This feature is a "must" in the CR80 system in order to make it possible to regenerate the memory address of a P-Bus operation (initiated by the CPU) upon a bus transfer error.

- A number of special purpose registers:

  - The OPREG, containing the instruction code of the current instruction and connected to the CPU control section.

  - The Process Status Word Register.

  - The TR register, connecting the ABUS and the IBUS, and providing byteswap facilities.

  - The AR register

  - The ODR register

  - The IDR register

2.4.7.2.4 The PBTC (P-BUS Timing Control Section)

Performs the following functions:

- Communicates with the arbiter in order to obtain P-Bus Access, when requested to do so by the CPU.

- Performs the actual bus operation (i.e. communication with "slave") when P-Bus access is granted.

- Detects faults, which may arise during bus operations, loading the fault register with information identifying type and parameters of the faulty operation and the fault cause,

and

- Forcing the CPU to jump to microprogram address Ø.

2.4.7.2.5 Interrupt Reception and Storage Section

Performs the following functions:

- Monitoring the PU control bus, detecting interrupts/notifications of relevance to the module. This comprises:

  - latching of timer interrupts and

  - latching notifications directed at the CPU, i.e. latching of notifications transmitted by the MAP module with the CPU/CACHE module's CPU # as the value of the INT (2:0) lines.

- Monitoring the CACHE "Parity-Error-Status" output, latching the condition, if an error occurs.

- Monitoring the Process Status Word, bit 1 (PSW1), detecting an active "trace flag" (= PSW1).

Interrupts/notifications received from the PU control bus are masked by PSW (14:15), allowing interrupts/notifications to be latched unconditionally, but preventing these from being transmitted to the CPU, if the corresponding mask bits in the PSW are set.

The "CACHE Parity Error" signalling to the "interrupts reception and storage" section may be disabled by a hardware switch. This causes no malfunction of the module if errors occur, but of course prevents the CPU from monitoring the CACHE "health".

Those of the interrupts/notifications, etc. that are not masked by the PSW are encoded into a 3 bit interrupt code, which may be tested by the CPU Control section. The interrupt code identifies the pending interrupt with the highest priority.
When using the interrupt code for microprogram address generation, the interrupt being serviced is cleared.

2.5 CR80 MEMORY MAP MODULE

2.5.1 General

The Memory MAP Module (MAP), together with it's associated Map Interface Adapter Module (MIA) contains all PE central system functions, such as:

- Timing and control
- P and C-bus Authority control (arbitration)
- Datachannel interface
- Logical to physical Address translation
- INTRA MEMORY DMA, control and multiplexing
- Interrupt Preprocessing
- V24 PE-System console I/F
- Bootstrap PROM
- PE-initialization and selfcheck control

In order to facilitate the following discussion of the above, a block diagram of the MAP module is included overleaf.

In order for the MAP module to perform it's central functions in the PE, all PE-buses (P-bus, C-bus and Datachannel) are connected to it. Also, it include it's own MAP processor for handling of INTRA MEMORY DMA, Interrupt Preprocessing, system console, and PE initialization.
The Address sourcing devices connected to the MAP is allowed to access the total PE-Memory, and privileged control memory of the MAP, C-Bus DMA's and Peripheral Modules in the following ways:

1) The CPUs on the P-bus may by privileged instructions access:
   The Translation Table Registers (TTR)
   The translation tables (TT)
   The access status registers (AS)
   The MAP processor RAM (MP RAM)

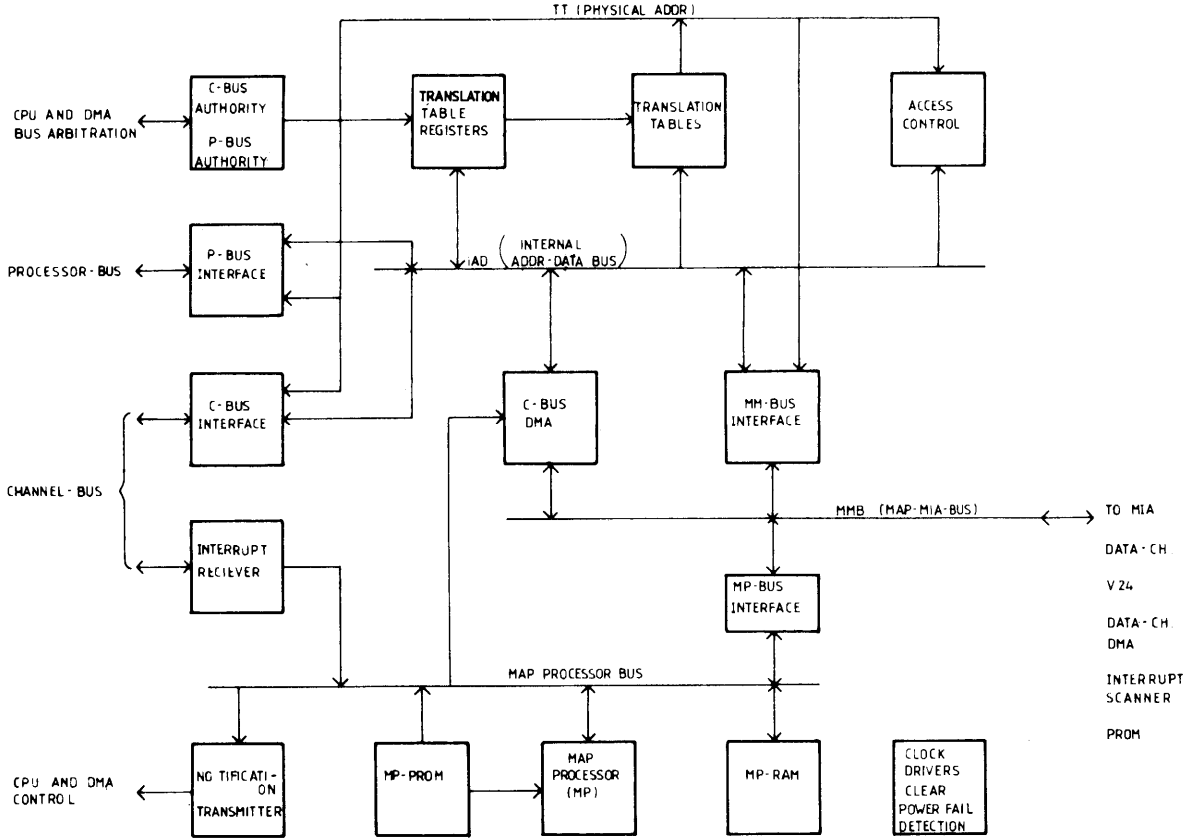


Fig. 2.5.1-1 Memory MAP module, block diagram

By privileged instructions, the CPU's may access:

C-Bus DMA's on the C-Bus

Peripheral Modules on the Data Channel

By memory operations the CPU's may via the logical-to-physical address translation, access PE-Memory:

Memory on the P-Bus

Memory on the Data Channel (RAM and Peripheral Modules)

2) The C-Bus DMA modules on the C-Bus , may by privileged MAP operations access:

Translation Table Registers (TTR)

Access Status Registers (AS)

Translation Tables (TT)

By memory operations the DMA's may via the logical-to-physical address translation, access PE-Memory:

Memory on the C-Bus

Memory on the Data Channel (RAM and Peripheral Modules)

3) The INTRA MEMORY DMA within the MAP may via the logical-to-physical address translation, access PE-Memory:

Memory on the C-Bus

Memory on the Data Channel (RAM and Peripheral Modules)

4) The MAP processor is allowed to access:

All devices connected to the MAP Processor bus (MP bus)

The Interrupt Receiver and Interrupt Scanner

Translation Table Registers (TTR)

Access Status Registers (AS)

Translation Tables (TT)

Control memory of C-Bus DMA's on the C-Bus

Control memory of Peripheral Modules on the Data Channel

PE-Memory on C-Bus and Data Channel

Access Arbitration

The P-Bus has the highest priority, but will not get two consecutive cycles, if other requests are active.

The C-Bus, the DMA and the MAP processor have equal priority, and a cyclic arbitration is used among these.

2.5.2 CR80 Memory Management

2.5.2.1 General

The concept behind the memory management features available with the CR80 Processing Element is that of "Logical-to-Physical Address Translation". The amount of memory required by a process is defined to be its "view" or "logical address space". This space may be as large as 2 x 64 1K pages. The areas of physical storage assigned to the process are defined to be its "physical address space". The address translation function which converts the addresses in the logical space to the addresses in the physical space is called the view for that user. The view is implemented by two Translation Tables which each contains 64 entries. Each of these entries defines the physical address of a logical 1Kword memory page. The view is totally under control of the system software. Each 1K page can be read and/or write protected. In addition, there are separate views for INTRA MEMORY DMA, Inter Processing Element DMA's (S-NET).

The CR80 Advanced Multiprocessing Operating System (DAMOS) determines what these maps are to be, and then relays this information to the address translation hardware. The figure (2.5.2.1-1) overleaf shows the logical-to-physical address translation.

The mapping ensures total integrity, security, and protection against interference by other users, e.g. programs being debugged while the system is running in actual on-line operation. Beyond being able to be read and/or write protected, each 1K page separately can be marked "absent" giving rise to a page fault interrupt when accessed. This feature is utilized in DAMOS to implement an efficient Virtual Memory System, allowing Application Software requiring a large accessable address space to run on smaller memory size CR80 computers. The page fault interrupt issued when a page not resident in memory is addressed by a CPU, will initiate that the page is rolled into memory from discs.

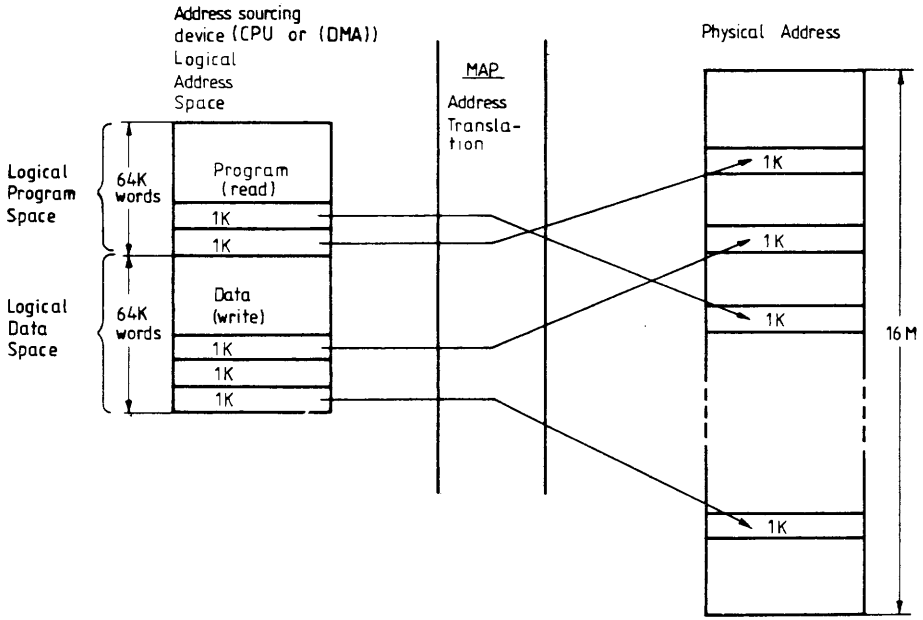


Fig : 25.2.1-1 Address sourcing device logical address space

4-1652

Each of the up to 5 possible CPUs in a Processing Element (PE) works with its own private pair of Translation Tables. One is allocated for program, i.e. the collection of machine instructions compiled to perform a certain function, while the other is allocated for the data i.e. the collection of constants and variables and data required to execute the function.

Likewise, each of the up to 5 possible C-Bus DMA's are prevented from interfering with users by working with its own private Translation Tables. One of the segments is allocated for read accesses to memory, while the other is allocated for write accesses to memory.

The Memory Map address translation circuitry, can contain 64 translation tables. This means that change of user (process switch) on a CPU (or DMA), and the associated change of view, often can be done just by changing the CPU's Translation Table pointers (TTR pair), not having to load the Translation Tables of the user view. This feature is used by DAMOS for allways having its Translation Tables resident in the Memory Map.

2.5.2.2 Address Translation

Memory addresses generated by the address sourcing modules, CPU's on the P-Bus and DMA's on the C-Bus, are translated from the logical address space (128K words) to the physical address space (16 mega words) by means of a set of Translation Table Register (TTR) pairs and address Translation Tables (TT) located in the MAP/MIA modules. Beside the address translation performed, the modules also protects against unauthorized accesses by means of protection bits associated with the physical address. The implemented translation function is illustrated overleaf (figure 2.5.2.2-1) and described below.

The address translation functions of the Memory Map is multiplexed between the P-Bus and C-Bus. The address sourcing device (CPU or DMA) issuing the logical address to be translated, by its device no. points to a Translation Table Register (TTR) pair. One of these registers is selected, depending on for CPUs: Program or data location addressed, for DMAs: read or write operation. The 6 bit contents (Translation Table no.) of the selected Translation Table Register points to the Translation Table (64 words), in the MAP Translation Memory. The six most significant bits of the logical address to be translated, then points to the actual word in the Translation Table. The output, 18 bits from the MAP translation memory, consist of:

2 bits for parity check
2 bits for access protection
14 bits which will form the most significant part of 24 bit physical address.

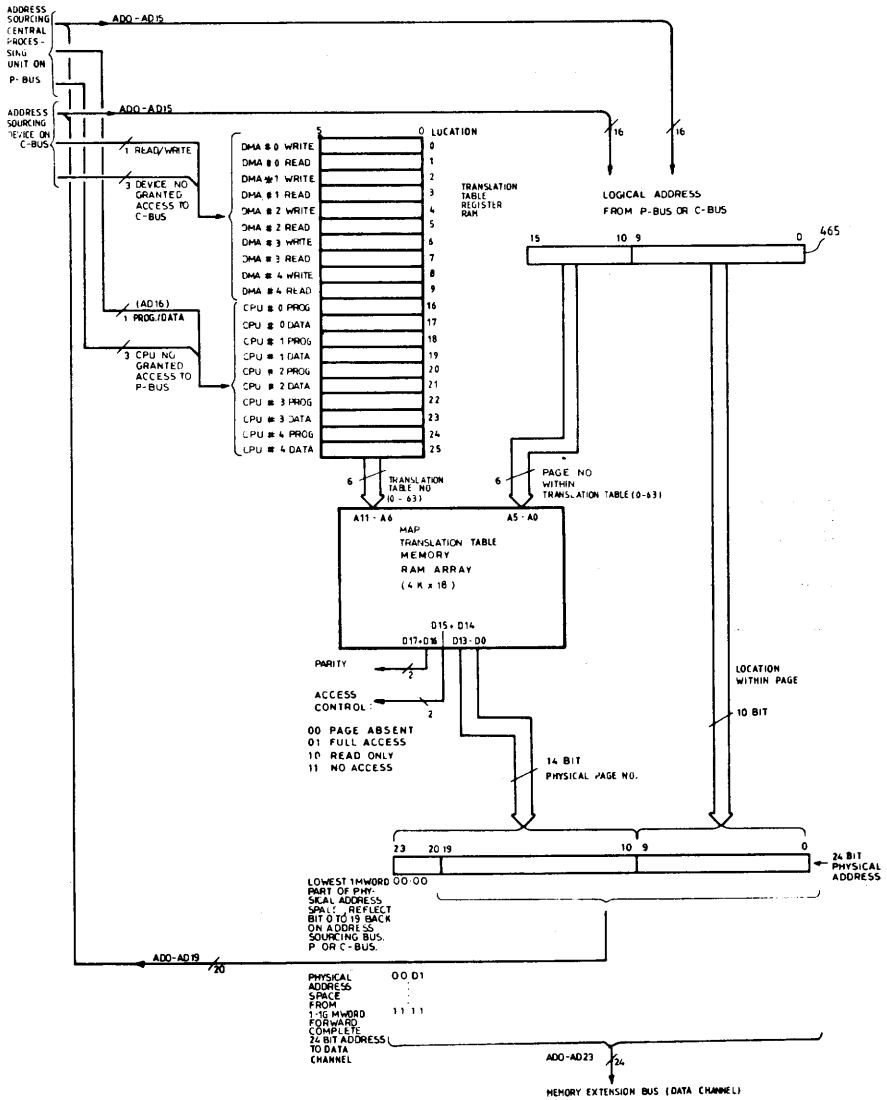
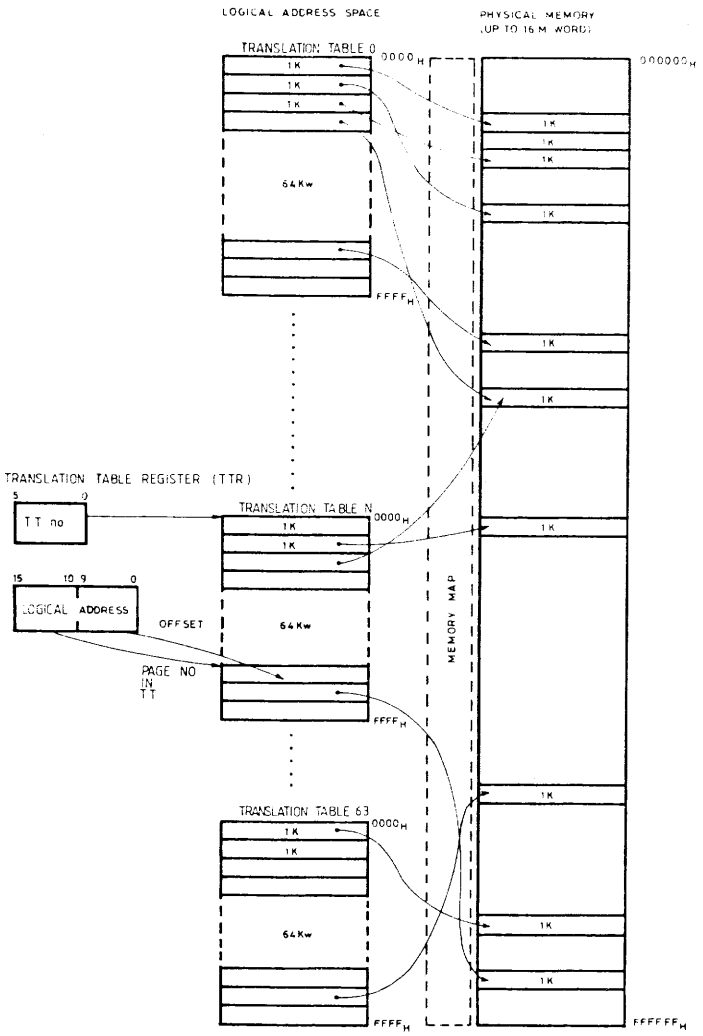


Fig 2.5.2.2-1 Processing Element Memory Map

3-3032



Processing Element
Logical to Physical Memory Mapping
Fig 2 S 2 2-2

The 10 least significant bits of the logical address to be translated, are not translated by the Memory Map, and constitute unchanged the 10 least significant bits of the physical address, while the 14 address bits output from the MAP translation memory forms the 14 most significant bits of the 24-bit physical address.

If the 4 most significant bits of the 24 bit physical address is 0, the addressed part of PE-Memory is located in the PU (1 Megaword max.). The address bits on the bus (P-or C-Bus) forwarded to the MAP module for translation by the address sourcing device (CPU or DMA) are three-stated, by signal from the MAP module to the Address Sourcing module, except for the 10 least significant bits. The MAP module now places bits 10-19 of the 24 bit physical address on the bus, forming a 20-bit physical address (1 Megaword) available for addressing in the PU.

If the 4 most significant bits of the 24 bit physical address is not 0 (1-15), the addressed part of PE-Memory is located in CU's (RAM or Peripheral Modules) and the complete 24-bit physical address is forwarded on the PE Data Channel.

During address translation the access protection bits are checked against the access rights of the address source:

Protection Bits		CPU in System State (AD17=1)	CPU in User State (AD17=0) or DMA access
0	0	Page absent	Page absent
0	1	Full access	Full access
1	0	Full access	Read only
1	1	Full access	No access

If an unallowed access is attempted, the transfer is terminated, without releasing physical address to buses, and the CPU's or DMA's are notified, and will fetch the error cause by microprogram routine (refer to the CR80 interrupt system description, section 2.6).

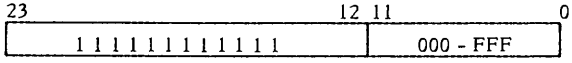
2.5.2.3 PROM

To support the system during initialization a 4K word PROM is included, which is made available for the CPU upon master clear by means of address translation set up by the MAP/MIA module.
The bootstrap loader is resident in the PROM, allowing the system to be loaded from disk, floppy disk or the V24 communication port at system initialization time.

The 4K PROM located in the MIA forms 4K words of the total 16-megaword physical memory space. Thus the logical-to-physical address translation may point to any of these four PROM pages as well as any RAM page.

- Physical address

The physical address of the 4K PROM is:



Corresponding to the upper 4 pages of the 1-megaword memory located in crate 15 (crate F) or, in other words, the uppermost 4 pages of the 16384 total pages available.

2.5.3 MAP Processor

The MAP processor (MP) handles interrupts, DMA transfers, AV24 communication port, self-test, system start-up and access status.

Communication with the CR80 CPUs located in the Processing Unit is carried out via the MP-RAM, which is accessible from the P-Bus using privileged MAP instructions.

Interrupt Processing is performed in accordance with the following section on the CR80 interrupt system, with all tables located in the MP-RAM. The MP fetches interrupt vectors from the interrupt Receiver and Interrupt Scanner in MIA, and CPU's are notified via the Notification Transmitter.

DMA (Intra Memory) transfers are setup by the MP upon request from CPU's. For this purpose the MP uses the INTRA MEMORY DMA in the MIA.

The AV24 Port may be used either as PE system port or as peripheral port for the MP.
In the first case, the MP handles blocking of data with buffer areas located in MP-RAM, sets up baud rate and other control variables, and controls the communication between the PE and the external device.
In the second case, the MP executes commands from the AV24 port, thus providing a tool for debugging the Processing Element (PE), since the MP may access all memory and modules accessible in the Processing Element (PE).

Self-Test and System Start-Up is performed by the MP upon Master Clear of the PU. The procedure includes:

- Check of all registers and RAM areas in the MAP
- Check of most important control functions
- Initialization of registers and RAM resident tables, including Translation Tables
- Activation of a CR80 CPU in the PE.

MP RAM

This forms the MAP processor memory space of 8K bytes, connected to the MP bus.
The P-Bus may access this memory by privileged MAP instructions, which are in fact executed by the MP.

2.5.4 INTRA MEMORY DMA

To support Memory block moves within the CR80 Processing Element a DMA channel is included in the MAP/MIA modules. The DMA is used for transfer of data within the PE-Memory (Memory in PU or on the Data Channel (RAM and Peripheral Modules)) and is controlled by the DAMOS system software.

Addresses generated by the DMA channel are, like the remaining address sourcing modules in the system, connected from the logical space to the physical space by the memory mapping function.

The DMA can move buffers of up to 16K words and has a maximum transfer rate of 1.2 Megabyte. The DMA supports byte addressing, as the buffer start and ending points are defined as byte addresses.

The required set up of the DMA are defined in the MAP module data sheet.

2.5.5 AV24 Communication Port

The AV24 Port included in the MAP/MIA modules has the following specifications:

Data mode:	asynchronous
Character length:	7 bits
Parity check:	even
Parity generated:	even
Stop bits:	1
Baud rate:	Selected by MIA 4 bit DIL switch.

The AV24 port may operate in either of two modes:

1. PE port to system console, maintenance and Configuration Processor (MCP), diagnostic load facility (such as cassette tape), hard copy terminal, etc.

2. PE off-line Maintenance connection making all the MAP, memory and modules directly accessible from a console.

Mode 1 is selected, when the MAP is in normal mode.
Mode 2 is selected, when the MAP is in maintenance mode.

Connection to the AV24 port is via a 25p, D-type connector on the MIA front panel.

2.5.5.1 Normal Mode

When the AV24 Port is selected to operate as a PE System Port, it is accessed from the P-Bus via the MAP Processor RAM. The MAP provides buffering of input and output data, in full duplex, at transfer rates up to 9600 baud.

Input and output are separated functions, but use the same interrupt vector.

2.5.5.2 Maintenance Mode

In mode 2, the AV24 port is used to issue commands to the MAP for PE off-line test and diagnostics.
This mode is selected, whenever the MAP is in maintenance mode.

The MAP accepts following commands from the AV24 port:

H	-	Halt PU
R	-	Run
S	-	Single Cycle
N	-	Notify CPU
M	-	Move Data

The general command format is described overleaf.

- Command Format

The format is defined by the syntax description below:

<command>:: = <command code> $\{<param>\}_0^2$ <CR> $\{<data>\}_0^\infty$

<command code> :: = H R S N M

<param> :: = $\{\{<sp>\}_0^\infty <digit>\}_1^6$

<digit> :: = 0 1 2 3 4 5 6 7 8 9 A B C D E F

<data> :: = $\{\{<del>\}_0^\infty <digit>\}_4^4$

<del> :: = <SP> <CR>

Whenever a BREAK is sent to the MAP, the command currently being inputted or executed is cancelled, and the MAP is ready for a new command.

The number of parameters depends on the command (The number of digits in a parameter is fixed for each parameter in the individual commands).

- Output from the MAP.

Whenever the MAP is ready to receive a command, it outputs a '*'
When the MAP detects an error in the input format, it outputs a '?'
When the MAP detects a Time-out, it outputs a 'T'.
When the MAP detects a Parity Error, it outputs a 'P'.
When it is attempted to overwrite locations #970-#977 in the MP-RAM, a " " is outputted.
Some commands produce output parameters, which are transferred to the AV24 port with the following format:

<output params>::= $\{\{<digit>\}_4^4$ <sp> $\{<CR>\}_0^1\}_0^\infty$

2.5.6 PE Initialization

Upon master clear, each CPU will run a self check routine without accessing the P-Bus. Notification reception must be disabled during the self check. After enabling of the Notification reception, the CPU will wait (still without accessing the bus) for a notification, and within 100 us after the first notification fetch its own Notification descriptor (c.f. 2.6) in the MAP.

The MAP will, following its own self check routine, initialize Translation Tables (TT), Translation Table Registers (TTR), and all MAP Processor RAM resident tables.

If in maintenance mode, the MAP is now ready for reception of commands.

If in normal mode, the MAP will notify CPU no. 0 and check, if it fetches its Notification. Descriptor (ND) within 100 us. If not, CPU no. 1 will be notified, etc. After CPU no. 4 , CPU no. 0 is tried again, and so on, until a CPU responds by fetching its ND. The ND will be no. 0000. The system is now operating.

The LED "NO-CPU" will be on during the notification sequence described above, and will be switched off, when a CPU responds.

- Translation Tables (TT)

  Upon Master Clear the address translation tables will map logical page 0,1,2,3 and 63 of Translation Table 1 into the 4K PROM in the MIA, and logical page 0 and 63 of Translation Table 0 will be mapped into physical pages 0 and 1, which must be part of a RAM-module.

- Translation Table Registers (TTR)

  For CPUs, the Program TTR will point out Translation Table 1, and the Data TTR will point out Translation Table 0.

2.6 CR80 Interrupt System

2.6.1 Introduction

Handling of interrupts in the CR80 memory mapped system is, like all CR80 processing, performed as distributed processing, to preserve the CPU power for application oriented work. The interrupt handling described in this section (interrupt preprocessing) covers those interrupts which are not directly related to the CPU, i.e. interrupts where other CR80 modules are involved in the handling. The CPU hardware/firmware reactions (micro program routines) to different types of interrupts are described in the Central Processor section.

The interrupt preprocessing is primarily performed by the MAP module, but partly distributed down to the Peripheral Module (peripheral processor) level.

2.6.2 Vectored Interrupt

The following interrupts belong to the vectored interrupt group (see fig. overleaf):

- Module interrupts issued from peripheral modules and C-Bus DMAs
- CPU interrupts issued from CPU to CPU.
- Power failure interrupts caused by power drop in the PU or CU's.
- Real time clock interrupt, interrupt time is software setable in the MAP module.
- Interrupts from INTRA MEMORY DMA and PE system port.

Each of the above mentioned interrupts has a unique interrupt vector (IV) of 10 bits specifying the interrupt cause as defined in table overleaf (2.6.2-1).
The vectored interrupt preprocessing is performed in the MAP module by means of tables located in its internal memory (MP-RAM) accessible from the CPU's by means of privileged MAP instructions.

The logical operations performed in conjunction with an interrupt are as described below:

Interrupts issued from the different sources, CPU's, peripheral modules, C-Bus DMAs, are received by the MAP's Interrupt Receiver (logical function). The received interrupts which are characterized by their IV are used as pointers into the IV record table which contains information on the current IV status. If the status of the IV record allows the CPU associated with the IV to be interrupted, the MAP will notify the CPU by means of a special set of signals outside the two transfer buses (P-bus and C-bus).

When the CPU recognizes the notification, a context switching in the CPU is performed. During the context switching, which is microprogrammed in the CPU, the IV causing the switch is fetched by a privileged MAP instruction and used as a pointer to the interrupt service routine for that specific IV.

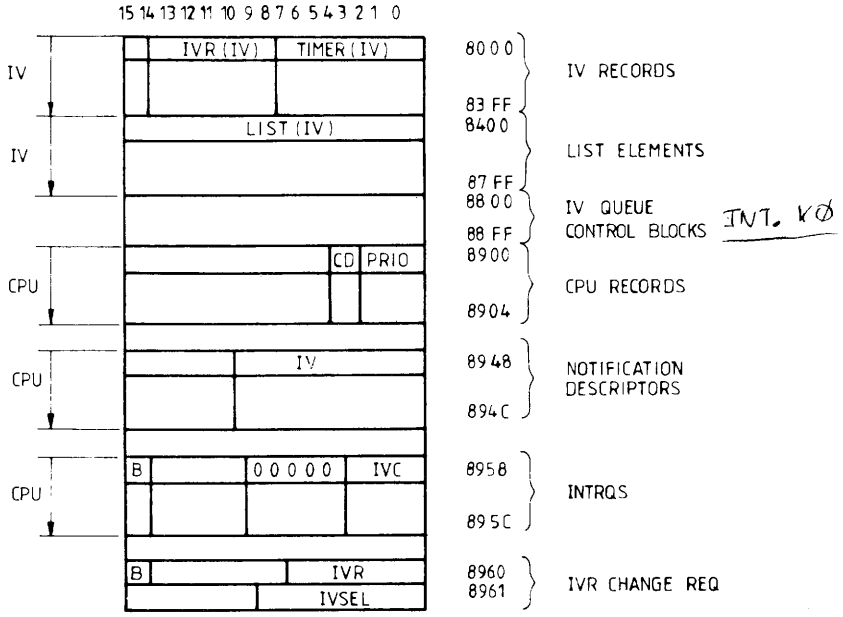
Table 2.6.2-1
Interrupt Vector Table

IV 9 8 7 6 5 4 3 2 1 0	IV (HEX)	Interrupt Source
0 0 0 0 0 0 0 0 0 0		CPU interrupt. (IVC is the vector
0 0 0 0 0 IVC 1 1 1 1 1	0 0 0- 0 1 F	issued from the source CPU, IVC is determined by system software.)
0 0 0 0 1 0 0 0 0 0		C-Bus Module (CBM) interrupt.
0 0 0 0 1 MA 1 0 1 1	0 2 0- 0 3 B	(The module address for CBM's shall be set within limits of MA).
0 0 0 0 1 1 1 1 0 0	0 3 C	AV24 System port in MAP module
0 0 0 0 1 1 1 1 0 1	0 3 D	INTRA MEMORY DMA
0 0 0 0 1 1 1 1 1 0	0 3 E	Real Time Interrupt
0 0 0 0 1 1 1 1 1 1	0 3 F	Power Failure in Processor Unit
0 0 0 1 0 0 0 0 0 0		
CA MA 1 1 1 1 1 1 1 1 1 1	0 4 0- 3 F F	Interrupts from Data Channel modules • CA:Channel Unit number (1-15). • MA(hex)= 3 F: power failure in CU • 00 <MA(hex) <3F available for Peripheral Modules, MA is switch-selectable on the module.

2.6.3 MAP, Internal Interrupt Control Memory

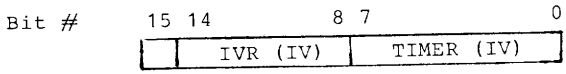
The preprocessing performed in the MAP module is based upon two tables, IV record table and CPU record table, in conjunction with the communication and control area located in the MAP internal memory (MP-RAM). This part of the internal memory is, during normal operation, accessed by means of special privileged interrupt instructions from the CPU's, but could be accessed by the MAP's general instructions, (not allowed during normal operation). The layout of the MAP internal interrupt control memory is shown below.

Figure 2.6.3-1
MAP INTERRUPT CONTROL MEMORY



2.6.3.1 IV Record Table

The IV record table holds a record for each of the 1K vectored interrupts in the Processing Element (PE). The record is formatted as shown below.



- TIMER(IV):

This 8-bit field is used for timing out of the interrupt.

Max. time (0-255) in units of 1024 ms until the interrupt (IV) shall occur.

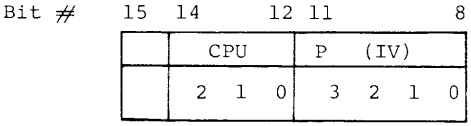
TIMER = 0 means: no time limit

TIMER > 0 means: MAP will decrement TIMER once per 1024 ms.
As TIMER is decremented
from 1 to 0, an IV interrupt is simulated.

Thus:
(TIMER-1) < time limit < TIMER.
When an interrupt (IV) occurs, TIMER(IV) is set to 0 by the MAP.
The TIMER(IV) may be changed by the CPU's without restrictions.

- IVR (IV)

This field defines the priority of the interrupt (IV) and the CPU which may be notified when the interrupt occurs.



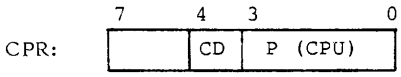
P (IV): IV priority, (0-15), 15 highest priority, bit 0 least significant bit. P (IV) is programmable from the CPU.

CPU (IV): CPU no (0-4, 7) associated with the IV.
CPU (IV) = 0, 1, 2, 3, 4 means that only the CPU no. specified, may be notified by the interrupt corresponding to the IV. CPU (IV) = 7 means that any of the 5 CPU's may be notified.

The IV records table is accessible from the CPUs using privileged MAP interrupt instructions, but can only be changed by the IVR CHANGE REQ instruction.

2.6.3.2 CPU Record

This field defines the CPU's priority, one byte per CPU as defined below:



P (CPU): CPU priority, 0-15:

15 is highest priority, bit 0 least significant. P (CPU) is programmable from the CPU's. When a CPU is notified, the MAP will set P (CPU): = P (IV).

CD: CPU disabled:
1 indicates that notification of the CPU is not allowed.
0 indicates notification allowed.
CD is set by the MAP, when the CPU is notified. The CPU may reset CD when it is ready to accept a new notification.

CPU records may be changed only by special privileged interrupt instructions.

2.6.3.3 IV Queue

The areas: List Elements and IV Queue Control Blocks, are used by the MAP module for storage of interrupts currently not allowed to notify a CPU. When a change in a CPU record is performed, the map will scan through its queues to determine whether any queued interrupt is allowed to notify the CPU.

2.6.3.4 Communication and Control Area

The remaining part of the interrupt control memory is used for communication and control of the MAP-CPU interface as described below:

Notification Descriptors:

This area consists of five words, and the contents correspond to the IV (10 bits) which has caused a CPU notification, one word per CPU. The contents is updated from the MAP when a notification is performed and fetched from the related CPU by its interrupt micro routine.

INTRQ's:

These five words, one per CPU, are used by the map module as interrupt receiver for CPU interrupts.

IVR Change Reg.:

These two words are used to communicate changes to the IV records. The timer part of an IV record (TIMER (IV)) is directly accessible from the CPU's and therefore no communication area in the memory is allocated to this.

2.6.4 Interrupt Preprocessing

2.6.4.1 Vectored Interrupts

The preprocessing performed by the MAP on receiving a vectored interrupt is based upon the contents of the IV records table (see fig. overleaf) and CPU records table as specified below:

Conditions for Notifications

The interrupts received by the MAP module will cause a CPU to be notified by the MAP if:

- An interrupt has been received or stored by the MAP

and
- The CPU disable bit is "0"

and
- The priority of the CPU is lower than the priority of the IV as defined in the IV record

and
- The CPU is assigned to the IV, or CPU (IV) = 7.

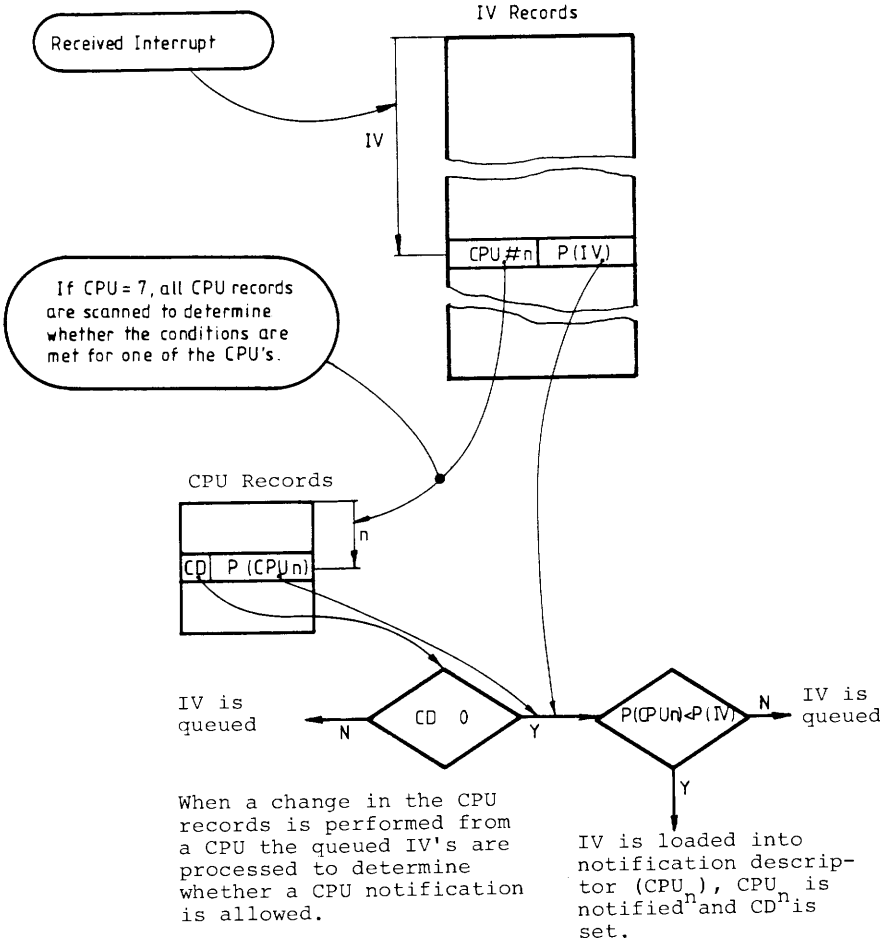
Selection of a CPU to be notified

The MAP selects the CPU with the lowest priority which fulfils the requirements above.
If this rule produces more than a single candidate, one of these is arbitrarily selected.

Queueing of Interrupts

If the current status of the IV record and CPU record does not allow for a CPU notification, the interrupt is queued until a change in the records is performed from the CPU, upon which the queued interrupts are examined to check whether they now are allowed to cause a notification.

Figure 2.6.4.1-1
INTERRUPT PREPROCESSING



If an interrupt (IV) is received while this IV is already in the queue, the new interrupt is thrown away.

The privileged MAP instructions used by the interrupt handler and the CPU's interrupt microprogram routines are as defined in the following.

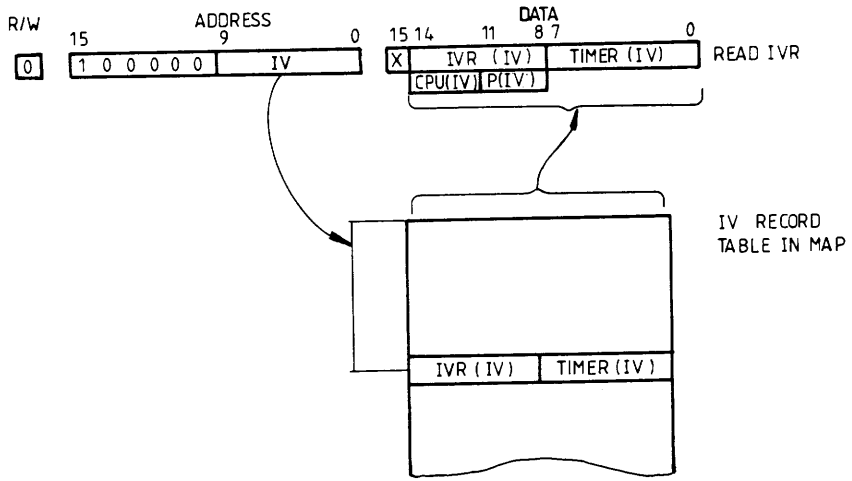
2.6.4.1.1 IV Record Instructions

(Read IVR, Read IVRQ, Write IVRQ, Write IVSEL, Write Interrupt Timeout).

These instructions are used for accessing the IV records table from the CPU's.

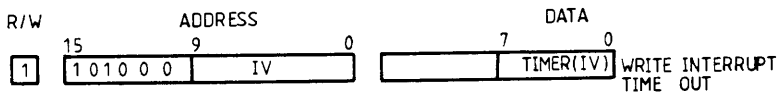
- Read IVR:

By this instruction the contents of the addressed IV is fetched as shown below:



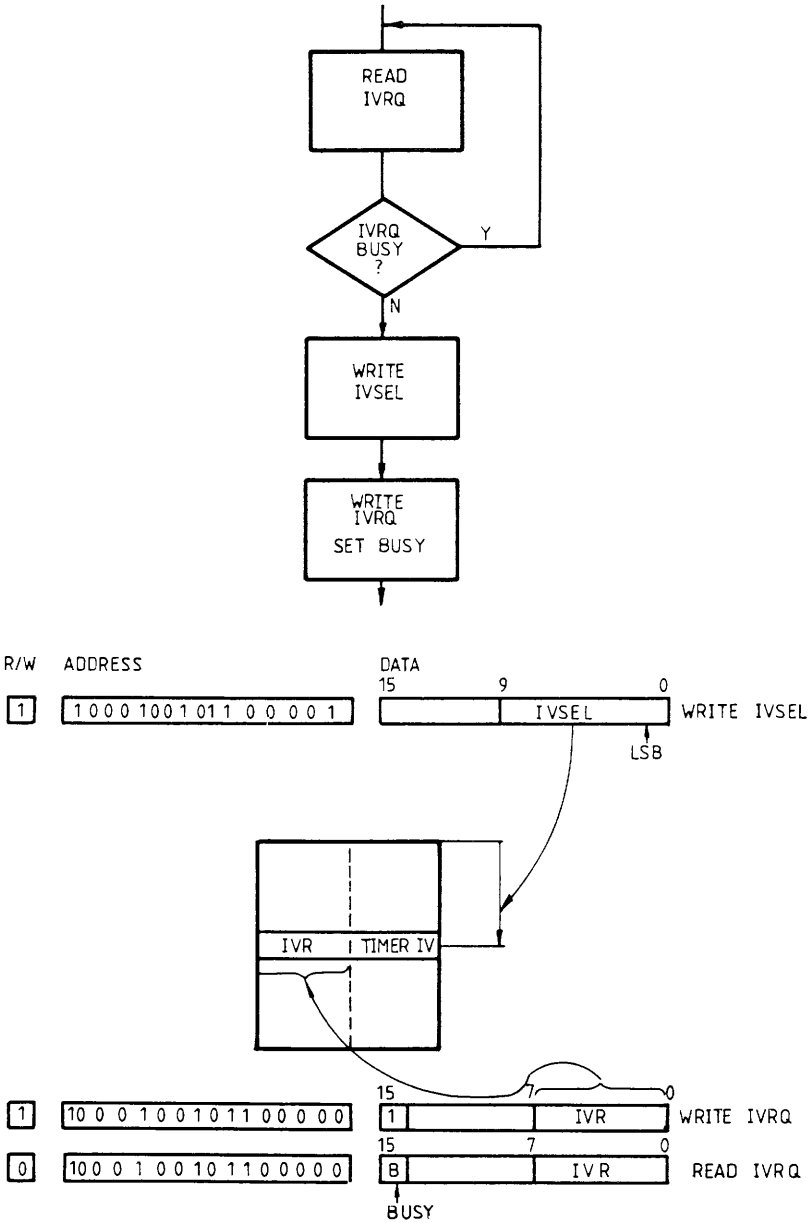
- Write Interrupt Timeout:

This instructions updates the TIMER (IV) of the addressed IV record.



- Read IVRQ, Write IVRQ, Write IVSEL:

  These three instructions are used for updating the IVR field in the addressed IV record as shown below:

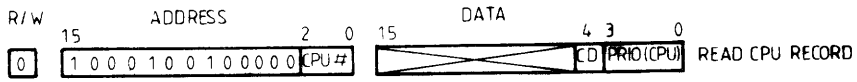


2.6.4.1.2 CPU Record Instructions (Read CPU record, Change CPU record).

These instructions are used for accessing the CPU records from the CPU's.

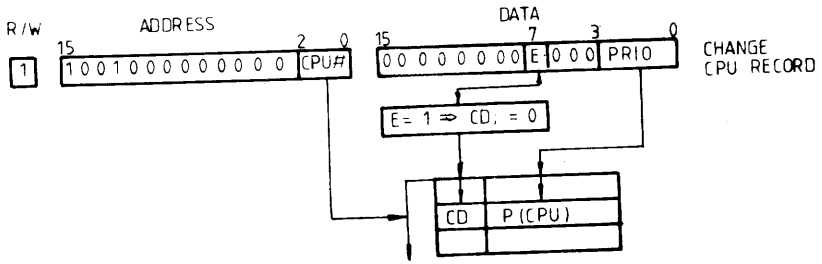
- Read CPU Record:

By this instruction the contents of the addressed CPU record (CPU no.) is fetched as shown below:



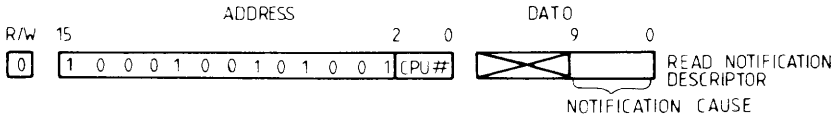
- Change CPU record:

This instruction is used for changing a CPU record as shown below:



2.6.4.1.3 Read Notification Descriptor:

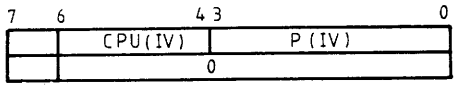
When a CPU has been notified from the map module, the cause (IV) is fetched from the map during the CPU's interrupt micro program routine. The map instruction used is as defined below:
CPU no. in the address field is the CPU notified and is used for addressing the correct notification descriptor.

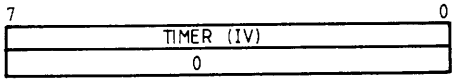


2.6.4.2 Initialization

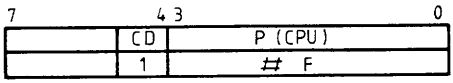
When power is switched ON or the PU is master cleared the contents of the different areas of the Interrupt Control Memory are preset as defined below:

IV Records

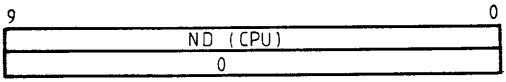




CPU Records

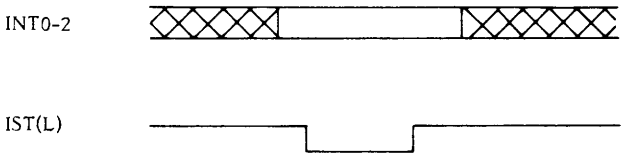


Notification Descriptors



2.6.5 CPU Notification

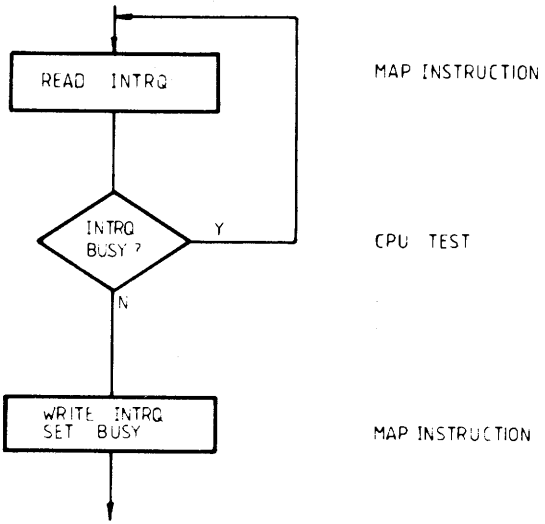
Notification of the CR80 CPU from the MAP module is performed by a set of special signals outside the transfer busses. The signalling allows for notification of a specified CPU as used for vectored interrupts or all CPU's as used with the unvectored fast timer interrupt. Four lines are used, three for addressing and one for strobe as shown below:



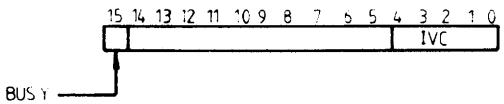
	INT	DESCRIPTION
bit no.	210	
	000	Notify CPU no. 0
	001	Notify CPU no. 1
	010	Notify CPU no. 2
	011	Notify CPU no. 3
	100	Notify CPU no. 4
	101	Not used
	110	Not used
	111	Fast Timer (250 us)

2.6.6 CPU Interrupt

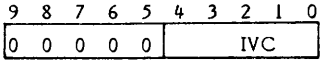
CPU interrupt, which is a standard CR80 instruction (ref. chapter on instruction set) is performed from a CPU by updating its associated "INTRQ" in the MAP module: "Interrupt communication and Control Memory". The update is performed by means of Map instructions as defined below:



The interrupt sent is defined by a five bits IVC in the five least significant bits of the write INTRQ instruction data word.



The resulting interrupt vector is as shown below.



IV for CPU interrupts.

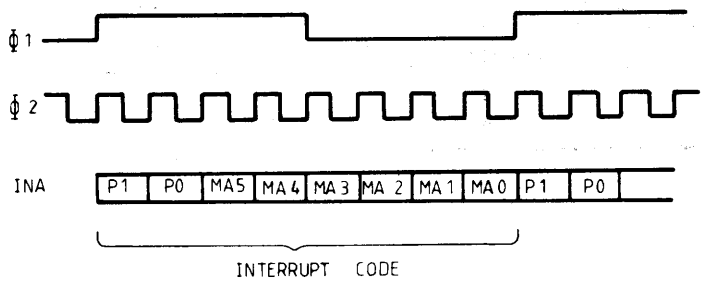
2.6.7 Vectored Module Interrupt

Vectored Module interrupts are issued from the Peripheral Modules and C-Bus DMAs when attention is required. They are transmitted on the CR80 transfer busses to the MAP module interrupt receiver as described below, the processing performed by the MAP module as previously described.

The Vectored Module interrupt paths in the system is shown overleaf (fig. 2.6.7-1), two different schemes are used for transmission of Vectored Module interrupt to the MAP module interrupt receiver.

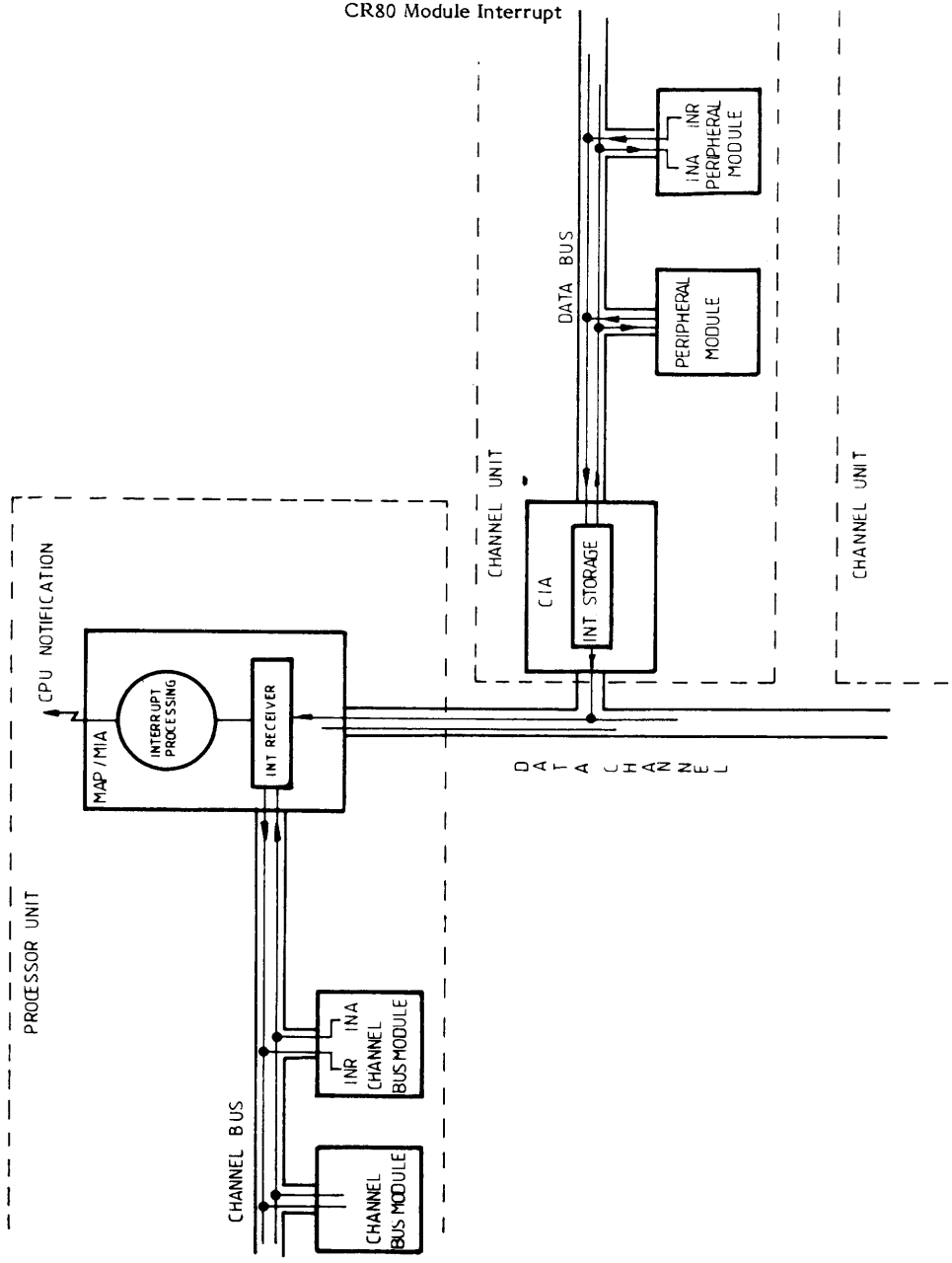
a. Interrupt transmission on C-bus, Data Bus A, or Data bus B.

Interrupt transmission on these buses are performed by serial transmission of the Peripheral Module or C-bus DMA's interrupt code on the open collector line INR. If the code received by the module on the responding INA line (INR inverted) correspond to that transmitted, the interrupt has been accepted by the Interrupt receiving module (MAP in PU, CIA in CU) and transmission is stopped. If the codes do not correspond a module with higher interrupt code has transmitted at the same time, and the module with low interrupt code has to retransmit its interrupt until successful. Transmission of interrupt codes is performed in synchronism with the clock on the transfer buses as shown below.



Vectored Module Interrupt on C-Bus or Data Bus A and B.

Figure 2.6.7-1
CR80 Module Interrupt



The bits in the interrupt code are defined as follows:

P1, P0: Priority bits, switch settable in the module. The priority is only used by the hardware and therefore not part of the interrupt vector (IV).

MA0-MA5: Module address, switch settable in the module. The module address defines an unique module within the CU or PU and is part of the interrupt vector (IV).

Module interrupts issued from C-Bus modules on the C-Bus are received by the MAP interrupt receiver and processed as described in the previous section on Vectored Interrupt, and may result in a CPU notification. The 10-bit Interrupt Vector (IV) for C-Bus modules are composed as follows.

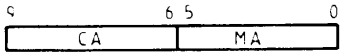
Bit	9			6	5	0
	0	0	0	0	MA	

MA = Module Address: ($20 \leq$ MA (HEX) $\leq 3B$)

The module address (MA) limits defined above for C-Bus modules (DMAs) module address are due to the fact that CPU interrupts and the special MAP interrupts use the most significant four bits as "0". When the MAP module does not want to accept interrupts from the C-Bus it simply pulls the INR line low (corresponding to highest interrupt code), thereby forcing pending interrupts to be queued on the bus until INR is released.

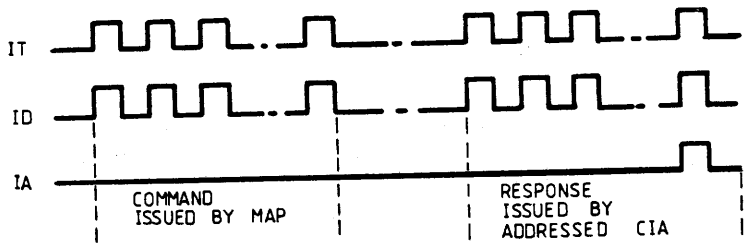
Module interrupt issued on the data buses from Peripheral Modules are received by the CIA's Interrupt Storage. The CIA module will as described for the MAP module above, queue interrupts on the bus until the storage has been emptied by the MAP module via the Data Channel as described below.

b) Interrupt transmission from the CU's on the Data Channel to the MAP module, of interrupt codes from the CIA interrupt storage is performed under control of the MIA module. The interrupt lines on the data channel are controlled from the MIA independently of the data transfer by a polling routine, which sequentially scans the status of the connected CIA's interrupt storage (max. 15 CIA's e.g. 15 CU's). When an interrupt is present in the polled CIA, the interrupt code is transmitted to the map module for processing and generation of CPU notification. The 10 bit interrupt vector (IV) for modules on the Data Channel is composed as follows.

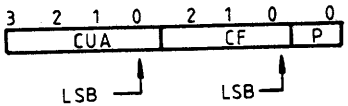


CA: CHANNEL UNIT NUMBER (1 - 15)
MA: MODULE ADDRESS (00 < MA (HEX) < 3F)

The signal sequences on the data channel are shown below.



Command format:



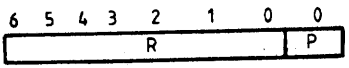
CUA 3:0: Channel Unit Address

CF 2:0: Control Field

P : Parity (odd)

bit	2	1	0	
CF	0	0	0	FETCH INTERRUPT
	0	1	0	FETCH TEST PATTERN
	1	0	0	RESET POWER FAILURE

Response format:



R: Response

P: Parity (odd)

bit	6	5	4	3	2	1	0	
R	1	ADDRESS						Interrupt
	0	0	x	x	x	x	x	No interrupt
	0	1	I	STATUS				Test Pattern

2.6.8 Power Failure Interrupt

Power failure interrupt is issued from the CIA module when its DC voltages (data bus A or B in the CU) drop below 95% of nominal value. The interrupt is handled as a normal Module interrupt. In the PU the MAP module monitors the DC power and generates Module interrupt if the power drops.

The interrupt vector IV for power failure is composed as follows:

IV

bit no.	9	8	7	6	5	4	3	2	1	0
	UA				1	1	1	1	1	1

UA: Unit address

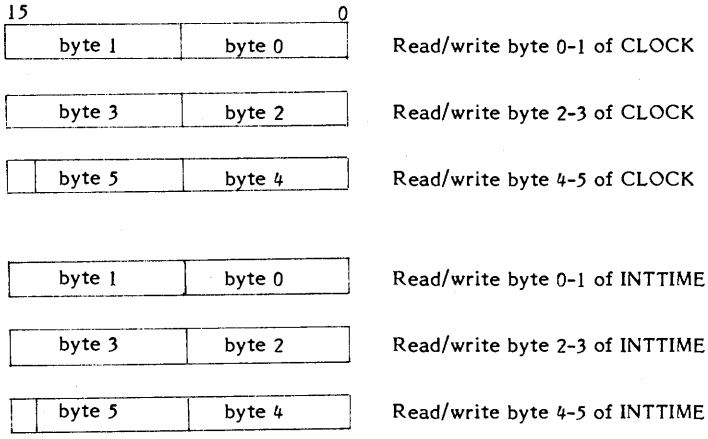
UA=0: PU

UA=1-15 CU address

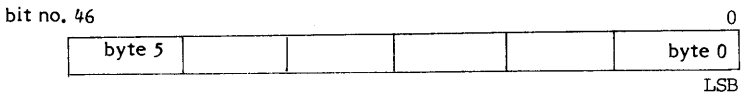
2.6.9 Real Time Interrupt

Real time interrupts are generated by the MAP module whenever its clock counter (47 bits) is incremented to the value of INT TIME register (47 bits). Both clock and INT TIME are accessable from system software by means the MAP instructions defined below.

DATA



CLOCK/INTTIME



The CLOCK counter is incremented each millisecond meaning that the interrupt interval for the real time is: $1ms \leq T_{RT} < 2^{47}-1ms$.

The interrupt vector associated with real time interrupt is defined below:

Bit no.	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	1	1	1	1	1	0

2.6.10 Unvectored Interrupts

This group of interrupt consists of 1) the fast timer interrupt, which is signalled to CPU's on the CPU notification lines to all CPUs in the PU, and 2) interrupts which are signalled to the CPU, currently performing a transfer on the P-Bus, by termination (due to Error) of the transfer from the MAP module without issuing a response signal (RS(L)) to the bus.

- Fast Timer Interrupt

The fast timer interrupt is sent from the MAP module to the CPU's by issuing the CPU notification code 7 together with the strobe.
The interrupt is issued every 250us.

- Termination Interrupts

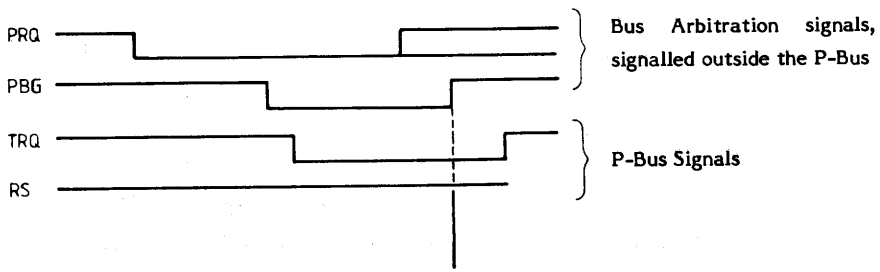
If an error or an unqualified access from a CPU occurs, the MAP module terminates the transfer by terminating the bus grant signal PBG to the CPU while transfer request is still active ref. figure overleaf.
The termination cause will be stored in the MAP module and is fetched from the CPU or from the C-Bus modules by a MAP privileged read operation. The read operation is performed by the CPU's error-handling micro routine.

The termination cause (Access Status Register) is defined overleaf, one register for each CPU and C-Bus module.

Access Status Register:

Bit 1-0:	Access protection bits.	
Bit 2:	Parity error in lower TT-byte.	
Bit 3:	Parity error in upper TT-byte.	
Bit 4:	Reserved	
Bit 5:	Reserved	
Bit 6:	Reserved	
Bit 7:	Timeout detected by MAP	
Bit 8:	Reserved	Error Status from Channel Unit
Bit 9:	Address Register Error	
Bit 10:	Parity Error during Write	
Bit 11:	Parity Error during Read	
Bit 12:	CU-error (bit 11-8 valid)	set by MAP
Bit 13:	Parity error in upper byte from Data Channel	
Bit 14:	Parity error in lower byte from Data Channel	
Bit 15:	Time out on Data Channel	

Generation of Termination Interrupts.



Termination Interrupt is recognized by the CPU if PBG is removed while TRQ is active.

2.7 PE Interface to S-Net

2.7.1 Introduction

Processing elements are interfaced to the Inter Memory Communication network (S-Net) and to TDX buses (or X-Net), by intelligent high speed channel DMA interfaces, accessing the Processing Element memory directly via the C-bus or C-bus and Data Channel in combination.

Figure 2.7.1-1, below shows a PU with S-Net interface, consisting of a SUPRA/TDX Interface controller modules (STI) and from 1 to 8 Adapter modules connecting to the SUPRA buses and/or TDX buses. Up to 4 such interfaces, consisting of a STI module and associated adapters, can be connected to the C-bus of a Processing Element.

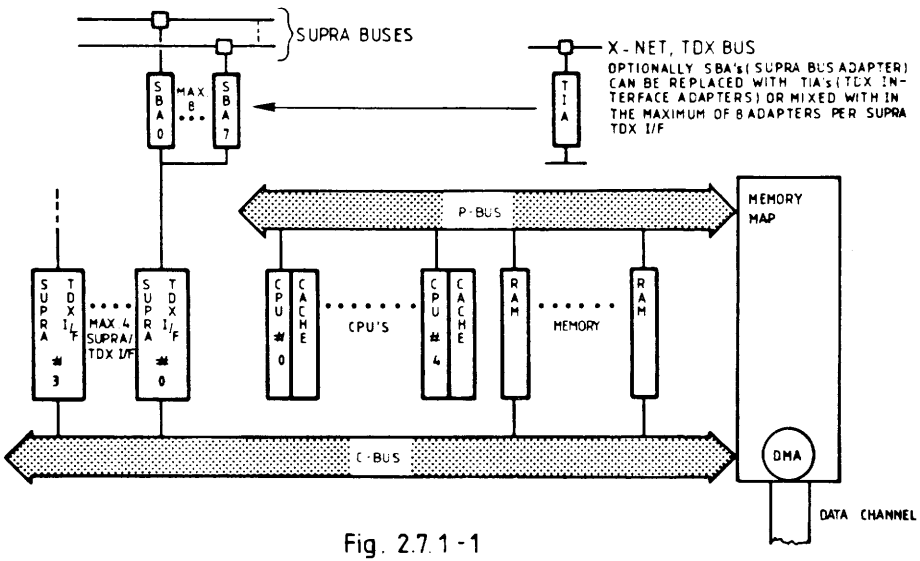


Fig. 2.7.1-1

The function of S-Net will be explained in connection with the SUPRA BUS adapter description in the following, while the TDX (X-Net) is treated in chapter 8 of this handbook.

2.7.2 SUPRA BUS, S-Net Frame Format and the SBA

The S-Net consists of coaxial SUPRA buses interconnecting Processing Elements as required by the application processing and bandwidth requirements. Each PE can connect with up to 15 other PE's. Communication in The CR80 FATOM is by symbolic names, with automatic search for the PE number housing the destination processes or objects, this means that no information of the physical S-Net configuration is needed by application or system processes. SUPRA bus connections need only be established between PE's housing processes and objects actually communicating. Also this provides for search of the new destination PE and back-up processor object, in case of failure of the PE housing the original process or object communicated with.

Further, multiple redundancy is provided by the S-Net, if more than one SUPRA bus is connected between two communicating PE's, by the frame/packet protocol automatically without interrupting PE-processors, reroutes frames via the alternative connections. A SUPRA bus consists of two shielded twisted wire coaxial cable buses of max. length 50 meter, shown overleaf in figure 2.7-1, the one being the DATALINE for serial interchange of data between connected PE's, the other being the RESPONDLINE which establishes flow control by acknowledging to the transmitting PE that the receiving PE is ready to receive data. Transfer rate on both the DATALINE and RESPONDLINE is 16 Megabit/sec. utilizing Alternate Mark Inverted encoding (AMI). The Transceivers are transformer isolated from both coaxial buses. The Transmission frame format (see fig. 2.7.2-2) on the DATALINE follows that of the TDX Bus (reference chapter 8, section 8.2.1.2) for STI/TIA communication with the following exceptions:

- o No FLAG or ABORT bytes
- o No bitstuffing
- o Each byte preceeded by a "one" bit (inserted by transmitter and removed by receiver)
- o One additional header byte (byte 5) extends number of data bytes in frame to 256 bytes.
- o Source PE number is inserted after frame seq.no. in header byte no. 4.
- o Frame length is variable depending on number of data bytes.

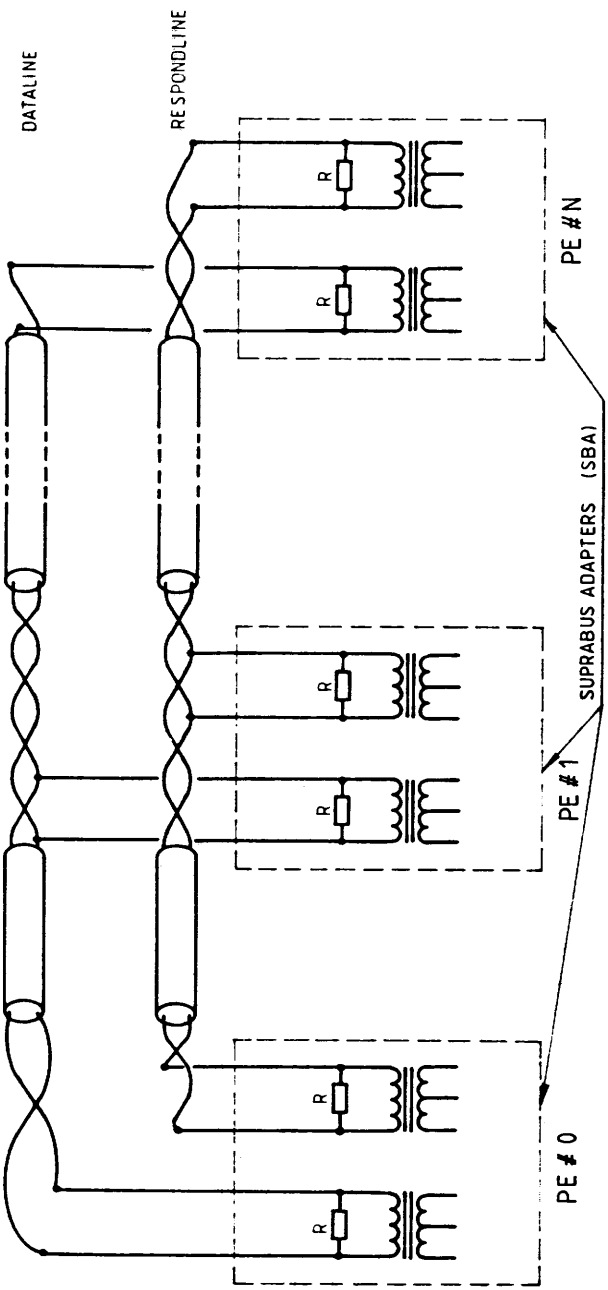
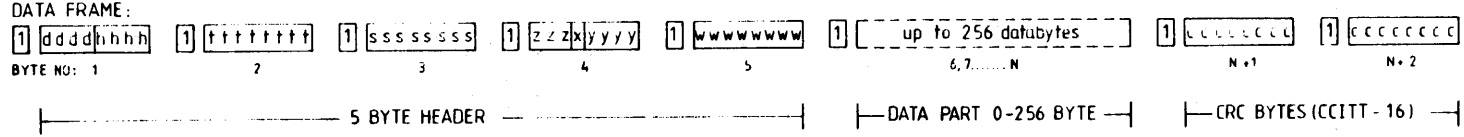


Fig. 2.7.2-1 Suprabus Link

DATABUS:

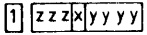
SIGNAL FRAME:

DATA FRAME:



1 : preceeding "one's"
d : datatype (4 bit)
h : destination Processing Element, PE (4 bit)
t : data stream number (6 bit)
s : communication protocol byte (8 bit)
z : cyclic sequence numbering of dataframes (3 bit)
x : dataframe contains data bytes: x = 1 / no data : x = 0 (1 bit)
y : source Processing Element, PE (4 bit)
w : number of following databytes - 1 (8 bit)
e : length of outgoing transfer (16 bit)
c : cyclic redundancy check word (16 bit)

RESPOND BUS:



z,x,y: content as for byte no 4 on the DATA bus

Figure 2.7.2-2
TRANSMISSION FORMAT ON
DATA AND RESPOND BUS

Format on the RESPONDLINE (see fig. 2.7.2-2) consist of a single byte informing the transmitting device that the receiver is ready to receive (receive buffer empty).

Overleaf in figure 2.7.2-3 is shown the block diagram of the Supra Bus Adapter (SBA) of which up to eight can be connected via the shown HI-Bus to a SUPRA/TDX I/F controller (STI). The SBA comprises two transceivers. The one is connected to the DATALINE and to the transmit circuit for parallel to serial conversion from the transmit buffer RAM of the frame to be transmitted, and to the receive circuit for serial to parallel conversion of a frame received to be stored in the receive buffer RAM. The other transceiver is connected to the RESPONDLINE and to the receive circuit for retransmitting the 4th byte of a received frame on the RESPONDLINE, and to a shift register and comparator, for comparing the 4th byte received from the RESPONDLINE with the 4th byte of a previously transmitted frame and latched in the register connected to the transmit circuit. The Transmit and Receive buffer RAM's are connected to the HI-Bus for access by the STI controller. The HI-Bus is further connected to the control/status registers also connected to the receive and transmit state controllers respectively. Assuming that the STI controller being informed that the transmit buffer RAM is empty by testing the associated control/status register, the STI may transfer a frame to the transmit buffer RAM, and shift the status of control/status register to indicate that the transmit buffer RAM is full, thereby initiating the transmit state controller to start arbitrating for the DATALINE. When successful the transmit state controller begins transmission of the frame on the DATALINE at the same time latching the 4th byte in the register. If the receiving SBA within a specified time has returned the 4th byte unchanged, tested by comparison with the latched byte, transmission is continued until all databytes has been transmitted, whereafter a CCITT-16, CRC checkword is appended from the CRC generator. On successful transmission of the frame the control/status register is again set to transmitter buffer RAM empty, ready for a new frame from the STI, otherwise transmission is retried until successful.

When receiving a frame from the DATALINE the SBA receive state controller checks in the associated control/status register if receive

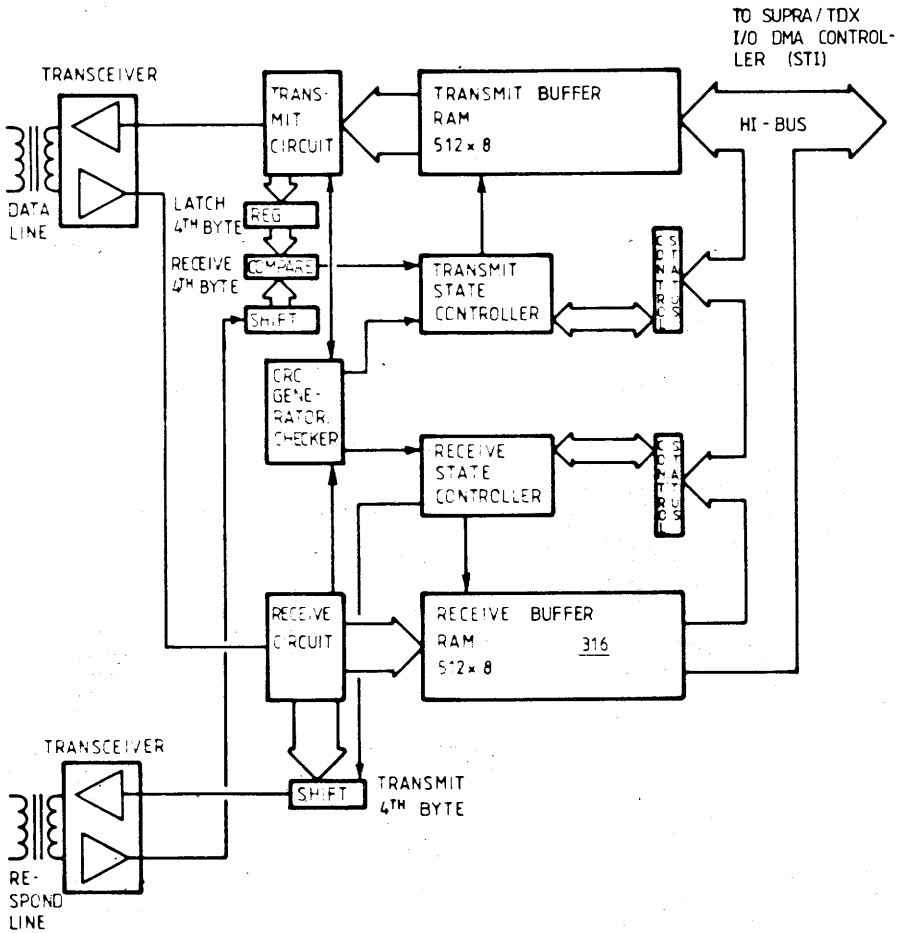


Fig. 2.7.2-3 Suprabus Adapter (SBA)

buffer RAM empty, if not the frame is ignored. If empty the received data is stored in the receive buffer RAM and the 4th byte received is stored in the shift register and retransmitted on the RESPONDLINE. When the complete frame has been received and if the received CRC checkword is correct the control/status register is set to receive buffer RAM full to indicate to the STI that a receive frame is available.

Each SBA (TIA) is seen by the STI as 1 Kilobyte RAM with the maximum of eight SBA's occupying 8K byte of address space on the HI-Bus.

The STI continuously scans the attached SBA's for empty receive buffer RAM's and full receive buffer RAM's, in order to transmit and receive frames. As explained above, a frame placed in a SBA transmit buffer RAM will keep the control/status register in the full indication until transmitted, this automatically forces the STI to distribute it's traffic to other SBA's if traffic on a SUPRA BUS connected to by a SBA is high. It also in the case of SUPRA BUS failure keeps the transmit buffer RAM full indication continuously on in the control/status register of the connected SBA's, thereby forcing retransmissions, automatically requested by the packet protocol timing out, as well as new traffic to alternative SUPRA BUS connections. These mechanisms provides for the unique automatic load distribution and multiple redundancy of the S-Net intermemory communication network.

2.7.3 STI, SUPRA/TDX Interface Controller

The STI controller shown in figure 2.7.3-1 overleaf, is centered around two internal buses the HI-BUS and the CI-BUS. The HI-BUS connects the two micro computers of the STI controller with the STI control memory, SBA and TIA adapters, and the CI-BUS, arbitration of the HI-BUS between the address sources being performed by the HI-BUS arbitrator.

The CI-BUS interconnects ingoing DMA, outgoing DMA, HI-BUS, C-BUS, Status Register (SR) and the Interrupt Controller (IC), arbitration of the CI-BUS between the address sources being performed by the CI-BUS arbitrator. The Ingoing processor comprising CPU, program storage in PROM and RAM as working memory is a self-contained microcomputer isolated from the HI-BUS when processing internally by the busseparator, and via the HI-BUS having access to receive buffers of the SBA's and TIA's and the STI control memory, and via the bus separator and the CI-BUS to Ingoing DMA, the status register (SR) and the interrupt controller (IC), and further via the C-Bus I/F and the C-Bus access to MAP translation memory of the Processing Element (PE) to which the STI controller is attached. The task of the ingoing processor is to scan the receive buffers of attached SBA's and TIA's and on finding one having received a frame transferred into it from a SUPRA BUS (SBA) or TDX BUS (TIA), to analyze the header and based on control information found in the STI control memory to set up the ingoing DMA to transfer the data part of the frame in the receive buffer into the specified area in memory of the Processing Element (PE). Further the ingoing processor also handles S-Net signal frames, received from the S-Net in receive buffer of the SUPRA BUS Adapters (SBA). An S-Net signal frame indicates that the Processing Element sending it over the S-Net wants to communicate with the Processing Element via a yet not specified datastream, the ingoing processor stores the number of the received requested datastream in the status register (SR) and interrupts the PE-CPU's via the interrupt controller (IC) for allocation of a memory area for the communication and specification in STI control memory of the datastream by a stream descriptor, without which the ingoing processor will ignore incoming data on that datastream from the S-NET.

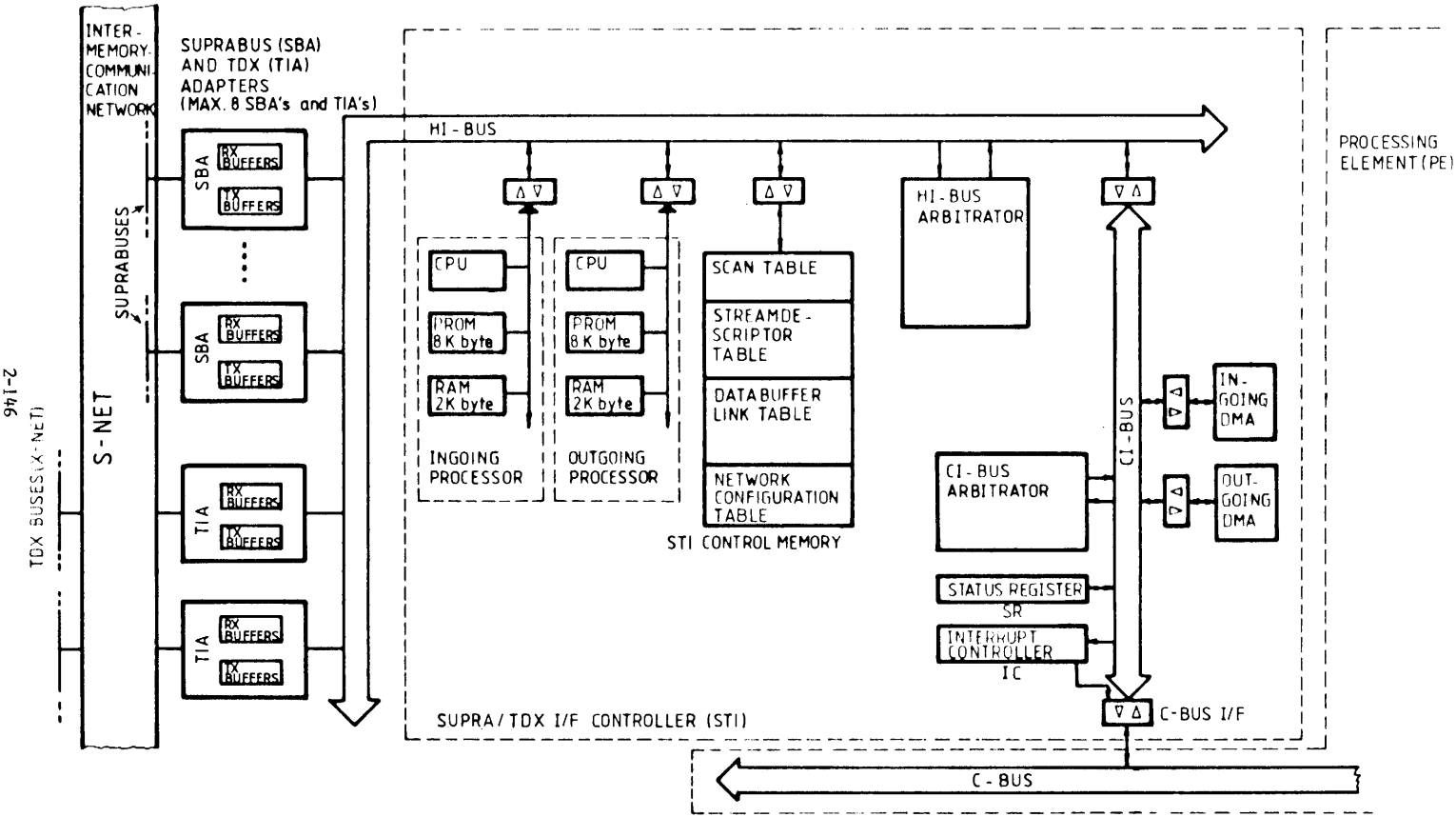


Fig. 2.7.3-1 STI, SUPRA / TDX I/F CONTROLLER

With reference to fig. 2.7.3-1 the outgoing processor comprising CPU, program storage in PROM and RAM as working memory, it is as the ingoing processor a self-contained microcomputer isolated from the HI-BUS by bus isolator when processing internally, and via the HI-BUS having access to transmit buffers of the SBA's and TIA's, the STI control memory, and via the bus separator and the CI-BUS to outgoing DMA, status register (SR) Interrupt Controller (IC) and further via the C-Bus I/F and the C-Bus access to the MAP translation memory of the Processing Element. The task of the outgoing processor is to handle outputting of frames on active datastreams from the output memory areas in the Processing Element to empty transmit buffers in the SBA's or TIA's, by analyzing a corresponding stream descriptor in the STI Control Memory specifying the datastream. The outgoing processor sets up outgoing DMA to transfer the datapart of a frame to be communicated to the S-Net or TDX buses, from memory of the Processing Element to the transmit buffer of a SBA or TIA connected by its associated SUPRA BUS or TDX bus, with the destination Processing Element or TDX device as specified in the network configuration memory (included in the STI control memory), also based on the information in the stream descriptor the outgoing processor directly loads the header into transmit buffer of the selected SBA or TIA.

2.7.3.1 STI Handling of S-Net Packet Protocol

The communication protocol ensuring error free communication on a datastream between Processing Elements via the S-Net or a channel to a TDX device on a TDX bus is jointly executed by the ingoing processor and the outgoing processor. The basic packet protocol used with the S-Net and TDX bus is identical, except for the lay-out of frames, the following is an overview of the protocol with relation to the S-Net, while more detail as well as information specific to the TDX bus can be found in the later chapter 8, on the TDX system (section 8.3.2 and 8.3.3). The STI processors cooperating with the corresponding ingoing processor and outgoing processor at the other end of the datastream via the S-Net, with variables relating to the communication protocol processing being stored in the intermediate result area of the stream descriptor, specifying the datastream at each end.

Data is transferred over the S-Net in packets consisting of one or more data frames. To transfer a packet over the intermemory network, S-Net, without error on a datastream being specified by a stream descriptor in both SUPRA/TDX Interface Controllers of two Processing Elements connected via the S-Net; packet control and status information is sent in both directions contained in the communication protocol byte 3, of data frames (figure 2.7.2-2) being transferred in both directions on the specified datastream. Each specified datastream is controlled by separate communication protocol independently of all other datastreams, the following refers to a single datastream.

With reference to figure 2.7.3.1-1 overleaf, for each packet there is an outputter 1 at the end of the datastream that originally had the packet and an inputter 1 at the other end that finally gets the packet. These terms must be distinguished from "sender" and "receiver" which are used in the conventional way to distinguish the two ends of a single transmission, for a communication protocol byte (control byte, see. chapter 8, figure 8.3.2-2) sent in one direction may contain information about a packet output in the other direction.

Because the datastream is full duplex (but not necessarily the same speed in each direction) for output and input as well as for sending and receiving, two packets may be transmitted independently and simultaneously in opposite directions over the datastream.

A packet consists of one or more data frames, the data frames within a packet is contiguously numbered modulo 8, transmitted in byte 4 of the frame header, starting with zero for the first data frame in the packet. The first and the last data frame in the packet contains a communication protocol byte which indicates the begin and end of the packet, as well as the output phase (0 or 1) of the packet.

With reference to figure 2.7.3.1-1 overleaf, completeness of a received packet is ensured by the inputter 1through contiguous numbered (modulo 8) datablocks between the first and the last datablock in the packet.

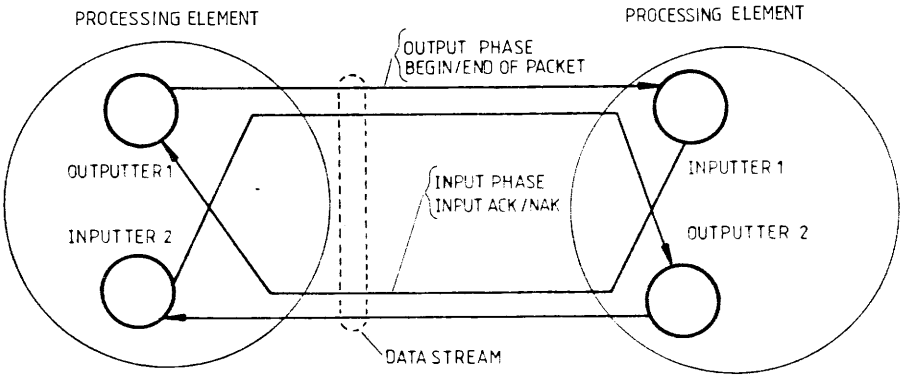


Fig. 2.7.3.1-1 Simplified Protocol-Control Flow Through a Datastream (Flow Control not Shown)

All errors will result in dataframes being rejected at the receiving station, errors are therefore detected by the protocol by the following dataframe arriving out of sequence (or time out by a receive timer, if last frame in packet), the complete packet is rejected immediately (without waiting for completion of packet) and retransmission requested by inputter 1, by replying Not AcKnowledged, NAK in the communication protocol byte (see fig. 8.3.2-2) of a dataframe transmitted in the opposite direction on the datastream. The packet phase indicated in the first and last frame of a packet is also checked, if it is n+1 (modulo 2), where n was the last packet accepted it is accepted via ACKnowledge, ACK by inputter 1 in the communication protocol byte of a dataframe in the opposite direction on the datastream. If it is n, it is accepted via ACK, but it is thrown away by the inputter 1 since it must have been duplicate transmission caused by an error in one of the acknowledgements.

ACK/NAK is transmitted by the inputter in the communication protocol byte. The communication protocol byte is transmitted either contained in a dataframe in a packet going from the inputter (any dataframe: first, last or in the middle of packet) or initiates sending of a dataframe, outside packets, containing no data and which is discarded by the outputter 1 except for the content of the communication protocol byte, this allows for immediate ACK/NAK response by the inputter 1 to a packet received from the outputter 1.

At the Outputter 1 each acknowledgement is also checked for errors, if erroneous it is thrown away, if error free and ACK, then packet n+1 is transmitted, and if NAK then packet n is retransmitted. A timer at the transmitter initiates retransmission of packet n in case that neither ACK nor NAK is received within a specified time (e.g. lost due to error of the link), retransmission is attempted 3 times before the protocol gives up on output.

The transfer of a packet in the opposite direction of the datastream is performed identically as described above replacing outputter 1 with outputter 2 and inputter 1 with inputter 2.

The outgoing processor (see figure 2.7.3-1) at each end of the datastream (figure 2.7.3.1-1) executing the part of the communication protocol related to the outputters respectively and ingoing processor (see figure 2.7.3-1) at each end executing the part related to the two inputters respectively.

2.7.3.2 STI and S-Net Datastreams

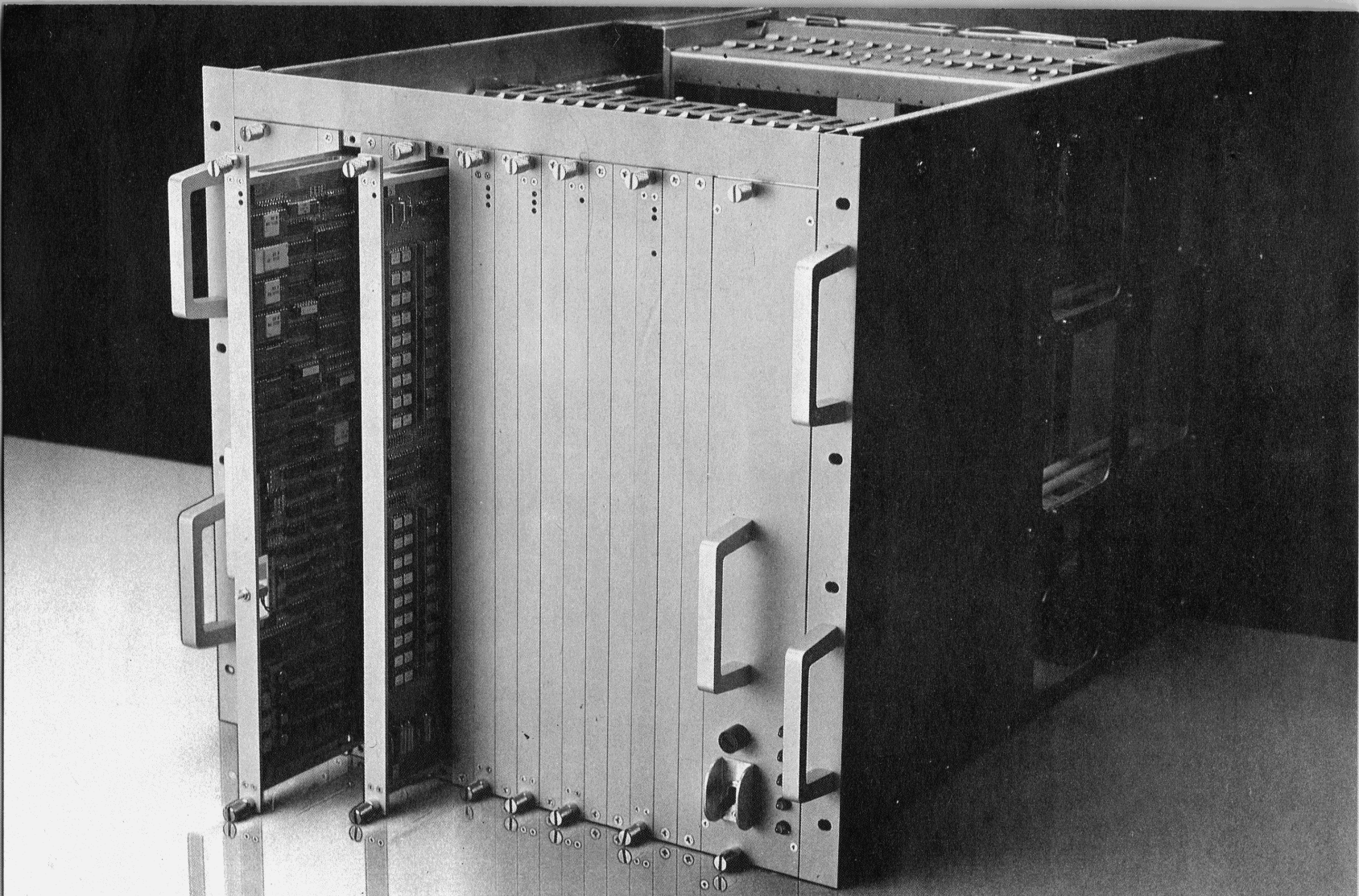
This section gives a general description of the basic functions related to datastreams between Processor Elements via the S-Net. Reference is made to Chapter 8, section 8.2.1.3.1, 8.2.2 and 8.2.2.2 for description of datastream separation, segmenting of LDU (Logical Data Units) into packets and dataflow in the SUPRA/TDX I/F controller which is common for the S-Net and the TDX bus (replacing channel descriptors with stream descriptors). Communication over the S-Net between two Processing Elements is done by specifying a datastream between the memories of the Processing Elements, a datastream when specified will transfer data content of an area in memory of a Processing Element to an area in memory of a second Processing Element, and in the opposite direction from a memory area in the second Processing Element to a memory area in the first Processing Element. Each of the up to 4 STI controllers in a Processing Element provides for up to 256 datastreams being specified to each of the up to 15 Processing Elements communicated with via S-Net, giving a total of up to 16128 possible independent datastreams being specified (by stream descriptors), from a Processing Element via S-Net to other Processing Elements. The possible datastreams are identified by the 16 bit CR-ID (figure 8.3.2-1, byte 1 and 2), where 8 bit identifies each of the two communicating Processing Elements by a 4 bit number, and the last 8 bit indicates the one of 256 possible streams between these used. Frames transferred between the S-Net and the Processing Element is multiplexed between the specified datastreams by the STI controllers with a data transfer capacity of up to 400K byte/sec (TDX: 400 K bit/sec) of each STI. It is an important feature that the large number of possible datastreams, transfers directly between data areas in memory of software processes in different Processing Elements with a need to communicate with each other, instead of having only a few fixed memory areas allocated, and thereafter having to multiplex the use of these few

areas between software processes needing to communicate, this multiplexing being done by software and thereby putting a heavy load on the CPU's. Also it is an important feature that a datastream can be specified directly between a compartmentalized memory accessible by a peripheral processor in one Processing Element and a similar second memory in another Processing Element accessible by another peripheral processor, and that the datastream can be specified to continuously transfer the content of a buffer in the first memory to a buffer in the second memory and vice versa, each time transferring the content of the buffer without interruption of the CPU's, this allows for direct communication between peripheral processors via the S-Net without overhead on the CPU's after having initially specified the datastream.

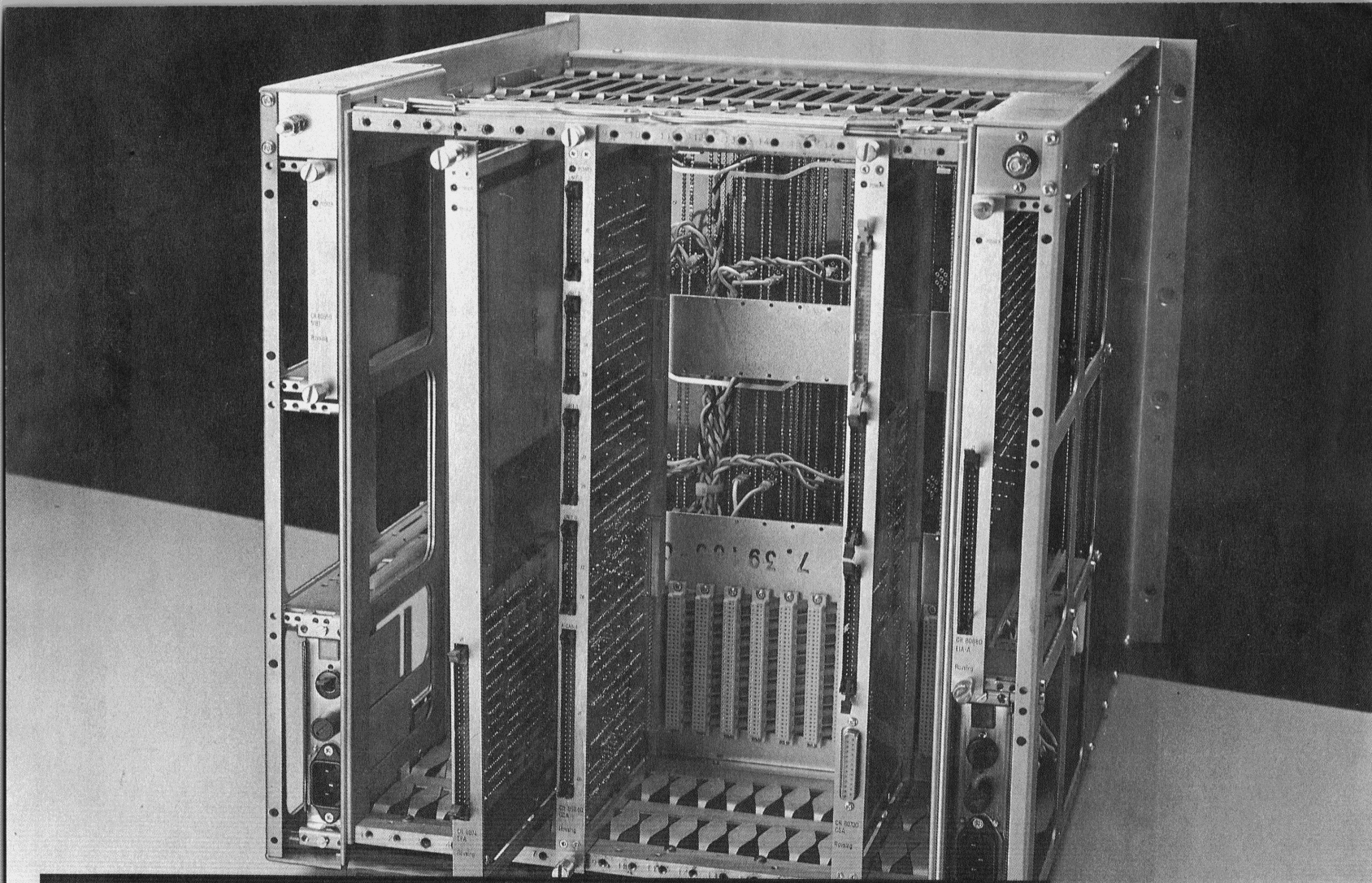
A datastream between two separate Processing Elements is specified by each Processing Element entering a stream descriptor in the control memory of one of their connected SUPRA/TDX I/F controllers, the stream descriptor defines the memory areas and control information to be used with that datastream in the Processing Element. Datastreams are not initially specified between Processing Elements but dynamically generated and specified, by a Processing Element initially entering a stream descriptor in the control memory of one of its connected SUPRA/-TDX I/F controllers specifying the other Processing Element to be communicated with, datastream number, and amount of data to be transferred, but no memory area for input and output of data. This generates a request signal frame (figure 2.7.2-2) containing data of the tentatively specified datastream to be sent via the S-Net to the specified Processing Element. If the Processing Element receiving the signal frame accepts to communicate with the originating Processing element on the indicated datastream number, it enters a corresponding stream descriptor in the control memory of one of its connected SUPRA/TDX I/F controllers, but now pointing to memory areas for input and output of data, the stream descriptor will initially send a signal frame as above in the opposite direction with amount of data to be transferred in that direction and afterwards the stream descriptor will be ready for normal data-transfer. The originally initiating Processing Element, in response to the received signal frame enters a stream descriptor defining memory areas for input and output in one of its SUPRA/TDX I/F Controllers. The

datastream is now said to be specified and transfer of data between the specified areas in memory of the two Processing Elements via the S-Net is autonomeously handled by the associated SUPRA/TDX I/F controllers until the data has been completely transferred, whereafter the SUPRA/-TDX I/F controllers interrupt their associated Processing Element CPU's for closing the datastream by deleting the corresponding stream descriptors in their control memory.

NOTES:



The CR80 M-Crate seen from the front. The outermost positions in each side occupied by the power supplies and 17 positions in between available for plug-in modules.

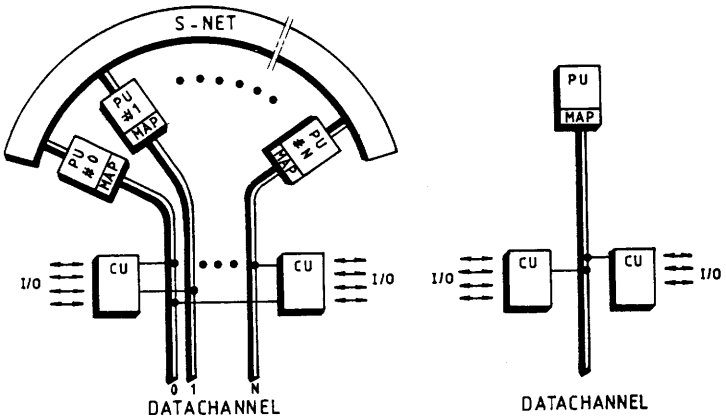


The CR80 M-Crate seen from the rear. The outermost positions are utilized for bus termination or datachannel attachment boards. The Adapter crate is placed in the center, and has 19 positions available for Adapter modules, interfacing to external and peripheral devices.

3. CR80 MAPPED SYSTEMS, PU AND CU CRATE ELECTRICAL, MECHANICAL AND BUS I/F

3.1 Introduction

This chapter discusses the General Mechanical and Electrical interfaces and buses of the Processor Unit (PU) and Channel Unit (CU) crate subsystems in CR80 Memory Mapped Computer Systems (CR80 FATOM and MAXIM), but is to a large extent also applicable for the non-mapped Computer Systems (CR80 MINI and TWIN).



CR80 FATOM CR80 MAXIM

Each unit (PU or CU) is a mechanical and electrical entity, normally housed in a standard CR80, 25-slot Crate (Card Cage), as shown (fig. 3.1-1) overleaf. Each PU or CU Crate has its own power supply (supplies), cooling system, and is galvanically isolated from all external connections:

- S-NET
- X-NET (TDX-Bus)
- DATACHANNEL
- CONFIGURATION BUS
- MAINS LINES

CR80M PROCESSOR UNIT & CHANNEL UNIT

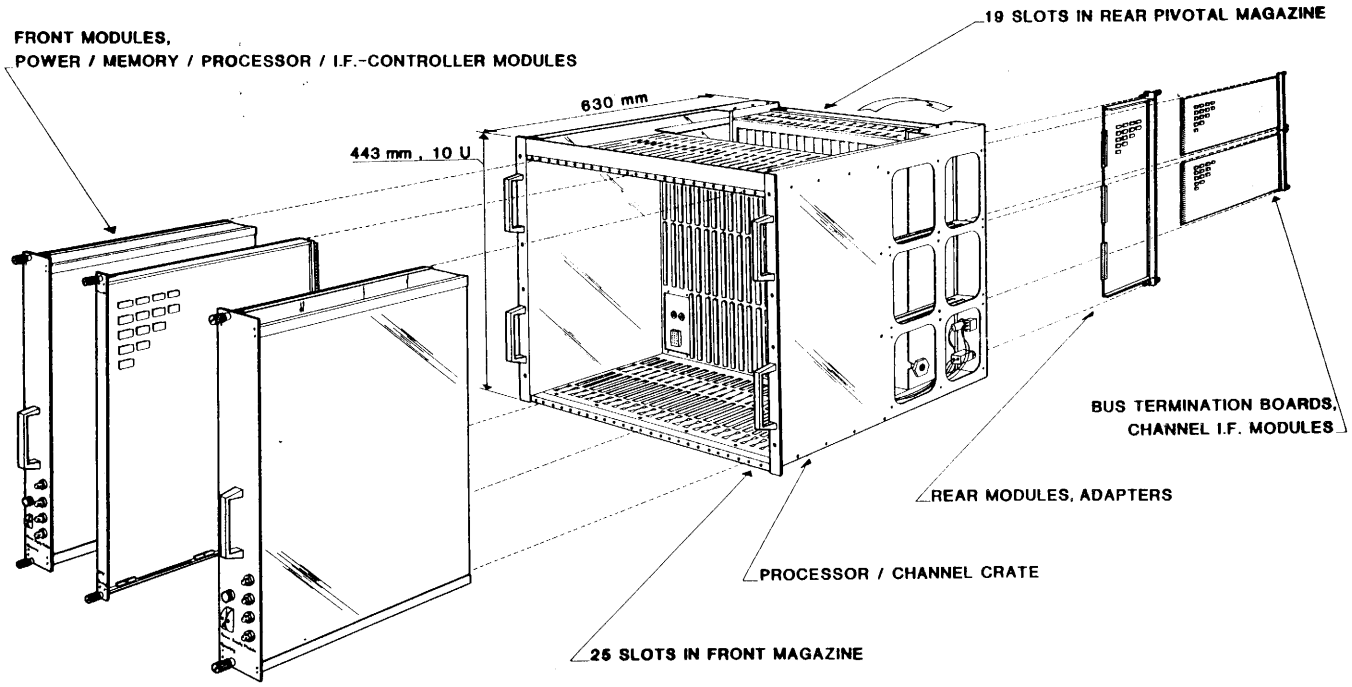


Figure 3.1.1.

The chapter is organized in the following order for top-down description and easy reference:

- PU (Processing Unit), describing the physical and electrical interfaces of the PU-crate, as well as interconnections between modules inserted into the PU-crate (except for the general databuses: P-bus and C-bus treated separately later in the chapter).

- CU (Channel Unit), describing interfaces and interconnections of CU-crate as above (general databuses : A-bus and B-bus treated separately later in the chapter).

- General Data Busses of the PU and CU crates, describing the physical and electrical interfaces of:

  - P-bus in PU crate
  - C-bus in PU crate
  - A-bus (Databus A) in CU crate
  - B-bus (Databus B) in CU crate

  The emphasis being on the P-bus, since the C, A and B-buses except for a a few exceptions are compatible with the P-bus.

- Interconnecting buses between PU's and CU's
  - Data Channel (between PU and CU's)
  - Suprabus S-NET (between PU's)
  - X-NET (TDX-BUS) and configuration bus treated in separate chapters of the handbook.

- Finally, positioning of PU and CU crates into RACKS's, FAN units and Power distribution is discussed.

3.2 CR80 PU (Processor Unit)

Introduction

The CR80 PU, is an electrically and physically self-contained unit, accomodating all address sourcing devices of a PE. The PU contains up to 5 CPU's, DMA modules, 1 megaword of memory, Memory Map, Power Supply (supplies) and Datachannel interface to Peripheral Modules and additional memory (up to 15 megawords). It is capable of connecting into large multicomputer configurations, with up to 15 other PE's via fast suprabuses and to terminal networks via TDX-bus.

Please refer to drawing overleaf (fig. 3.2-1) for functional and mechanical lay-out of the PU.

Physically, the CR80 modules constituting the PU are housed in a 19" crate (card-cage) of height 443 mm (10U), and depth 630 mm.

The PU crate supports insertion of the following CR80 standard modules in its 25 front and 19 rear positions:

Front positions (25):

- up to 2 Power Supply modules (1 each side)
- MAP
- CPU modules
- RAM modules
- STI modules (Supra /TDX interface).
- CO-Processors

Rear Adapter cage positions (19):

- MIA (MAP I/F Adapter)
- SBA's (SupraBus Adapters)
- TIA's (TDX I/F Adapters)
- CO-Processor Adapters
- CCA (Configuration Control Adapter)

Rear side positions:

- Bus Termination Modules, MBT (P-bus and C-bus)

As can be seen from the CR80 datasheets combination crates also exists, which pack 2 small or medium size PU's into one 19" Crate, or combine a PU and CU in one crate. In addition a smaller (15 front and 7 rear position) MINICRATE exists. Since these are interface- and bus-compatible with the large PU-crate, only the latter is treated in the following.

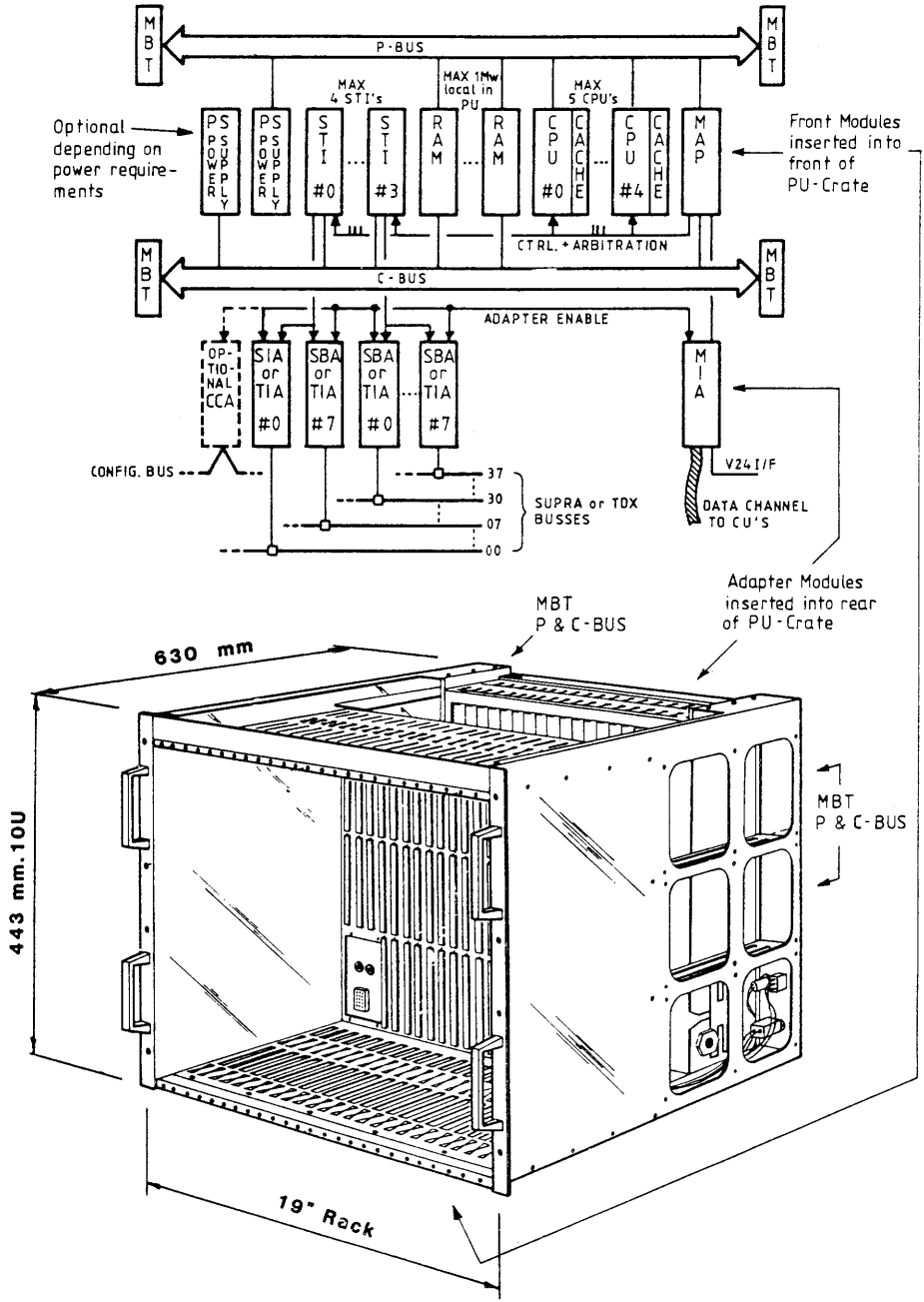


Fig. 3.2-1 PU (Processor Unit), Functional & Mechanical Layout.

The functional block diagrams of the PU correspond physically with the busses, backpanels and interconnections in the PU-crate as follows:

Front Magazine:

- Top (J1) — P-Bus
- Middle (J2) — C-Bus
- Bottom (J3) — Module interface connectors with Backpanel for connecting control and arbitration signals between modules 64 connector pins free in each position for individual flatcable connection from front module to corresponding Adapters in rear magazine (exception: Special Power Supply connectors in each side).

Rear Adapter Magazine

- Top (J3) — Backpanel for distribution of power and enable signal to Adapters.
- Middle (J2) — Normally not used.
- Bottom (J1) — Free for 64pin flatcable connection between individual Adapters and corresponding Front Module.

Rear Side positions:

- Top: Insertion of Bus Termination Modules, MBT, for the P-Bus (one each side of crate).
- Middle: Insertion of Bus Termination Modules, BTM, for the C-Bus (one each side of crate).
- Bottom: Mains plugs, one each side, for corresponding, plug-in Power supplies in Front Magazine.

Since the P-Bus, C-bus Datachannel and Suprabus are separately treated later in the chapter, this section concentrates on the other Aspects of the Processor Unit (PU):

- Physical and Electrical Specifications of PU-Crate
- Physical dimensions and specifications of:
  - Front Modules (Front Magazine)
  - Adapter Modules (Rear Magazine)
  - BTM modules and Mains Connections (rear side positions).
- Front Module Interface Connector position:
  - control and arbitration backpanel
  - MAP
  - CPU's (arbitrating for P-Bus)
  - STI's and other DMA modules arbitrating for the C-Bus.
- Processor Unit External I/F's:
  - V24 I/F of MIA module (Map Interface Adapter)

3.2.1 PU-Crate

The CR80 processing modules are housed in a standard 19" Module Magazine, the Processing Unit Crate (PU-Crate). Different types of crates are available from the CR80 product line as defined in the data sheets. The data sheets give the specifications concerning module positions within the PU-crate and the number of module slots available, while the general interface specifications for the PU-crate are defined in this section.

The PU-crate consists basically of a front magazine for the front modules (processing modules) and a rear magazine for rear modules (adapter modules) placed back to back as shown overleaf (figure 3.2.1-1).

The interface between the front modules and the rear adapter modules is performed by means of 64p flat cable, while the interface between the front module is established by means of multilayer printed circuits boards, two identical for the two data transfer buses P-bus and C-bus and a third in the module interface connector position as shown overleaf (figure 3.2.1-1).

The Adapter module interfaces the PU external connections on the Adapter module panel and by means of 64p flat cable to the front module. Power and configuration control signals to the Adapter Modules are distributed via printed circuit board J3 backplane in upper connector position of the rear crate.

Connector specifications for Front and Adapter backplanes are given in the following figures 3.2.1-2 and 3.2.1-3. Also shown in figure 3.2.1-2 is the flat cable connection between the backplane connectors of a front module (J3) and its corresponding Adapter module (J1). The actual module locations, within the various standard PU's where flat cable connections are implemented, are defined in the PU datasheets. The pin layout and signals of the various PU-Front Modules backplane interface connector (J3) is discussed in detail in section 3.5.3.

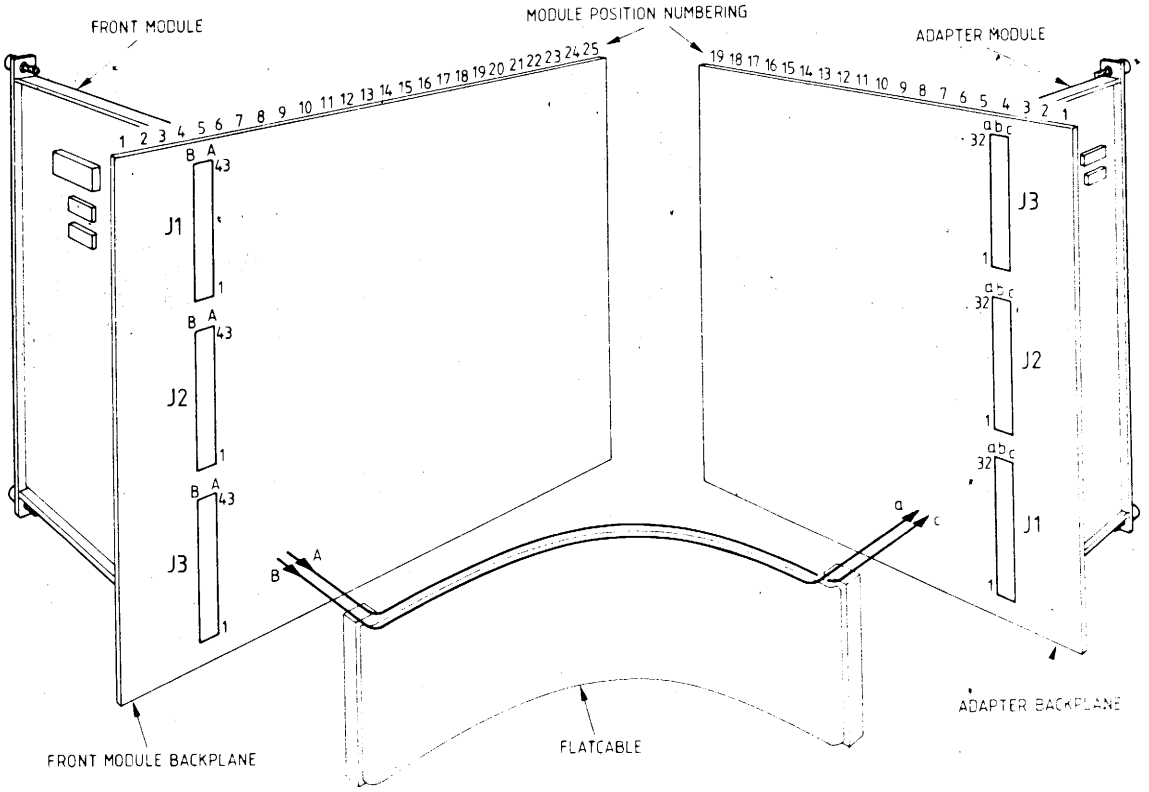
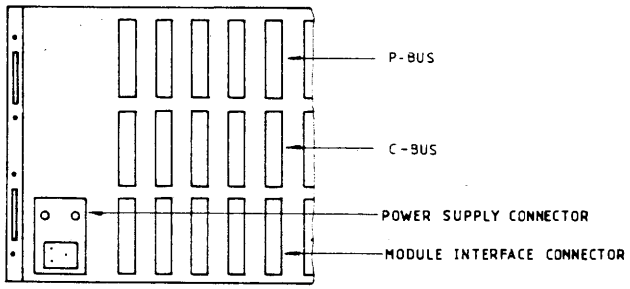


Fig. 3.2.1 - 1 Connector positioning and Pin numbering on Front Module Backplane and Adapter Backplane

FRONT VIEW



SIDE VIEW

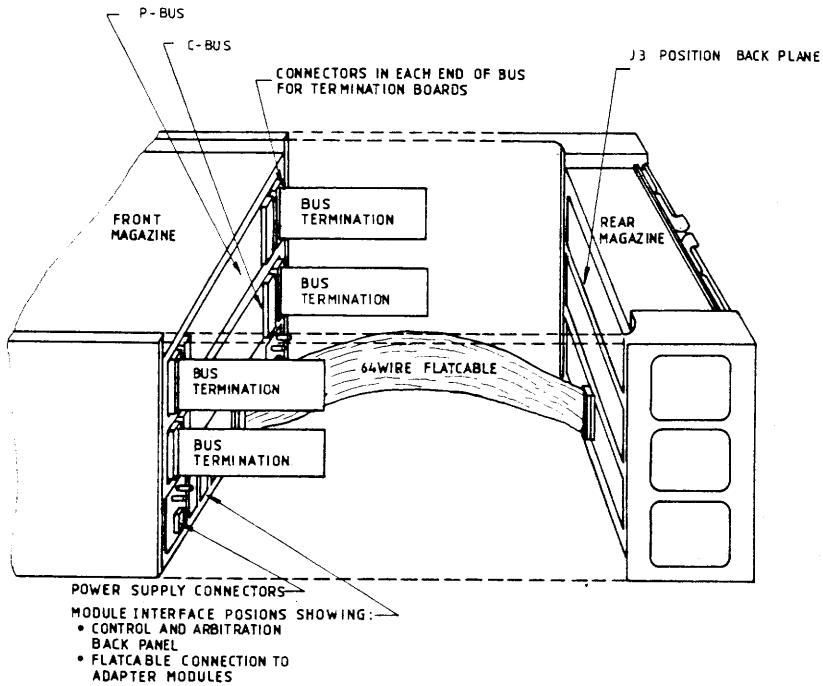


Fig. 3.2.1-2
PU-CRATE
Back Planes

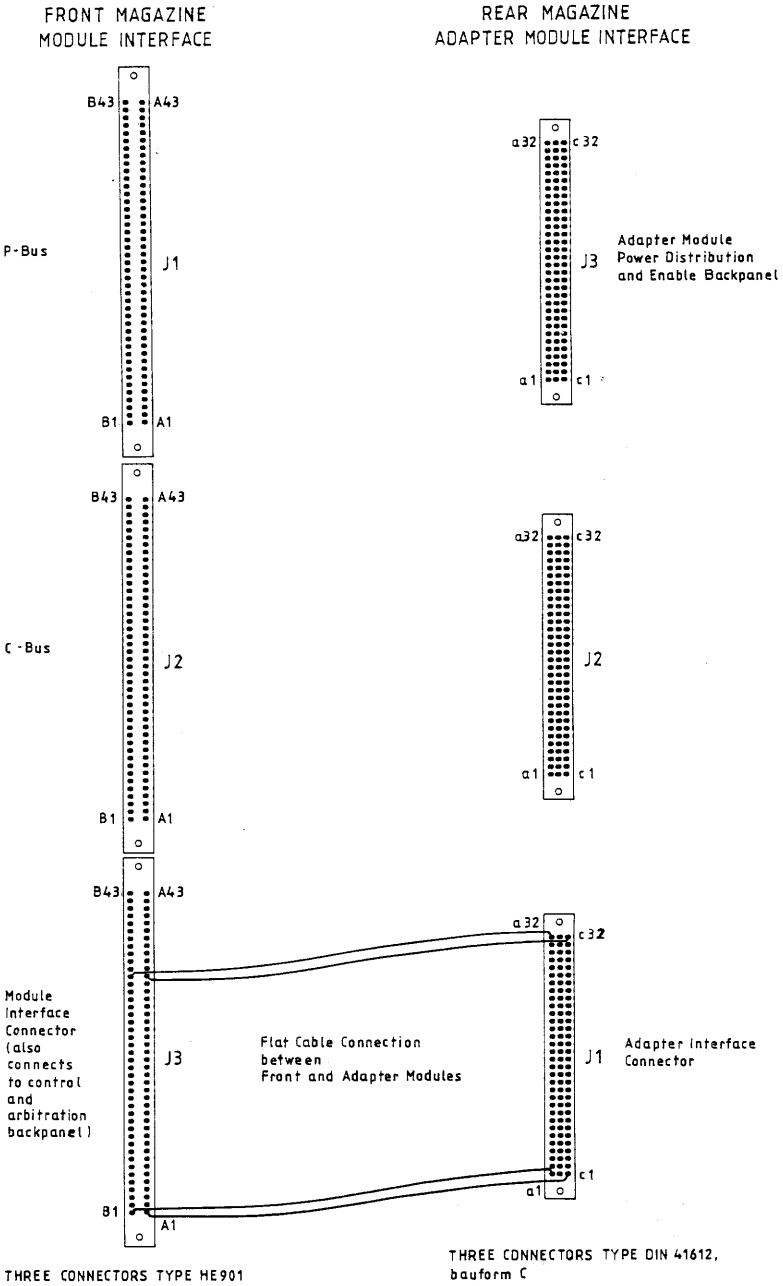


Fig.3.2.1-3 PU-Crate Back Planes, Front & Rear Magazine Connectors

For the connection between C-bus modules (address sourcing modules on the C-Bus, e.g. STI module) and corresponding adapters (e.g. SBA or TIA) the interface may be implemented as connection between the front module and multiple adapters, as defined in the Module datasheets.

The bus termination boards for the P-Bus and C-Bus are inserted from the rear of the crate and interfaced to the connectors located at the rear of the C-bus and P-bus motherboards as shown in the following (figure 3.2.1-3).

Connectors for Mains power to the processor crate are located at the rear of the crate at the left and/or the right side of the crate. Internally, within the crate, the mains power is connected to the combined power supply AC/DC connectors located at each side of the crate in the interface connector position. The DC supplies are fed from the Power Supply connector to the P-bus and C-busmotherboards, which carry the DC supplies by means of discrete wiring. If more than one Power Supply is installed in a PU-crate these are connected together and are operated in parallel. The processor crate power distribution scheme and the connections are specified in the following figure 3.2.1-5a and 3.2.1-5b.

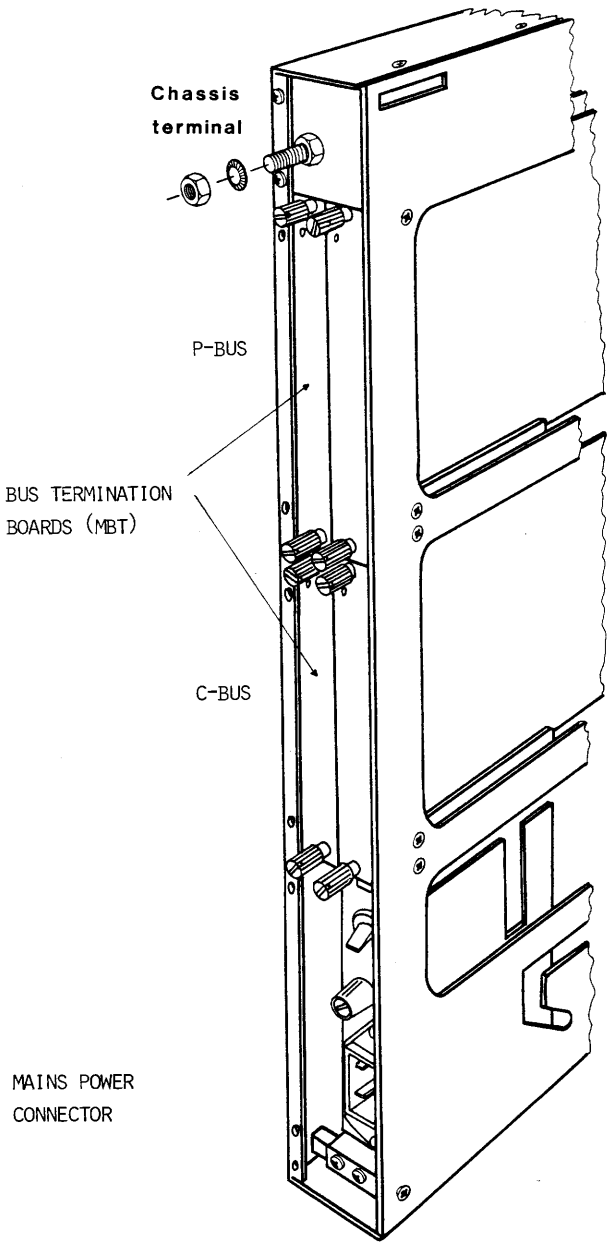
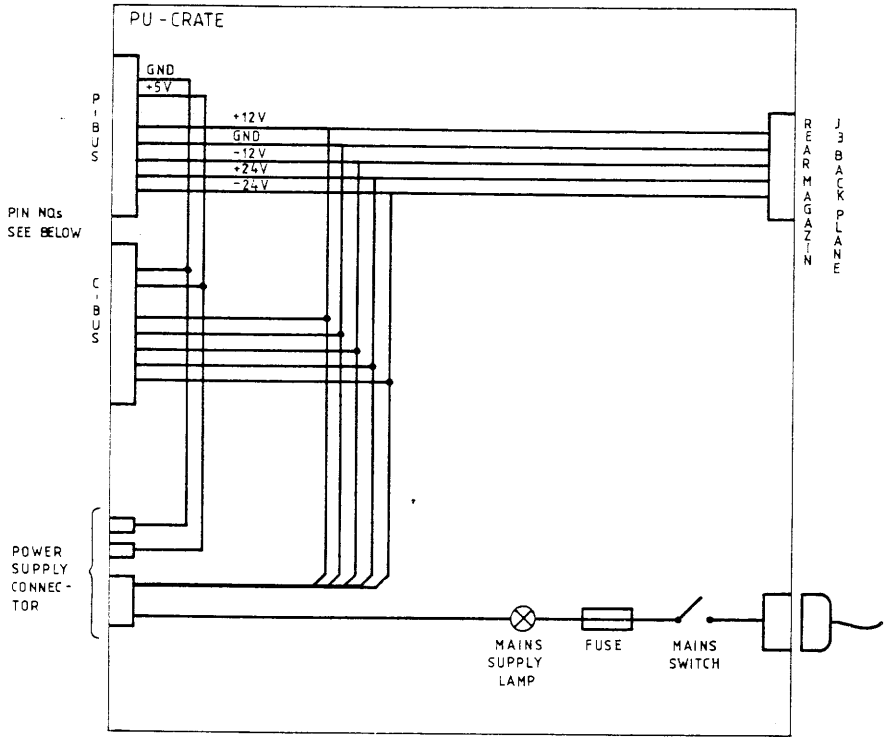
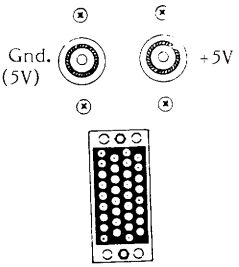


Fig. 3.2.1-4
Position of P- and C-Bus Termination Boards



Note: If two Power Supplies are used (both) DC outputs are connected in parallel.

Power Supply Connection
Crate Front View



For pin no. ref
fig. 3.2.1-5b

P-Bus and C-Bus
pin. no. of DC distribution

Pin No.	Signal
A43, B43	GND(5V)
A42,B42	GND(5V)
A41,B41	+5V
A40,B40	-12V
A14,B14	+12V
A13,B13	GND(12,24V)
A11	+24V
B11	-24V

Mains Input
Connector
Crate Rear View

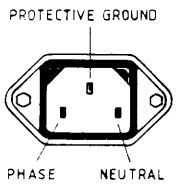
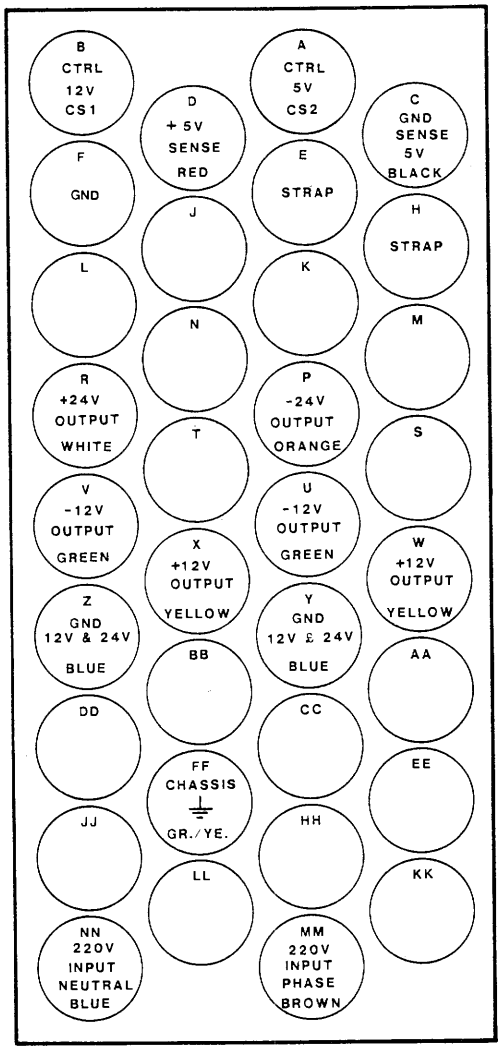


Figure 3.2.1-5a
PU Crate
Power Distribution Scheme



WIRE SIDE VIEW

Figure 3.2.1-5b

Power Supply

Pin Number Assignments

3.2.2 PU-Front Modules

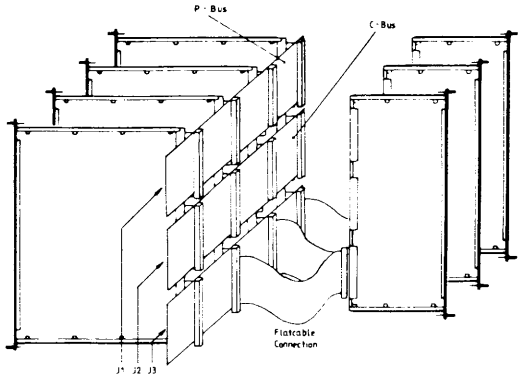
Different types of front modules are available for the CR80 Computers, such as MAP, CPU's (address sourcing on the P-bus), C-bus DMA modules (address sourcing on the C-bus), Memory modules and Power Supply.

The performance characteristics of the modules are given in the datasheet, while general specifications are given below.

The general layout of a Front Module is shown overleaf in Fig. 3.2.2-2 and Mechanical specifications in Fig. 3.2.2-3 and 3.2.2-4 for logic Modules and in Fig. 3.2.2.-5 for the Power Supply Modules.

The front modules interface to the internal bus structure of the PU-crate and if necessary to other front modules and/or rear modules as shown below.

Figure 3.2.2-1



The back panel in the module interface connector positions (J3), are designed to include standard connections for arbitration and control and so that the

interface between front and rear modules can be implemented as defined in section 3.2.5.

The connection between the MAP module (front) and the MIA module (rear) does not follow the general front-to-rear interface, but is specially designed as specified in section 3.2.5.1 on MAP module interface connector J3 specification.

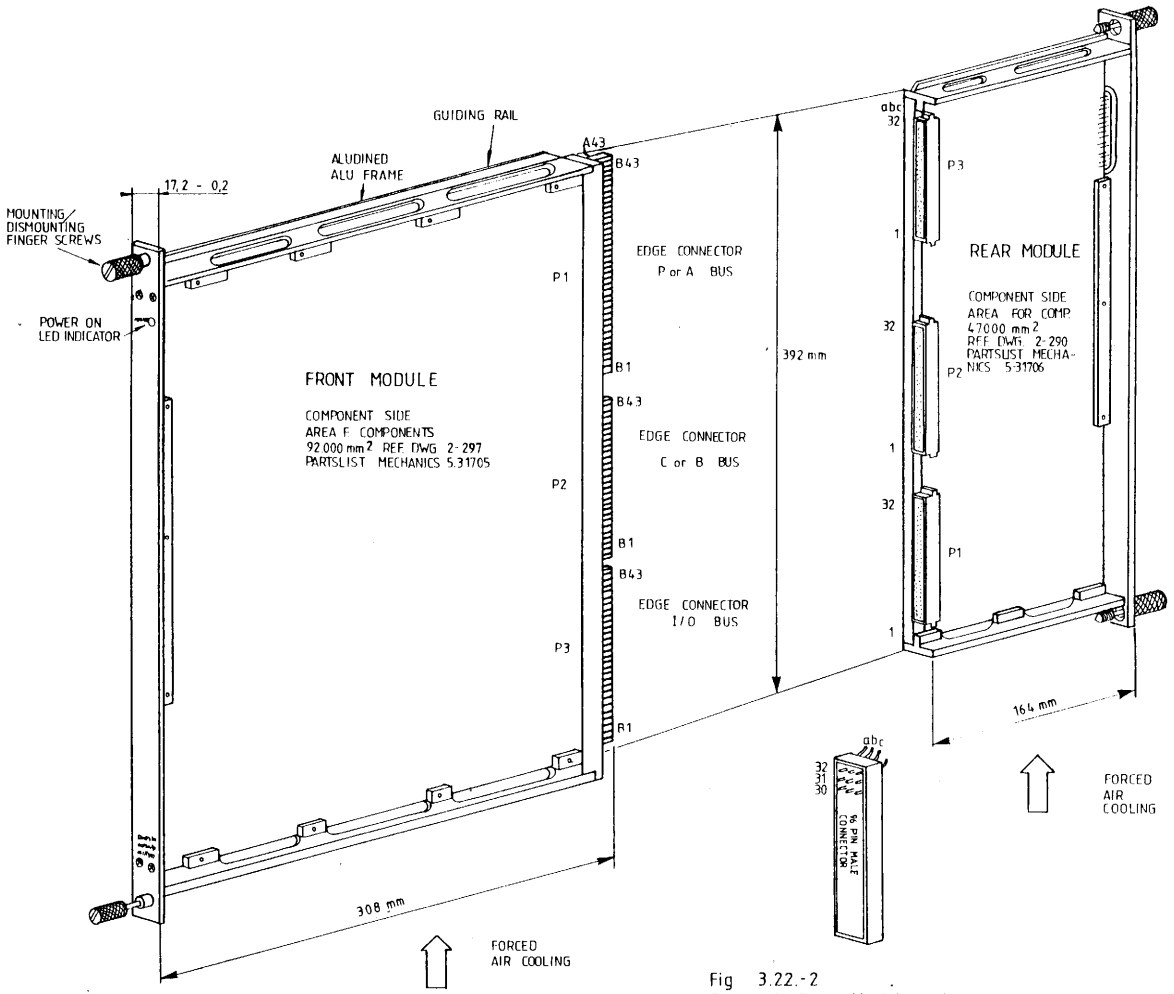


Fig 3.22.-2
Front & Rear Modules
General Layout

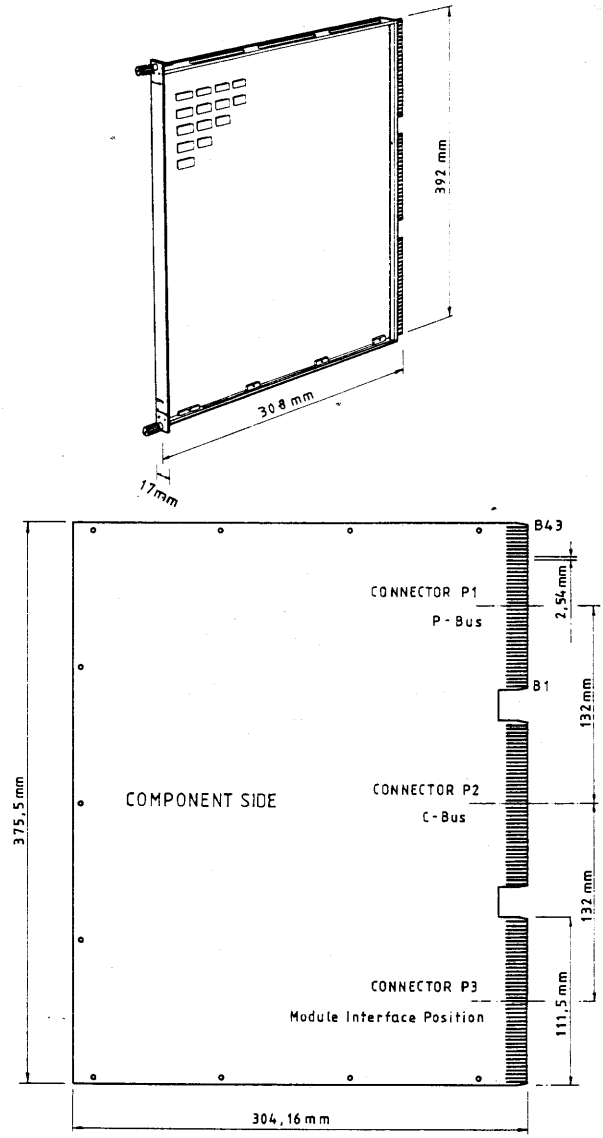
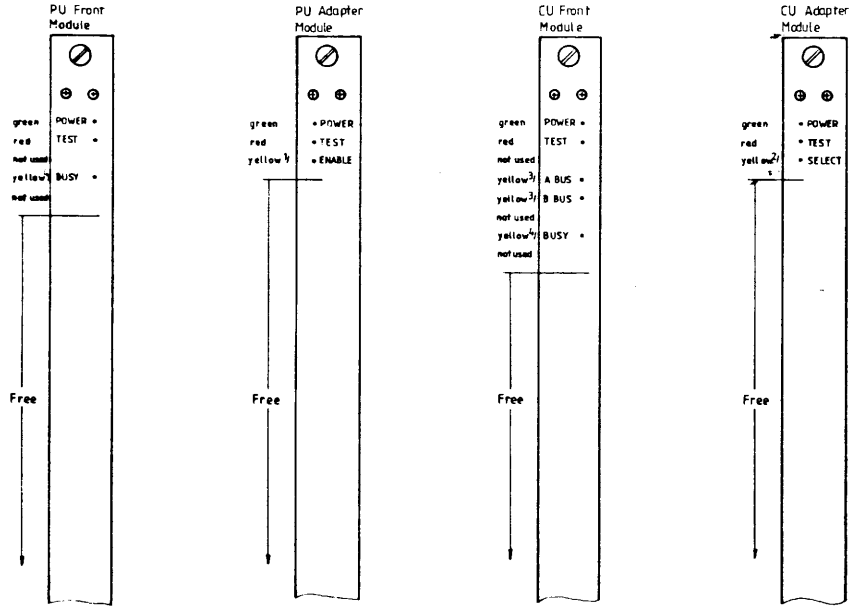


Fig. 3.2.2-3

Front Module Mechanical Specification



All LED position, except 1 and 2, only yellow LED's allowed. Red is however allowed in case of ERROR, ALARM or SIMILAR indications which require operator assistance.

1) Indicates that PU-adapter is enabled by the "PU-enable signal" in J3
2) Indicates that I/O-adapter is selected, e.g. by maintenance and configuration processor (MCP) as being connected
3) Indicates if A-Bus, B-Bus or none have bus interface selected for the CU-Module
4) "BUSY" indicates some activity in module

Fig 3.2.2-4

General indicator lay out for CR 80 Front Modules and Adapter Modules

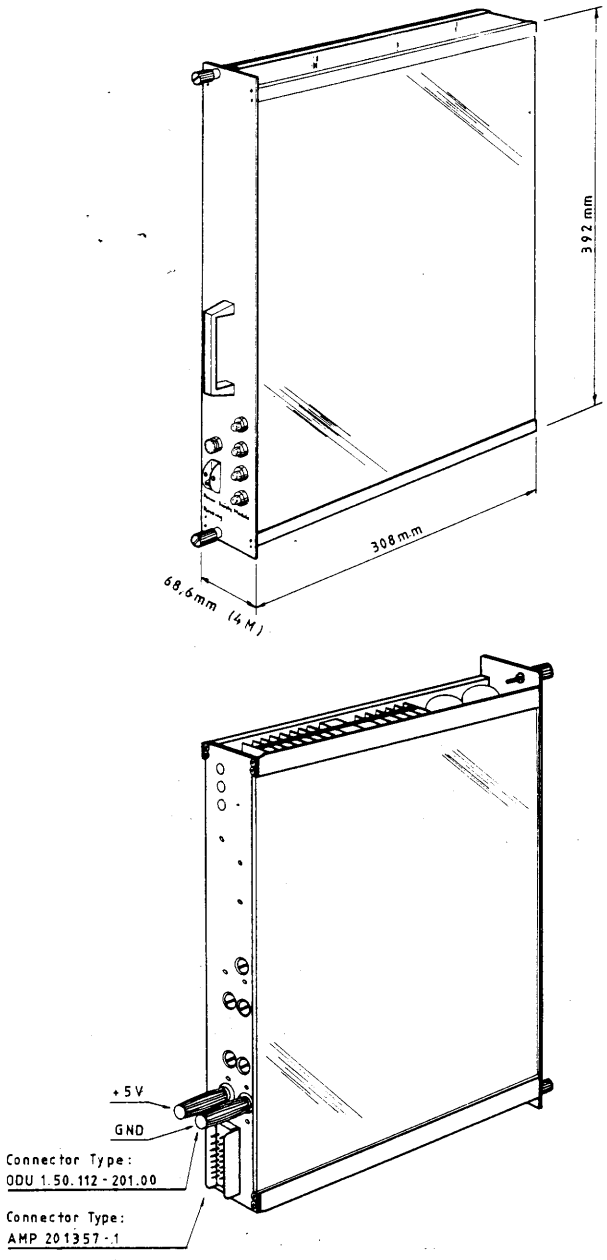
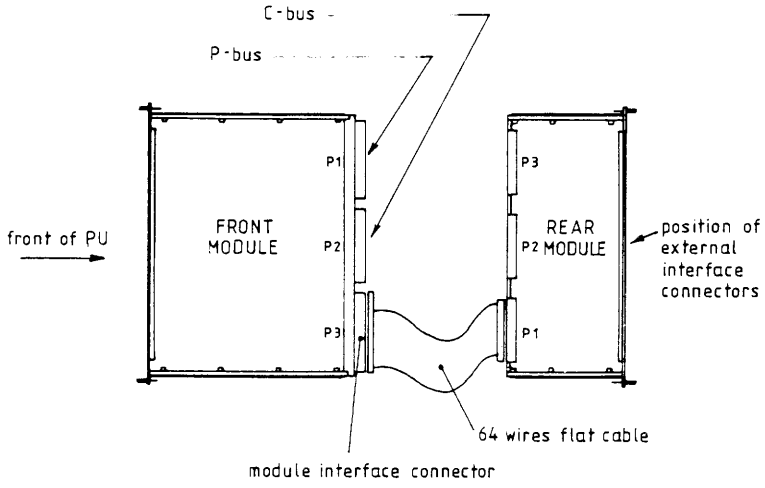


Figure 3.2.2-5
Power Supply Mechanical Specification

3.2.3 PU Adapter Modules (Rear Modules)

Different types of Adapter Modules are available for the PU, e.g. MIA (MAP Interface Adapter) and SBA (Supra Bus Adapter). The purpose of the Adapter modules is to make the physical interface between the processing module and the PU external connection. The adapters are rear modules to allow for the external connections on the adapter front panel. Connection between the front module and the adapter module is performed by means of flat cables integrated in the crate. The principles are illustrated in figure 3.2.3-1 below.

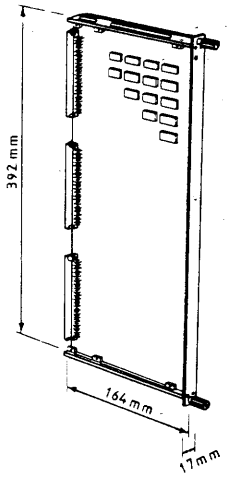
Figure 3.2.3-1, Adapter Module Interfaces



The PU external connections are such as Supra Bus (section 3.8), Data Channel (section 3.7), TDX Bus (Chapter 8) and Configuration Bus (Chapter 7).

The performance characteristics of the different Adapter Modules are given in their datasheets. The general layout of a rear module shown in Fig. 3.2.2-2 and mechanical specifications in figure 3.2.3-2 and general indicator layout in figure 3.2.2-4. The interface between the different Adapter Modules and Front Modules, when connected by the standard 64-wire flat cable is specified in section 3.2.5. Table 3.2.5a on Front to Rear Module backplane Interface Connections (J3 to J1) in particular describes the general case.

Power and Configuration Control Signals is distributed to the Adapter Modules via the Adapter Crate, J3 position backplane, section 3.9.



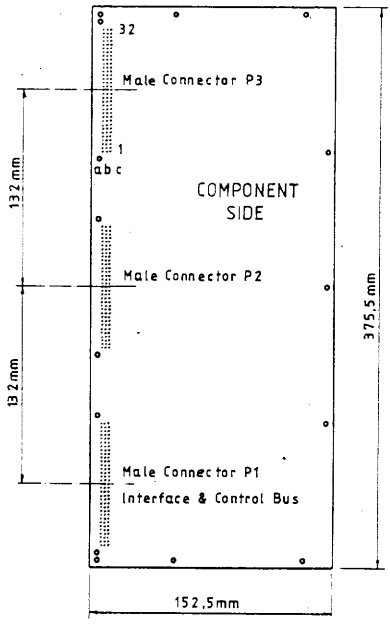
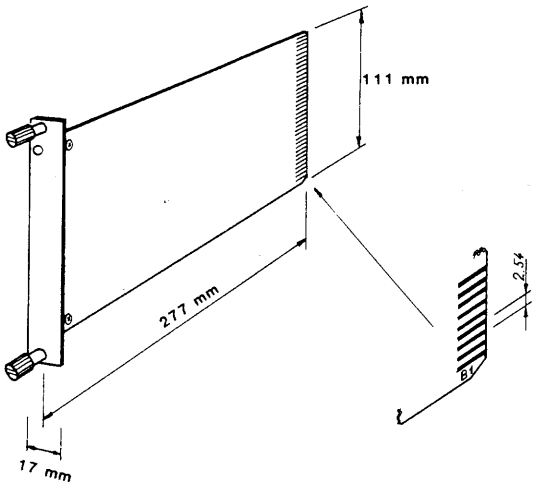


Fig. 3.2.3-2
REAR MODULE MECHANICAL SPECIFICATION

3.2.4 PU Bus Termination Modules

Two types of Bus Termination Modules are available for the CR80 system. One is a printed circuit board with no mechanical frame; this board is an integrated part of the mini crate (see data sheets). The other, specified below, is a plug-in module used for termination of the standard CR80 data transfer buses: P, C, A and B-bus (Processor-, Channel Bus, Data Bus A and Data Bus B) in PU and CU-crates.

The module is inserted in the crate from the rear (one, in each end of the bus) and fixed by means of two finger screws. The mechanical specifications for the Bus Termination Module is given in figure (3.2.4-1) overleaf and the electrical specifications comply with the bus specifications, (refer to the P-bus specification, section 3.4).



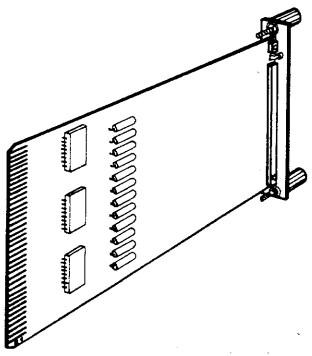
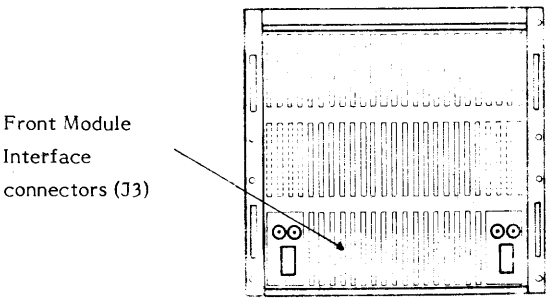


Figure 3.2.4-1
Bus Termination Module (MBT)
For P and C-Bus (A and B-Bus)

3.2.5 Front Module Interface Connector (J3)

The Front module Interface connectors J3 are located within the PU-crate in the front magazine as shown below. This section covers the general aspects of the J3 position, while MAP, CPU, C-Bus DMA's etc.'s interface in the J3 position is treated in depth in the following sections (3.2.5.1 - 3.2.5.4).

Figure 3.2.5-1
PU-Crate Front View



The Front module backplane interface connectors are used for individual connections between front modules and their corresponding adapter module(s). The Front module interface connectors are mounted on a printed circuit board with the connectors for the front modules (2x43 pin edge connector) on the front of the print and connectors for the adapter interface flatcable (64 pin) at the rear, (see figure 3.2.5-2 overleaf).
Remaining pin connections are used for control and arbitration, as defined in the following sections for individual Front module types:

3.2.5.1 : MAP
3.2.5.2 : CPU (address sourcing on the P-bus)
3.2.5.3 : STI and other DMA modules (address sourcing on the C-bus)
3.2.5.4 : Other Front Modules

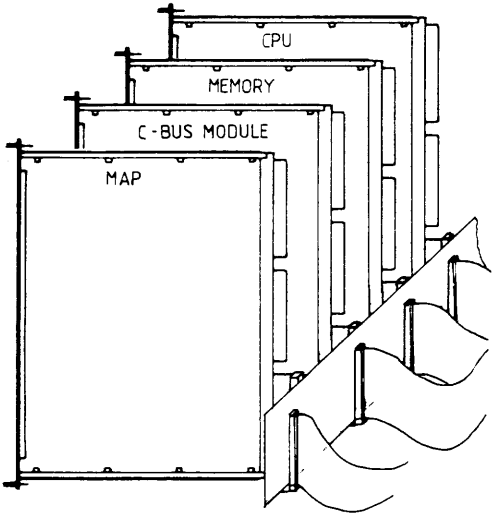


Figure 3.2.5-2, Front Module Interface connectors (J3), Front Module to Adapter module connectors, and J3 backpanel for control and arbitration signals

The following should be observed with regard to the Front Module interface connectors (J3) position:

1. The connector for the MIA (MAP Interface Adapter) flatcable is not located directly behind the MAP backplane connector (J3).

2. Connection to Adapters from C-bus addressing source modules uses front module backplane interface connectors (J3) with wrap pins.

3. The CPU modules are interfaced to the MAP module (for P-Bus arbitration and control) by means of J3 backpanel and therefore no rear connectors are mounted.

4. Memory Modules have no interface in the Front Module backplane Interface connector (J3) position

Front Magazine J3	Connection to	Front Magazine J3	Connection to
A43	Defined by module type (Slot No.)	B43	Defined by module type (Slot No.)
A42	-"-	B42	-"-
A41	-"-	B41	-"-
A40	-"-	B40	-"-
A39	-"-	B39	-"-
A38	-"-	B38	-"-
A37	-"-	B37	-"-
A36	-"-	B36	-"-
A35	-"-	B35	-"-
A34	Not used	B34	Not used
A33	c32 Rear Magazine J1	B33	a32 Rear Magazine J1
A32	c31 -"-	B32	a31 -"-
A31	c30 -"-	B31	a30 -"-
A30	c29 -"-	B30	a29 -"-
A29	c28 -"-	B29	a28 -"-
A28	c27 -"-	B28	a27 -"-
A27	c26 -"-	B27	a26 -"-
A26	c25 -"-	B26	a25 -"-
A25	c24 -"-	B25	a24 -"-
A24	c23 -"-	B24	a23 -"-
A23	c22 -"-	B23	a22 -"-
A22	c21 -"- standard	B22	a21 -"- standard
A21	c20 -"- 64p	B21	a20 -"- 64p
A20	c19 -"- flatcable	B20	a19 -"- flatcable
A19	c18 -"- connection	B19	a18 -"- connection
A18	c17 -"-	B18	a17 -"-
A17	c16 -"-	B17	a16 -"-
A16	c15 -"-	B16	a15 -"-
A15	c14 -"-	B15	a14 -"-
A14	c13 -"-	B14	a13 -"-
A13	c12 -"-	B13	a12 -"-
A12	c11 -"-	B12	a11 -"-
A11	c10 -"-	B11	a10 -"-
A10	c9 -"-	B10	a9 -"-
A9	c8 -"-	B9	a8 -"-
A8	c7 -"-	B8	a7 -"-
A7	c6 -"-	B7	a6 -"-
A6	c5 -"-	B6	a5 -"-
A5	c4 -"-	B5	a4 -"-
A4	c3 -"-	B4	a3 -"-
A4	c2 -"-	B3	a2 -"-
A2	c1 -"-	B2	a1 -"-
A1	Not used	B1	Not used

Note: The MIA to MAP interface does not follow this scheme, but is defined in section 3.2.5.1.

Table 3.2.5a
General Front Magazine backplane connector(J3) and
Rear Magazine backplane connector (J1) Definitions

3.2.5.1 MAP Backplane Interface connector (J3) specification

One module slot in the front magazine is prepared for the MAP module and one location in the rear magazine for the MIA module.
Beside the connections to the MIA module the MAP also has connections to the CPU's and DMA modules (STI's etc.) via the control and arbitration backpanel, for performing the bus arbitration function and interrupt function.

The MAP Interface and Control Bus backplane connector (J3) pin layout is as defined in the table overleaf (3.2.5.1a). Signals between the MAP and the MIA are defined in the relevant product specifications, while the remaining signals are specified in the following.

REAR VIEW

	B		A	
	GND	43	PUAEN	
	ALLEN	42	ALLEN	
	EMM(L)	41	ECL(L)	
	GND	40	PFL(L)	
	PBG0(L)	39	GND	
	PBG2(L)	38	PBG1(L)	
	GND	37	GND	
	PBG4(L)	36	PBG3(L)	
	GND	35	GND	
	INT2	34	IST(L)	
	INT0	33	INT1	
Signals for special (wire Wrap)connections and signals to CPU's and C-Bus modules	GND	32	GND	Signals for special (wire wrap) connections and signals to CPU's and C-Bus modules
	CRQ1(L)	31	CBG1(L)	
	CRQ2(L)	30	CBG2(L)	
	GND	29	GND	
	CRQ3(L)	28	CBG3(L)	
	CRQ4(L)	27	CBG4(L)	
	GND	26	POF(L) to MIA	
	PRQ0(L)	25	LBG(L)	
	PRQ2(L)	24	PRQ1(L)	
	GND	23	GND	
	PRQ4(L)	22	PRQ3(L)	
	GND	21	GND	
		20		
		19		
		18		
		17		
		16		
		15		Connections to MIA by means of flat cable, (signals specified in MAP MIA product specification)
		14		
		13		
		12		
		11		
		10		
		9		
		8		
		7		
		6		
		5		
		4		
		3		
		2		
		1		

Table 3.2.5.1a
Backplane Interface connector (J3) of the MAP Module

3.2.5.1.1 MAP, J3, Electrical Interface Specification

- P-Bus and C-Bus Arbitration Lines:

  PRQ0-PRQ4

  CRQ1-RQ4
  PBG0-PBG4
  CBG1-CBG4
  Driver: $I_{OL} \geq$ 48 mA
  Receiver: $I_{IH} \leq$ 100 uA
  $I_{IL} \leq$ 2 mA

  LBG:
  Open collector signal.

  Driver: $I_{OL} \geq$ 48 mA
  $I_{OH} \leq$ 100 uA
  Receiver: $I_{IH} \leq$ 100 uA
  $I_{IL} \leq$ 2 mA

- Map notification lines to CPU's (CPU Interrupt):

  INT (2:0), IST:

  Driver: $I_{OL} \geq$ 60 mA
  Receiver: $I_{IH} \leq$ 100 uA
  $I_{IL} \leq$ 2 mA

- PFL(L), ECL(L), EMN(L)

  Open collector signal:

  Driver: $I_{OL} \geq$ 48 mA
  $I_{OH} \leq$ 100 uA
  Receiver: $I_{IH} <$ 100 uA
  $I_{IL} \leq$ 15 mA

- PUAEN, ALLEN:

  Driver: Current sourcing "open collector" with following specifications:
  $I_{OH} >$ 70 mA V_{out} = 4V
  $I_{off} <$ 100 uA V_{out} = 0.4V
  $I_{off} <$ 8 mA V_{out} = 4V

  Receiver: $I_{IL} <$ 100 uA
  $I_{IH} <$ 4 mA V_{in} = 2.7V
  $I_{IH} <$ 8 mA V_{in} = 5V

3.2.5.1.2 MAP, J3, Functional and Timing Interface

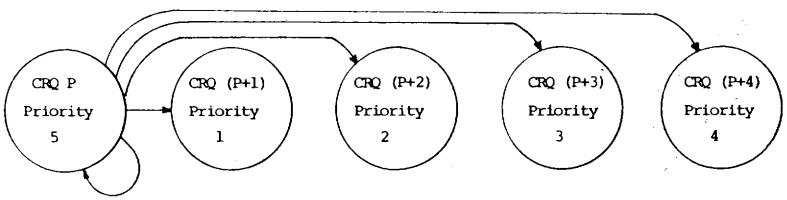
- MAP, Bus Authority Control

The MAP incorporates the arbitration function for the address sourcing modules on both PU busses: P-bus and C-bus . The arbitration is controlled by Module Request/Grant signals on each of the busses. These signals are connected between the CPU's (P-Bus), DMA's (C-Bus) and the MAP by means of the backpanel in the J3 position.

For the C-bus, the following signals are used (for pin-lay-out, refer to table 3.2.5.1a):

C-Bus Request 1-4: CRQ1-CRQ4
C-Bus Grant 1-4: CBG1-CBG4

The Data Channel DMA included in the MAP uses CRQ0 and CBG0. Each of the address sourcing modules on the C-Bus (DMA's) have one dedicated request signal, and a grant-signal, which is monitored to determine whether it is allowed to access the bus or not.
The authority control gives the same access rights to all modules and is implemented as defined below.



Module currently accessing the bus "CRQ P".
The priority of the modules bus request (CRQ) depends of the current condition of the authority control; the module accessing the bus "CRQ P" has lowest priority while the module with CRQ (P+1) has highest priority, meaning that the bus authority (CBG) will be given to the requesting module with the currently highest priority.

The P-Bus authority is controlled by means of the following signals:

P-Bus Request 0-4, PRQ0-PRQ4
P-Bus Grant 0-4, PBG0-PBG4
Lock Bus Grant , LBG

Each of the attached address sourcing modules are assigned a set of PRQ, PBG-signals, and a common LBG-line is used which may be activated by an address sourcing module in order to "lock" bus arbitration.

The timing and signalling sequences are shown in the following figures (3.2.5.1.2-1,2,3) and are valid for both P-Bus and C-bus arbitration (allowing for the fact, that no "lock bus" facility exists on the C-bus).

In the timing specs, the following notation is used.

- Numbers in square brackets e.g. $0<t<40$ are absolute maximum ratings for the propagation delay from the reference signal to an output (measured at the bus connector, with the module connected to the bus).

- Numbers in curled bracket e.g. $t>50$ specify guaranteed worst case delay between two input-signals, assuming a maximum bus length of 2m (corresponding to a skew of ≤ 5 ns.)

Figure 3.2.5.1.2-1

TIMING SPECS. FOR BUS ARBITRATION

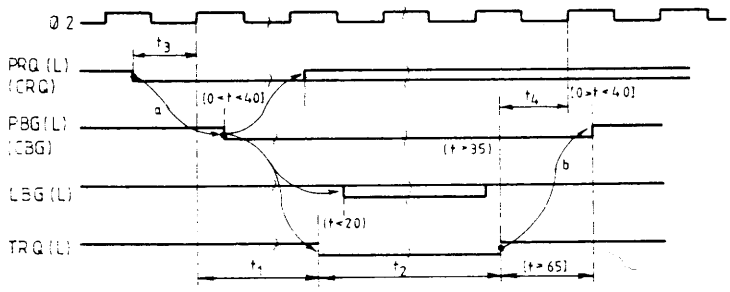


Figure 3.2.5.1.2-2

TIMING SPECS FOR BUS ARBITRATION, AT "NO-TRQ"-CONDITION

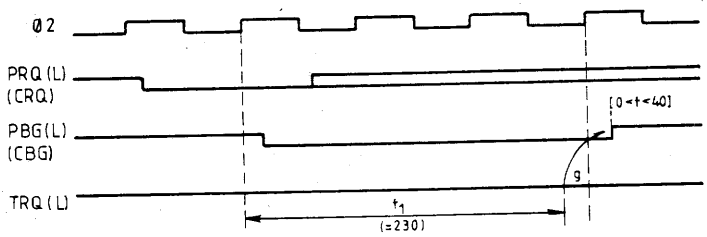
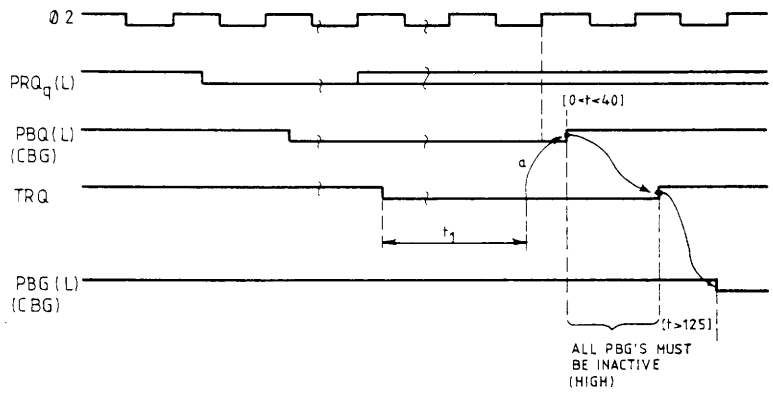


Figure 3.2.5.1.2-3
TIMING SPECS. FOR BUS ARBITRATION; AT "TIMEOUT" - OR
"ILLEGAL ACCESS ATTEMPTED" - CONDITION

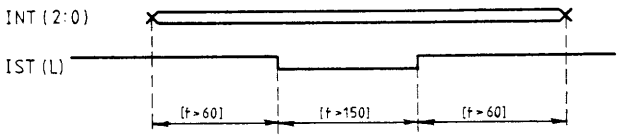


- MAP, J3, Notification Signals to CPU's (CPU interrupt):

The interrupt signals INT(0-2), IST are used for notification of the CPU's in a processing unit. The meaning of the lines INT (0-2), with IST low, is seen in the table below, while the timing is defined in the figure below.

INT(0-2)	Function
000	Notify CPU # 0
001	Notify CPU # 1
010	Notify CPU # 2
011	Notify CPU # 3
100	Notify CPU # 4
101	NOT USED
110	NOT USED
111	Timer Interrupt

Figure 3.2.5.1.2-4
TIMING SPECS. FOR "INTERRUPT PROCESSING CIRCUIT"
(NOTIFICATIONS, TIMER INTERRUPT)



- Other MAP, J3, Signals

ECL(L)

ECL(L) (External Clear) is an input to the MAP module. When ECL(L) is low, the MAP generates a Master Clear on the C-Bus and P-Bus, meaning that the PU is cleared.

ECL(L) may be activated by the Maintenance and Configuration Processor (MCP) system or other devices external to the MAP.

PFL(L)

Power Failure Lookahead. This is an input to the MAP. When PFL is activated, the MAP generates a power failure interrupt. This interrupt will activate special system software, which may save part of main memory, e.g. on disk storage.

PFL is an early warning, which is generated in the power supply or by an external detector, e.g. in the AC power system, when the power tends to fail.
PFL should be active, until the power becomes sufficient again.

EMM(L)

EMM(L) (External Maintenance Mode) is an input to the MAP module. The MAP may operate in either of two modes: Normal Mode and Maintenance Mode.

Maintenance Mode is obtained by activating EMM(L) (e.g. from the Watch-dog sytem) or by a switch on the MAP front panel.

Maintenance Mode is only possible while the PU is disabled. As EMM(L) goes high, the system returns to normal mode, provided the MAP front panel switch is not in the 'maintenance mode' position.

PUAEN, ALLEN

PUAEN, PU Adapter Enable, must be high to enable the connections external to the PU (Supra bus, TDX bus, Data Channel). If PUAEN is low, all adapters will disable their external interfaces, except the AV24 port in the MIA.
PUAEN is normally sourced from the MIA (see fig. 3.2.5.1.2-5 overleaf), but other modules may be used, as well as PUAEN may be directly connected to ALLEN:

ALLEN, ALLow ENable, is a MAP generated signal, which is taken high, when the MAP allows the PU to be enabled. Normally ALLEN and PUAEN are used as shown in the figure overleaf. Thus the PU is enabled, only if all "switches" are closed.

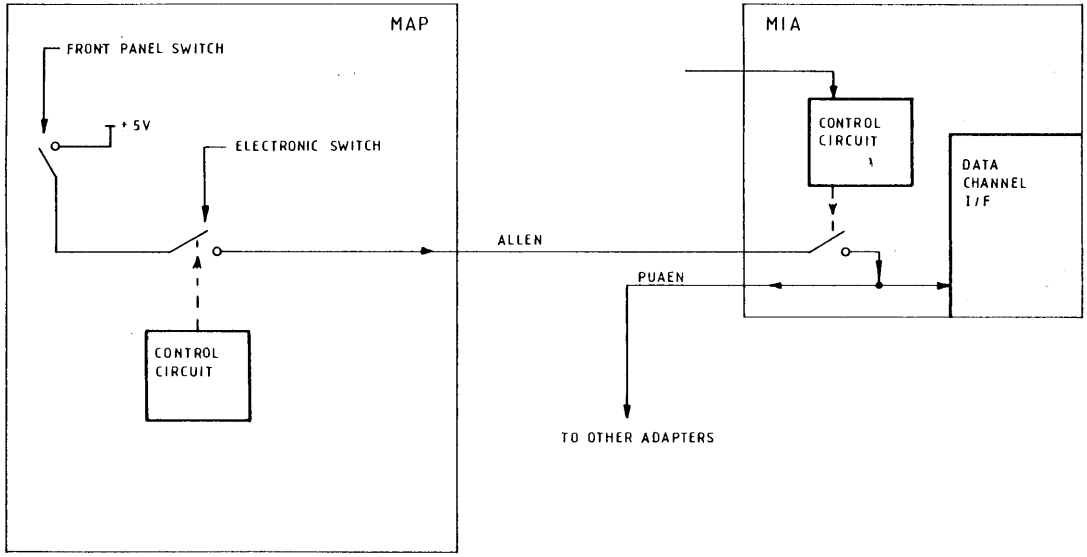


Figure 3.2.5.1.2-5
FUNCTION OF PU ENABLE SIGNALS

3.2.5.2 CPU Module, Interface Connector (J3) Specification

Up to five module slots are available in the front magazine for interfacing CPU modules. The control and arbitration backpanel (J3) carries the signals to the MAP for bus arbitration on the P-Bus and interrupt control.

The CPU Interface connector (J3) pin layout is as defined in table overleaf (3.2.5.2-1). The signals elecrical and timing specifications are given in the MAP module interface connector specification, section 3.2.5.1 except for the signal FIN(L) which is used in connection with CO-Processor modules to notify the CPU that its CO-Processor has completed an operation. The signal is normal TTL with the following specifications:

Driver: $I_{OL} \geq 16$ mA
$I_{OH} \geq 5.2$ mA

Receiver: $I_{IH} \leq 100$ uA
$I_{IL} \leq 0.5$ mA

REAR VIEW

B		A	
GND	43	IST	
GND	42	INT2	
GND	41	INT1	
GND	40	INT0	
GND	39		
GND	38	LBG(L)	
GND	37		
GND	36	FIN(L)	
GND	35		
GND	34	PRQ(L)	
GND	33	PBG(L)	
	32		
	31		
	30		
	29		
	28		
	27		
	26		
	25		
	24		
	23		
	22		
	21		
	20		Not connected
	19		
	18		
	17		
	16		
	15		
	14		
	13		
	12		
	11		
	10		
	9		
	8		
	7		
	6		
	5		
	4		
	3		
	2		
	1		

Figure 3.2.5.2-1
Backplane Interface Connector (J3))
Pin layout for CPU slots

3.2.5.3 C-Bus module (CBM), Interface Connector (J3) Specification

Up to four module slots are available in the front magazine for interfacing Modules connected (and address sourcing) on the C-bus (STI, DMA modules, CO-Processors, etc). The control and arbitration backpanel (J3) carries the signals to the MAP for bus arbitration on the C-Bus.

The connector pin lay-out is as defined in table overleaf (3.2.5.3a). The signals, electrical, and timing specifications are as given for the MAP module interface connector specification, section 3.2.5.1 except for the signals FIN0(L) - FIN4(L) which are as defined in CPU module interface connector specification section 3.2.5.2 (one FIN signal per possible CPU).

REAR VIEW

B		A	
GND	43	FIN0(L)	
GND	42	PBG0(L)	
FIN1(L)	41	GND	
PBG1(L)	40	FIN2(L)	
GND	39	PBG2(L)	
FIN3(L)	38	GND	
PBG3(L)	37	FIN4(L)	
GND	36	PBG4(L)	
GND	35	CRQp(L)	
GND	34	CBGp(L)	
	33		
	32		
	31		
	30		
	29		
	28		
	27		
	26		
	25		
	24		
	23		
	22		
	21		
	20		Available for 64 wires flatcable connection to adapter.
	19		
	18		
	17		
	16		
	15		
	14		
	13		
	12		
	11		
	10		
	9		
	8		
	7		
	6		
	5		
	4		
	3		
	2		
	1		

Table 3.2.5.3a
Backplane Interface Connector (J3)
Pin layout for C-Bus Module Slots

3.2.5.4 Other PU-Front Modules, Backplane Interface Connector (J3) Specification

A number of module slots will be available in the front magazine depending on the selected crate type, some of these slots can be used for application defined purposes while the remaining slot have to be equipped with modules without connection to the Interface Connector (J3).

The positions which can be used for special application purposes are not equipped with connector in the control and arbitration backpanel but the printed circuit board is prepared for it. When the connectors are mounted one for the front module and one for the 64-wire flat cable to the rear module the interface will be as defined overleaf in table 3.2.5.4a.

REAR VIEW

B		A
	43	
	42	
	41	
	40	Wire Wrap pins for special connections
	39	
	38	
	37	
	36	
	35	
	34	
	33	
	32	
	31	
	30	
	29	
	28	
	27	
	26	
	25	
	24	
	23	
	22	
	21	
	20	Available for 64 wires flatcable connection to adapter.
	19	
	18	
	17	
	16	
	15	
	14	
	13	
	12	
	11	
	10	
	9	
	8	
	7	
	6	
	5	
	4	
	3	
	2	
	1	

Table 3.2.5.4a
Interface Connector (J3) Lay-out
For optional module slots

3.2.6 PU External Interfaces

The external interfaces of the Processor Units are the following:

- Data Channel (separately defined later in section 3.7)
- Supra Bus (S-NET) (separately defined later in section 3.8)
- TDX Bus (described under the TDX and X-Net subsystem chapter 8 and 14)
- Configuration Bus (described under the Maintenance processor and Configuration Subsystem (MCP) Chapter 7)
- PU system console I/F

The connectors for these interfaces are located on the front panel of the corresponding adapter modules, while details about the module position depends of the selected PU-Crate (as defined in the PU-data sheets).

As listed above, the PU external interfaces are found elsewhere in this chapter, while the system console interface electrical, physical and timing are given below.

The system console interface is located at the front of the MIA and has the following specifications:

Compatible with CCITT V24/V28 recommendations DCE Interface.
The Interface is asynchronous and operates at up to 9.6K bit/sec.

The pin layout of the connector is shown overleaf (Table 3.2.6a).

Connector type: Cannon DB25S

Pin layout

Pin No.	Signal
1	Protective GND (strap selectable)
2	Transmitted data (circuit 103)
3	Received data (circuit 104)
4	Request to send (circuit 105)
5	Ready for sending (circuit 106)
6	Data set ready (circuit 107)
7	Signal GND (circuit 102)
8	Data carrier detect (circuit 109)
9	+12V (strap selectable)
10	Disable PU (TTL, active low, strap selectable)
11	NC
12	NC
13	NC
14	NC
15	NC
16	NC
17	NC
18	NC
19	NC
20	Data terminal ready (circuit 108/2)
21	NC
22	NC
23	NC
24	NC
25	NC

Table 3.2.6a MIA System Console Connector

3.3 CR80 CU (Channel Unit)

Introduction

The CR80 Channel Unit, (CU), is an electrically and physically self-contained Extension subsystem for the CR80 Processor Units (PU's), increasing physically the possible memory size of Processing Elements (PE's). Modules typically positioned in Channel Units (CU's) are RAM Modules and Peripheral Modules containing compartmentalized memory. The capacity of the CU is dependent on the power consumption and interface requirements of the installed modules, but can be as high as 17 Peripheral Modules (e.g. corresponding to 64 communication lines etc.) or/and up to 1 megawords of PE memory. Up to 15 CU's can be attached to a PE-Datachannel, making the CR80 MAXIM computer incorporate up to more than 200 Peripheral Modules, or a fully expanded CR80 FATOM computer incorporate up to more than 2000 Peripheral Modules (e.g. corresponding to 7-8000 communication lines).

Refer to drawing overleaf (fig. 3.3-1) for functional and mechanical lay-out of the CU.
Physically the CR80 modules constituting the CU are housed in a 19" Crate (card cage) with a height of 443mm (10U) and a depth of 630mm.

The CU Crate supports insertion of the following CR80 standard modules in its front and rear positions:

Front positions (25):

- 1 or 2 Power Supply modules (1 each side) dependent on single or dualized supply of power to the installed modules.
- Peripheral modules (single or dual-ported (A and B Bus), e.g. disc, tape, communication controller.
- RAM modules (single or dual-ported (A and B Bus)

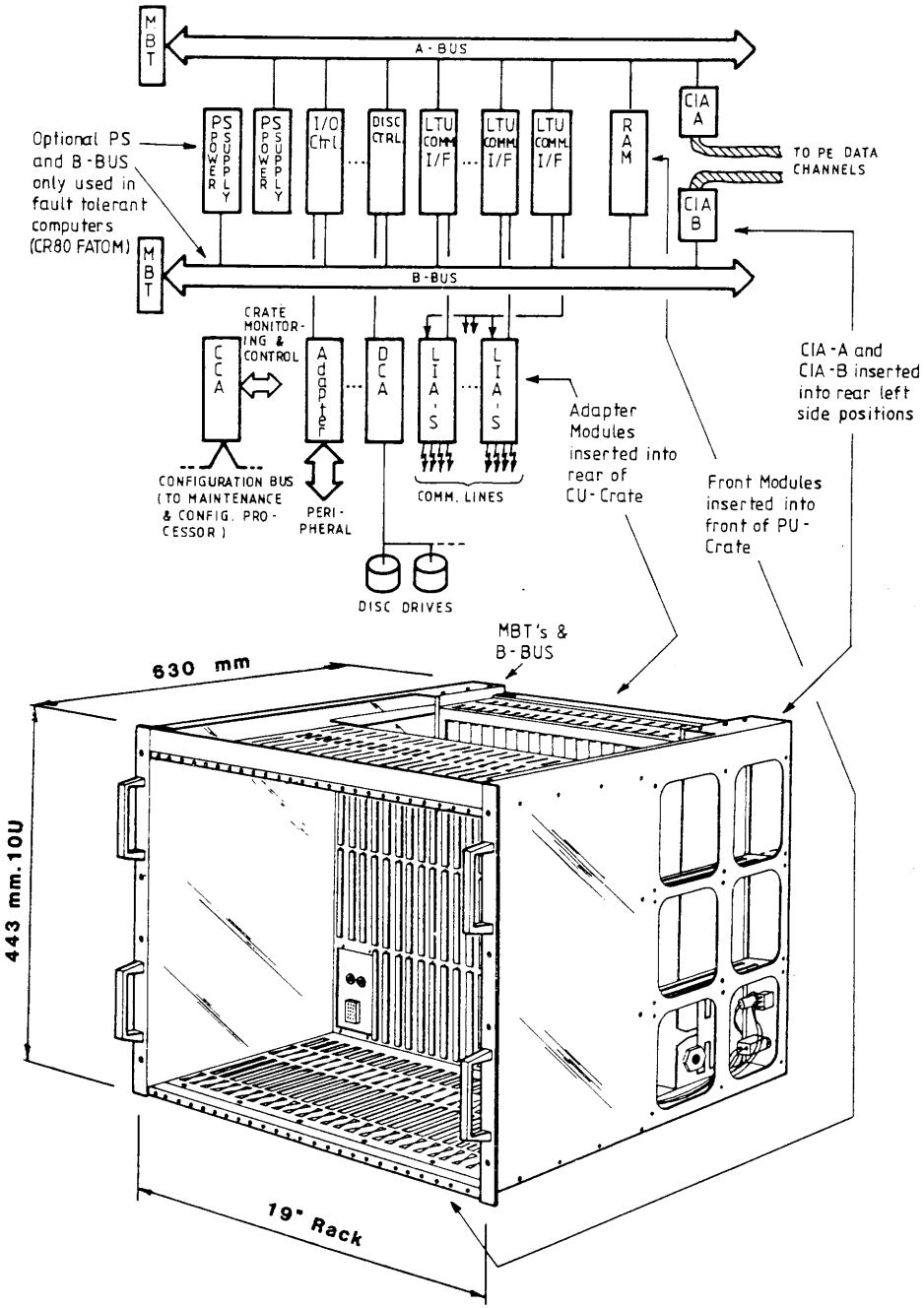


Fig. 3.3-1 CU (Channel Unit), Functional & Mechanical Layout.

Rear Adapter Cage positions (19)

- o Interface Adapters for peripheral controllers
- o CCA (Crate Configuration Adapter)

Rear side positions:

- o Bus Termination Modules, MBT (A and B Bus)
- o CU-Crate Interface Adapters, CIA-A and CIA-B, for interface of A and B Bus to PU-Datachannels.

As can be seen from the CR80 datasheets a combi-crate exists, which combines a PU and CU into one 19" Crate; since this is interface and bus compatible with the large CU-crate, only the latter is treated in the following.

The functional block diagram of the CU corresponds physically with the buses, backplanes and interconnections in the CU-crate as folows:

Front Magazine:

• Top (J1)	A-Bus (Data bus A)
• Middle (J2)	B-Bus (Data bus B)
• Bottom (J3)	Module Interface connectors, leaving minimum 64 connector pins free in each position for individual flatcable connection from front module to corresponding Adapters in rear magazine. (Exception: Special Power Supply connectors in each side).

Rear Adapter Magazine:

- Top (J3) — Backpanel for distribution of power and configuration control signals from CCA to Adapter Modules.
- Middle (J2) — Used with backpanel for switching I/O lines from Adapters to spare peripheral Modules in front magazine (also used for configuration control signals from CCA)
- Bottom (J1) — Free for 64p cable connnection between individual Adapters and corresponding front modules.

Rear, left side positions:

- Top and Middle: — CIA-A and CIA-B, CU-Crate Interface Adapters to PE-Data Channels (these modules also includes the rear left side terminations for the A and B-Buses).
- Bottom: — Mains Plug for A-Bus plug-in Power Supply.

Rear, right side positions:

- Top: Insertion of the rear, right side Bus Termination module, MBT, for the A-Bus.
- Middle: Insertion of the rear, right side Bus Termination module, MBT, for the B-Bus.
- Bottom Mains Plug for B-Bus plug-in Power supply.

As the A-Bus, B-Bus and Datachannel are separately treated later in the chapter, this section concentrates on the other aspects of the Channel Unit (CU):

- Physical and Electrical specifications of the CU-crate
- Physical dimensions and specifications of CU-modules

3.3.1 CU Crate

The CR80 Peripheral Modules are housed in a standard 19" Module magazine Channel Unit Crate (CU-Crate)

Different types of crates, for installation of modules in accordance with actual system requirements, are available from the CR80 standard product line as defined in the CU-datasheets while this section gives the general specifications for the CU-crate.

The CU-crate consists basically of a front magazine for Peripheral and/or RAM Modules and a rear magazine for the corresponding Adapter Modules placed back to back as shown overleaf (fig. 3.3.1-1).

The interface between the front modules (RAM and Peripheral Modules) and the rear modules (Adapters) is performed by means of 64p flat cable, while the interface from the front module to the CR80 system is established by means of one or two transfer busses (Data Bus A and B) connected to the PU's of associated PE's via the Data channel(s).

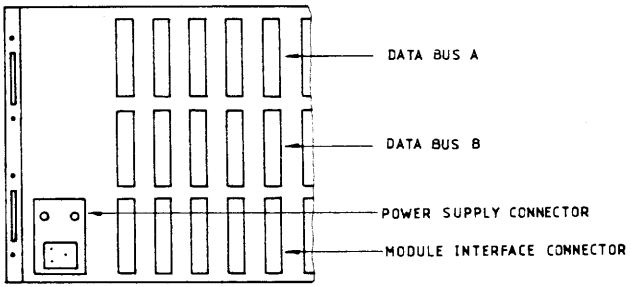
The CU-crate external connections (to Peripherals etc.) are located at the front panel of the Adapter Modules.

The interfaces described above are illustrated in figure (3.3.1-2) overleaf, where it is seen that the transfer busses (Data Bus A and B) are implemented as multilayer printed circuit board (Motherboards) equipped with connectors for receiving the edge connectors of the inserted modules (in non dualized systems only data bus A is implemented).

The rear magazine also includes two printed circuit boards (J2 and J3) which distributes dual DC power and control lines for switching of Peripheral interface signals to spare Peripheral Modules in case of primary Peripheral Module failure.

Connector specifications for front and adapter modules are given in the following figures. Also shown is the the standard flat cable connections between a front module and the corresponding Adapter Module. The actual locations within the various standard CU's, where connections are implemented, is defined in the CU-datasheets.

FRONT VIEW



SIDE VIEW

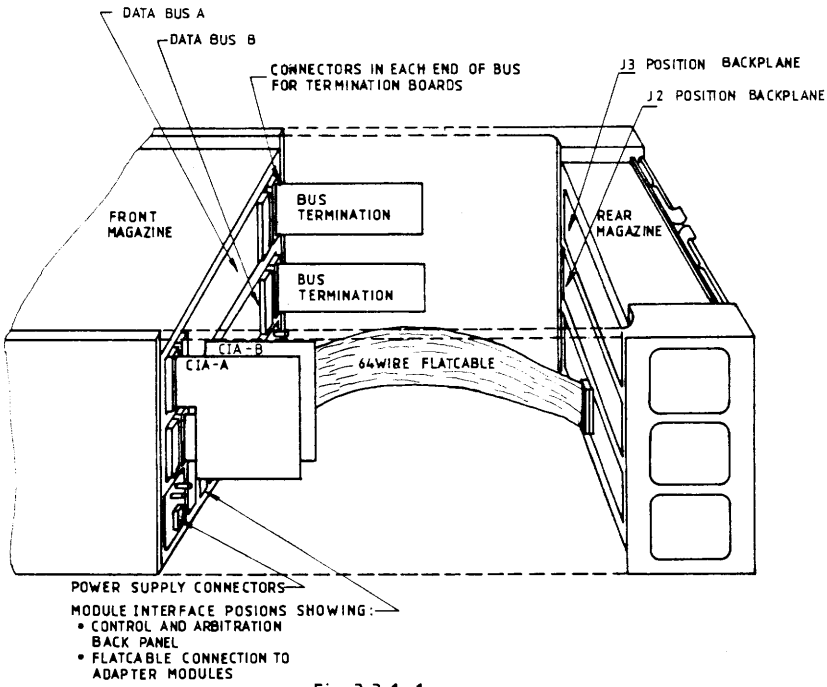


Fig. 3.3.1-1
CU-CRATE
Back Planes

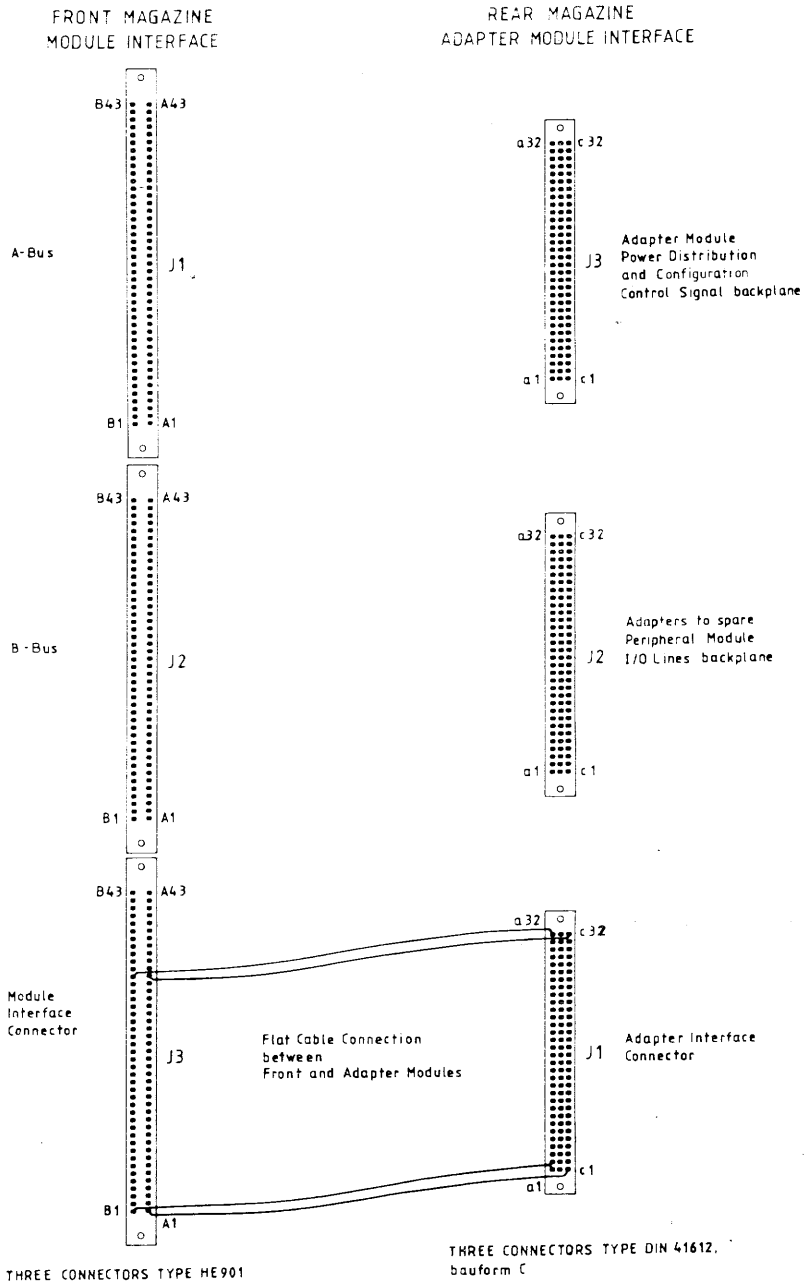


Fig. 3.3.1-2 CU-Crate, Front & Rear Magazine Connectors

Front Magazine J3	Connection to		Front Magazine J3	Connection to	
A43	Defined by module type (Slot No.)		B43	Defined by module type (Slot No.)	
A42	-"-		B42	-"-	
A41	-"-		B41	-"-	
A40	-"-		B40	-"-	
A39	-"-		B39	-"-	
A38	-"-		B38	-"-	
A37	-"-		B37	-"-	
A36	-"-		B36	-"-	
A35	-"-		B35	-"-	
A34	Not used		B34	Not used	
A33	c32 Rear Magazine J1		B33	a32 Rear Magazine J1	
A32	c31 -"-		B32	a31 -"-	
A31	c30 -"-		B31	a30 -"-	
A30	c29 -"-		B30	a29 -"-	
A29	c28 -"-		B29	a28 -"-	
A28	c27 -"-		B28	a27 -"-	
A27	c26 -"-		B27	a26 -"-	
A26	c25 -"-		B26	a25 -"-	
A25	c24 -"-		B25	a24 -"-	
A24	c23 -"-		B24	a23 -"-	
A23	c22 -"-		B23	a22 -"-	
A22	c21 -"-	standard	B22	a21 -"-	standard
A21	c20 -"-	64p	B21	a20 -"-	64p
A20	c19 -"-	flatcable	B20	a19 -"-	flatcable
A19	c18 -"-	connection	B19	a18 -"-	connection
A18	c17 -"-		B18	a17 -"-	
A17	c16 -"-		B17	a16 -"-	
A16	c15 -"-		B16	a15 -"-	
A15	c14 -"-		B15	a14 -"-	
A14	c13 -"-		B14	a13 -"-	
A13	c12 -"-		B13	a12 -"-	
A12	c11 -"-		B12	a11 -"-	
A11	c10 -"-		B11	a10 -"-	
A10	c9 -"-		B10	a9 -"-	
A9	c8 -"-		B9	a8 -"-	
A8	c7 -"-		B8	a7 -"-	
A7	c6 -"-		B7	a6 -"-	
A6	c5 -"-		B6	a5 -"-	
A5	c4 -"-		B5	a4 -"-	
A4	c3 -"-		B4	a3 -"-	
A4	c2 -"-		B3	a2 -"-	
A2	c1 -"-		B2	a1 -"-	
A1	Not used		B1	Not used	

Table 3.3.1a
Front Module (J3) to rear module (J1)
Standard Interface Connection

Connectors for Mains power to the CU-Crate are located at the rear of the crate in either the left and/or the right side of the crate.
Internally in the CU-Crate the AC power is connected to the combined power supply AC/DC connectors located at each side of the crate in the interface connector position.
The DC power outputs from the power supplies are connected to the Data Bus A and Data Bus B by means of discrete wiring terminated in power bus bars at the rear of the motherboards.

When redundant systems are implemented, i.e. independent DC power to each of the two databusses, the left most power module supplies data bus A while the right most power module supplies data bus B (seen from front of the CU).

In dualized systems, the voltages (+5V, +/- 12) are increased by 0,5 V, to allow for voltage drop in the fail safe power circuits included in the modules, by means of two jumpers installed in the AC/DC connector.
The power distribution within the CU crate and corresponding connector specifications are found in figures overleaf (3.3.1-3,-4a,b,c,d).

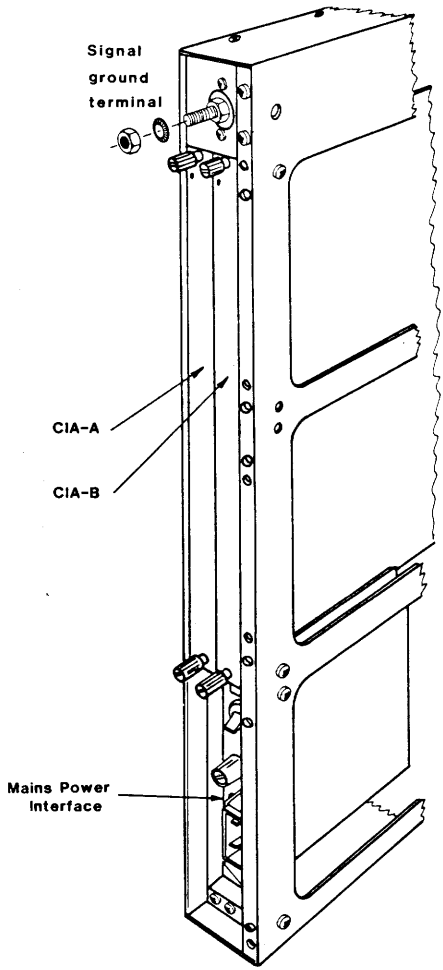
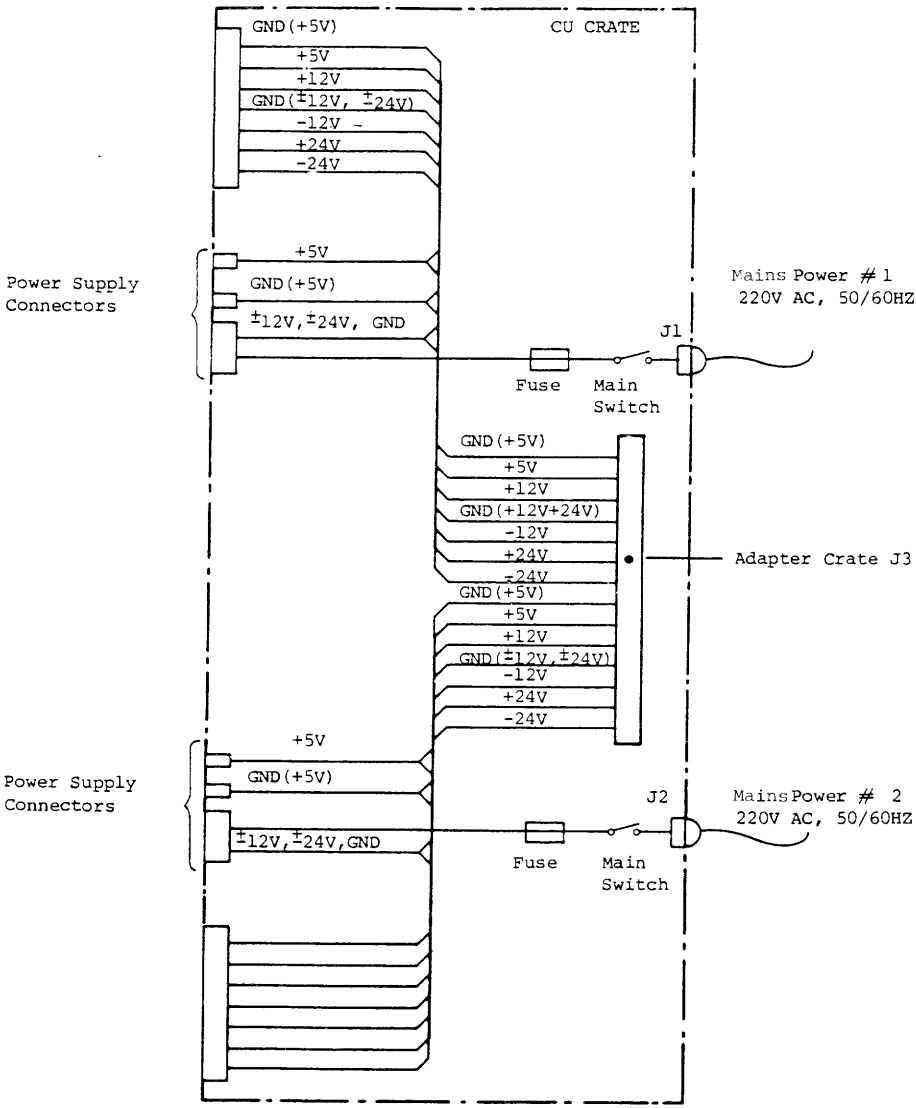


Figure 3.3.1-3



Note: If non dualized systems only mains power # 1 and the corresponding DC powers (Data Bus A, Adapter Crate J3) are available.

Figure 3.3.1-4a
CU CRATE
POWER DISTRIBUTION

Data Bus A and
Data Bus B (DC power pins)

Pin no.	Signal
A43,A42,B43,B42	GND(+5V)
A41,A40,B41,B40	+ 5V*
A14,B14	+12V*
A13,B13	GND(+/-12V,+/-24V)
A12,B12	-12V*
A11	+24V
B11	-24V

*in crates with dualized buses the nominal value of these voltages is 0.5V higher than in single bus crates.

Rear Magazin
(DC power pins)

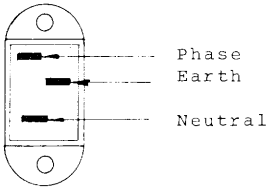
Pin no.	Signal
a32,a31,a30,c32,c31,c30	5V*
a29,a28,c29,c28	GND (5V)
a27,c27	+12V*
a26,c26	GND(+/-12V)
a25,c25	-12V*
a24	+24V
a23	GND(+/-24V)
a22	-24V
a19,a18,a17,c19,c18,c17	5V*
a16,a15,c16,c15	GND(5V)
a14,c14	+12V*
a13,c13	GND(+/-12V)
a12,c12	-12V*
c24	+24V
c23	GND(+/-24V)
c22	-24V

*in crates with dualized buses the nominal value of these voltages are 0.5V higher than in single bus crates
**in single bus crates only source A is available

Figure 3.3.1-4b
CU-CRATE
Power Distribution

Mains Power Connectors
[J1], [J2]

CU crate rear view



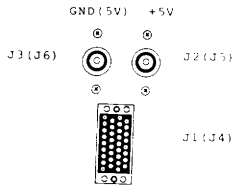
Connector type: Schaffner
FN 322Z

Input: 220V+/-
50/60Hz+/-

AC/DC POWER CONNECTORS

Power Supply Connectors
J1, J2, J3 and J4, J5, J6

Seen from front of CU Crate

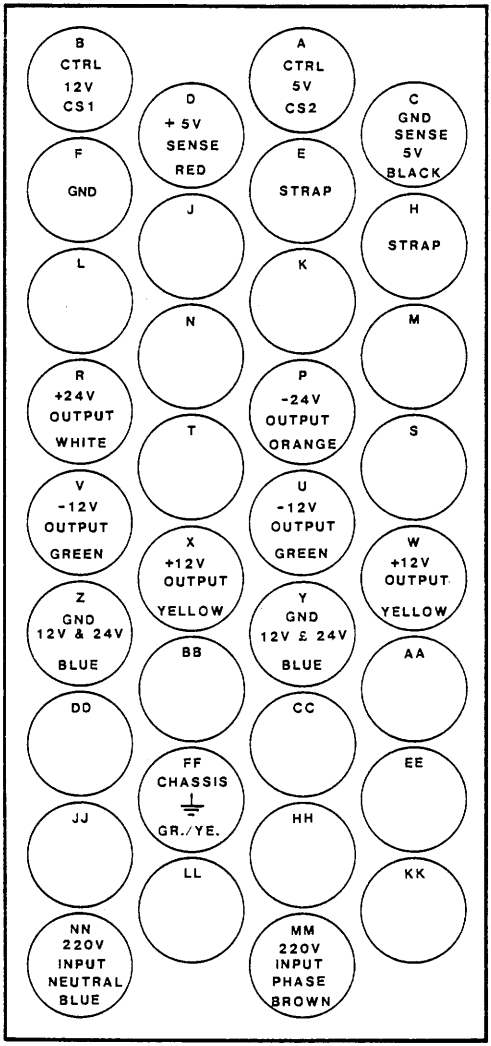


Connector Types:

J2,J3,J5 and J6: ODV 1.5.112-20100
J1 and J4 : AMP 201357-2

Pin layout for J1(J4)
shown in fig. 3.3.1-4d

Figure 3.3.1-4c
CU Crate
Power Distribution



WIRE SIDE VIEW

Note: CU-Crates with Dualized Busses (A and B)

Jumper between pin
F and H to give the
following voltages:
5V output becomes +5.7V
+12V output becomes +12.7V
-12V output becomes -12.7V

Figure 3.3.1-4d
CU CRATE
Power Distribution

3.3.2 CU-Front Modules

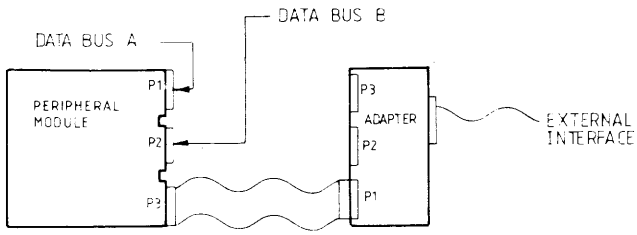
CU-Front module types include RAM, Peripheral modules, Maintenance and Configuration Processor (MCP) modules and power supplies. A full range of Peripheral modules such as mainframe interfaces, terminal interfaces, communication interfaces, and peripheral storage interfaces are available as part of the CR80 standard product line, with operational characteristics given in the datasheets. The general specifications of CU-Front modules are given below, while the data bus A and B electrical interfaces are described separately later in the chapter.

Peripheral modules are inserted from the front of the CU and interfaced to the external lines via an adapter.

Both dual data bus (A&B) modules (failure tolerant systems) and single data bus modules (single systems) are available. The dual-bused modules operate with higher DC supply voltage than the single-bused versions to allow for the voltage drop in the fail-safe power OR-ing circuit implemented in these modules.

The general layout and mechanical specifications are as defined in the previous section 3.2.2 for the PU front modules.

Figure 3.3.2-1
Pheripheral Module Interfaces



Note: In non failure tolerant systems, only Data Bus A interface is implemented

3.3.3 CU-Adapter Modules

Different types of Adapter modules are available, corresponding to the Peripheral Modules, to which they interface the external connections. (Refer to the datasheets for detailed information).
The adapter modules are inserted from the rear of the CU crate and the location within the crate is determined by the crate type and the location of its corresponding Peripheral module.
The CU-Adapter Modules connect to both the Adapter Crate, J3 and J2 backplanes (sections 3.9 and 3.10) for Power Distribution, Configuration Control and N+1 Adapter I/O line to spare Peripheral Module connection.

The general layout and mechanical specifications for CU Adapter Modules are identical to those for the PU Adapter Modules, as described in section 3.2.3.

3.3.4 CU-Bus Termination Modules

The bus termination modules are used for termination of the data transfer buses Data Bus A and Data Bus B when required by DCU crate configuration. In some of the standard crates the modules are not required because the Data channel interface modules (CIA- A and B) include the termination circuit.

The bus termination modules used in the CU are identical to the modules used in the PU, (refer to section 3.2.4 for the specifications), except that the input power is fed through a diode in dual bused (A&B) systems to reduce the higher supply voltages. The module is inserted from the rear of crate and the location within the crate is determined by the selected CU crate.

3.3.5 CU-Data Channel Interface Modules (CIA-A and B)

The Data channel interface modules CIA-A and CIA-B perform the CU's interface from data bus A and data bus B to PE-data channels; a bus termination circuit is included. In non-dualized systems only CIA-A is used, while both modules are used in failure tolerant systems. The modules include a diode in the input power line to give a voltage drop (0.5V) in dual-bused systems.

The modules are inserted from the rear of the crate in the positions defined for the selected crate type.

The mechanical specifications for the CIA-A and CIA-B modules are given in figure 3.3.5-1 overleaf. The data bus electrical interface complies with the Databus A and B, and Datachannel specifications treated separately later in the chapter (sections 3.6 and 3.7).

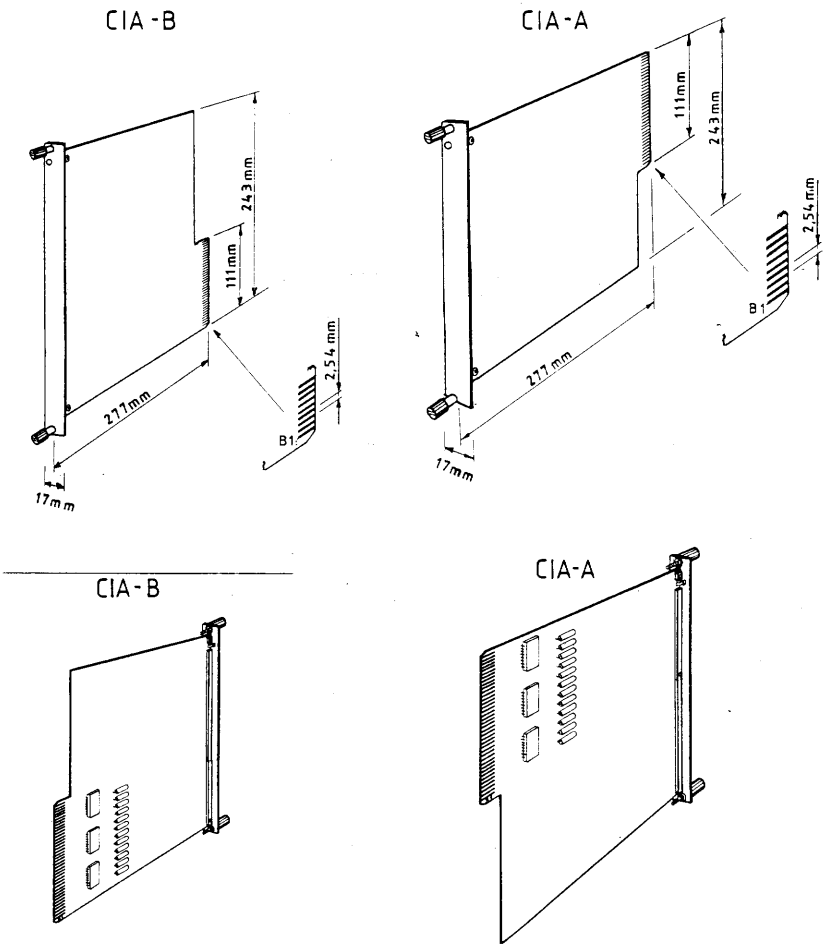


FIGURE 3.3.5-1
CIA-A and CIA-B
Mechanical Specification

3.4 P-Bus Specification

The P-bus is located within the PU-Crate in the front magazine as defined in figure 3.4-1 below.

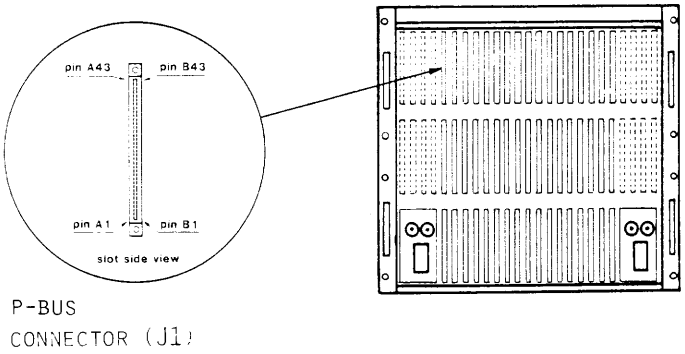


Figure 3.4-1
PU-Crate, Front View

The P-bus is a multilayer printed circuit board (motherboard) equipped with connectors (2x43 pin) for the module edge connector. The number of connector positions available on the bus depends on the selected crate type as detailed in the datasheets. The bus is terminated by Bus Termination modules in one or both ends, determined by the bus length.

The motherboard provides a parallel bus structure, meaning that the bus does not put any restrictions on the locations of the interfaced modules. The bus is (as shown in figure 3.4-2) overleaf equipped with connectors on both sides, the front connectors are for insertion of front modules , while the rear connectors are for interfacing of bus termination modules. Bus-bars are mounted on the rear of the motherboard to support DC power distribution to inserted modules.

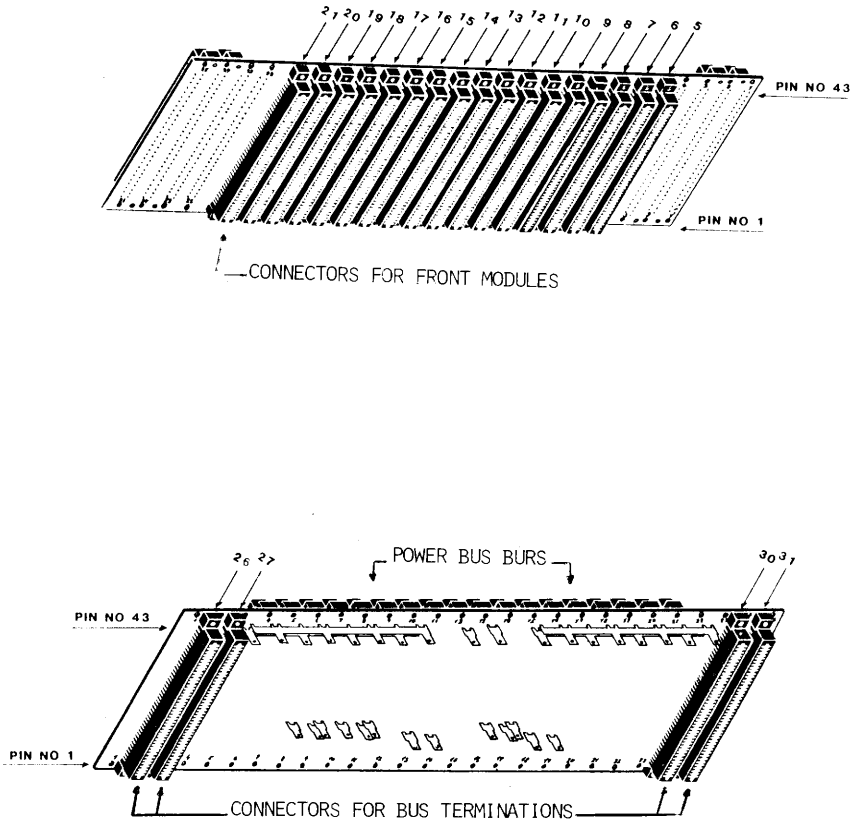


Figure 3.4-2
P-bus Motherboard

3.4.1 Physical Interface

Both the front mounted and rear mounted connectors on the motherboard have the interface defined below:

P-Bus Connector Pin Lay-out
Rear View

	B		A	
	gnd	43	gnd	
	gnd	42	gnd	
	+5V	41	+5V	
	+5V	40	+5V	
	gnd	39	gnd	
	gnd	38	Ø2 ← 8 MHz	
	gnd	37	AE(L)	
	AD19	36	AD18	
	gnd	35	Ø1 ← 1 MHz	
	gnd	34	R/W(L/H)	
	gnd	33	INR(L)	
	gnd	32	INA(H)	
	gnd	31	TRQ(L)	
	gnd	30	RS(L)	
	MC(H)	29	BD(L)	
	LP	28	UP	
	gnd	27	gnd	
	LS1	26	LS0	
	AD17	25	AD16	
	AD15	24	AD14	
	AD13	23	AD12	
	AD11	22	AD10	
	AD 9	21	AD 8	
ADDRESS (AD)	gnd	20	gnd	ADDRESS (AD)
	AD 7	19	AD 6	
	AD 5	18	AD 4	
	AD 3	17	AD 2	
	AD 1	16	AD 0	
	gnd	15	gnd	
	+12V	14	+12V	
	gnd	13	gnd	
	-12V	12	-12V	
	24V	11	+24V	
	gnd	10	gnd	
	DA15	9	DA14	
	DA13	8	DA12	
	DA11	7	DA10	
	DA 9	6	DA 8	
	DA 7	5	DA 6	
DATA (DA)	DA 5	4	DA 4	DATA (DA)
	DA 3	3	DA 2	
	DA 1	2	DA 0	
	gnd	1	gnd	

3.4.2 Electrical Interface

The electrical specification for modules connected to the P-Bus is given below.

Power Lines

Pin No.	Description	
A11	+24V	+0V +0.25V
B11	-24V	-0V -0.25V
A12, B12	-12V	-0V -0.25V
A13, B13	GND ground for +/-12V and +/-24V	
A14, B14	+12V	+0V +0.25V
A40, A41 B40, B43	+5V	+0V +0.25V
A42, A43 B42, B43	GND ground for +5V	

Maximum power consumption per connector:

+24V, -24V:	1A
+12V, -12V:	4A
+5V:	10A

Signal Lines

The levels for all the Bus signals are normal TTL logic levels, i.e.:

	Receivers	Drivers
High level:	$2.0V<V_H<5.0V$	$2.4V<V_H<5.0V$
Low level:	$0V<V_L<0.8V$	$0V<V_<0.5V$

The load on the different signal lines for one module are as specified in the following (all currents are numeric values).

Data Lines & Address Lines

DA0 - DA15, UP & LP, AD0 - AD(19), R/W(L/H), LS0 & LS1.

3-state signals with the following requirements:

Drivers:	I_{OH}	$\geq$	5.2 mA
	I_{OL}	$\geq$	16.0 mA
	I_O off	$\leq$	100.0 uA
Receivers:	I_{IH}	$\leq$	100.0 uA
	I_{IL}	$\leq$	0.5 mA
(Note 1)	I_{IL}	$\leq$	2.0 mA

Note 1: This low level input current is only allowed when the module is addressed.

Master Clear & Interrupt Acknowledge

MC(H), INA(H)

Open collector with the following requirements:

Driver:	I_{OL}	$\geq$	60.0	mA
	I_{OH}	$\leq$	250.0	mA
Receiver	I_{IH}	$\leq$	100.0	uA
Schmitt Trigger:	I_{IL}	$\leq$	0.5	mA

Transfer Request & Address Enable

TRQ(L), AE(L)

Open collector or 3-state signal with the following requirements:

Driver:	I_{OL}	$\geq$	80.0	mA
	I_{OH}	$\leq$	250.0	uA
Receiver	I_{IH}	$\leq$	100.0	uA
Schmitt Trigger:	I_{IL}	$\leq$	0.5	mA
(1)	I_{IL}	$\leq$	2.0	mA

Note 1: I_{IL} 2 mA is only allowed for the Authority Controller & RAM modules with a memory area of 32K words or more.

Clock Signals

Ø1 og Ø2

Driver:	I_{OL}	≥	120.0 mA
	I_{OH}	≥	80.0 mA
Receiver	I_{IL}	≤	2.0 mA
Schmitt Trigger:	I_{IL}	≤	100.0 uA

Block Address Disable

BD(L)

Open collector or 3-state signal with the following requirements:

Driver:	I_{OL}	≥	16.0 mA
	I_{OH}	≤	250.0 uA
Receiver	I_{IH}	≤	100.0 uA
Schmitt Trigger:	I_{IL}	≤	0.5 mA

Response & Interrupt Request

RS(L), INR(L)

Open collector signal with the following requirements:

Driver:	I_{OL}	≥	60.0 mA
	I_{OH}	≤	250.0 uA
Receiver	I_{IH}	≤	100.0 uA
Schmitt Trigger:	I_{IL}	≤	2.0 mA

3.4.3 Bus Termination

The P-Bus signal lines are terminated at one or both ends of the motherboard by a "Bus Termination Module".

The signal lines (group 1):

DA (15:0), UP, LP
AD (19:0),
LS1, LS0,
R/W

are each terminated in the circuit shown in figure 3.4.3-1

The signal lines (group 2):

BD, MC, TRQ, RS, INA, INR, AE

are all open collector lines and are each terminated in the circuit shown in figure 3.4.3-2

The clock signals (group 3) Ø1 and Ø2 are terminated in the circuit shown in figure 3.4.3-3

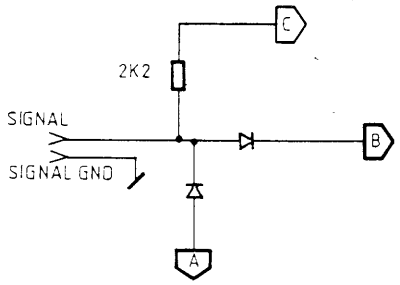


Figure 3.4.3-1
Termination for each signal line in group 1 (Diodes = Silicon)

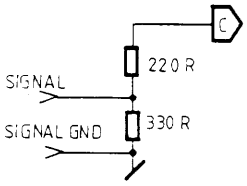


Figure 3.4.3-2
Termination for each signal line in group 2.

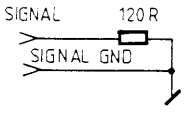


Figure 3.4.3-3
Termination for each signal line in group 3.

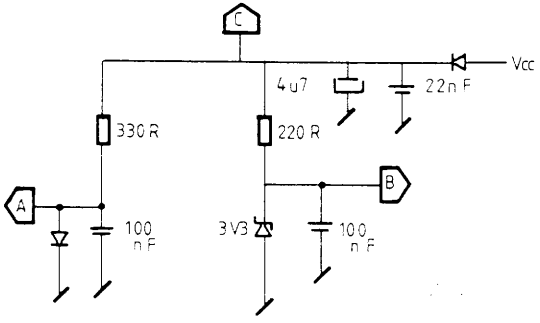


Figure 3.4.3-4
Common circuit for signal line terminations. (Diodes = Silicon).

3.4.4 Functional and Timing Interface

The P-Bus constitutes the media for exchange of information between the modules connected to the bus as well for the interrupt and timing signals.

Data and instruction transfer is performed under control of a CPU or DMA module by means of the handshaking signals TRQ(L), AE(L) and RS(L). The CPU or DMA module transmits the location address to the bus address lines AD0 - AD19, (AD(19:18) are outputs of the MAP only), LS0, LS1 and R/W(L/H) and transmits/receives the addressed information (16-bit word/8-bit byte).

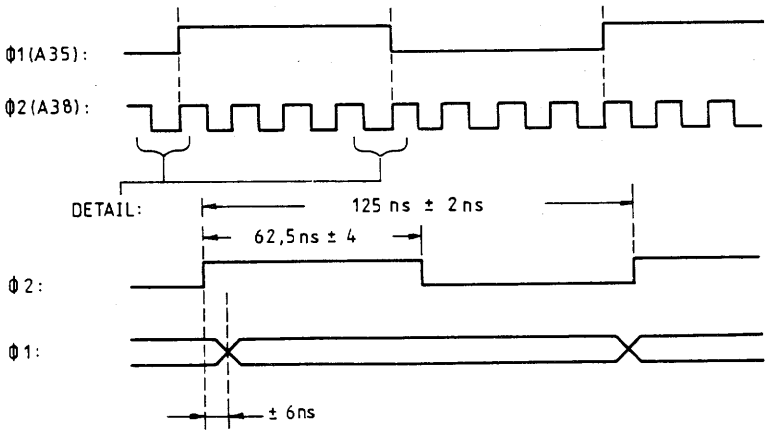
The transfer is asynchronous so that modules with different response times can be connected to the bus.

The interrupt code is transmitted in serial form from the modules in synchronism with the master timing signals.

In the following the signals on the Bus are defined:

a. Master Timing Ø1 (A35) and Ø2 (A38)

Two clock signals, Ø1 and Ø2, are available on the bus for interrupt transmission and for internal timing in the modules connected to the Bus. The timing diagram below (figure 3.4.4-1) defines the signals.



Frequency stability of Ø2: 8 MHz ± 2 x 10^{-2} %

b. Interrupt Signals INA(H) (A32) and INR(L) (A33)

These signals are not used with P-Bus in memory mapped systems (CR80 MAXIM and FATOM).

The CR80 interrupt system is based upon serial transmission of the interrupt code from the interrupting module.

Two of the Bus lines are used for interrupt transmission:

INR (Interrupt Request) & INA (Interrupt Acknowledge).

The interrupt code consists of 8 bits, of which 2 are priority bits and the remaining 6 bits are the module number. The 8 bits are transmitted within one Ø1 (1MHz) cycle, each bit synchronized to the Ø2 (8MHz) clock. The interrupt code is transmitted on the INR line (see figures 3.4.4-2,3 overleaf).

INR is an open collector line so that all the connected modules can transmit on the same line, which means that if one module is transmitting a "L" and another a "H", the line will contain the "L".

INA is an open collector line too, and it is driven from the termination modules. The contents of INA is INR inverted. Each of the interrupting modules compare for each of the eight bits the contents of INA with the bits it is transmitting to INR. If the comparison does not match which means that a module with a higher interrupt code is transmitting at the same time, the module disables the transfer of its code to INR until the next 1MHz period.

In this way the contents of INR will be unique and correspond to the highest priority interrupt transmitted during that period. The module which detects this situation has got its interrupt acknowledge and will stop the interrupt sending.

The diagram below illustrates the function and the connection of the interrupt circuit:

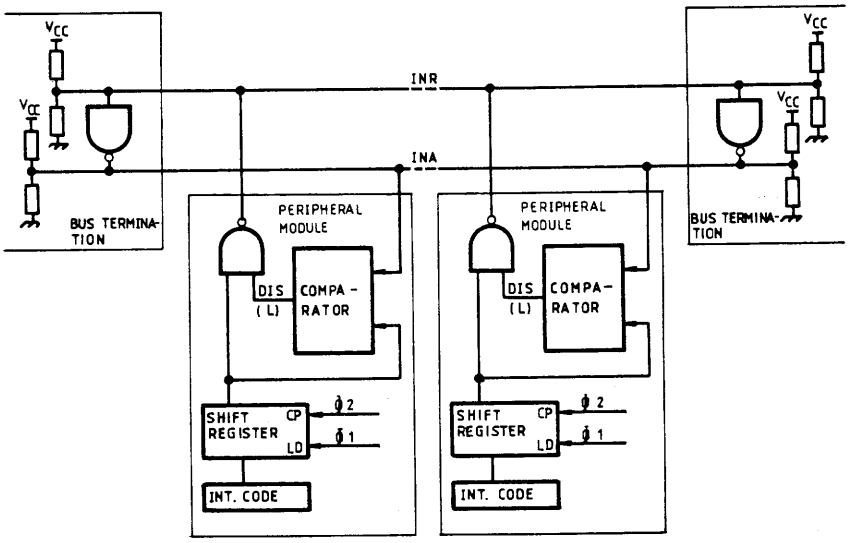


Figure 3.4.4-2
INTERRUPT CIRCUIT

The 8-bit interrupt code consists of two parts: a 2-bit priority corresponding to four levels and a 6-bit module address corresponding to 64 modules which can operate with interrupt in each crate. Two of the 64 module addresses are restricted:

The address 00 wil not be recognized because it corresponds the idle state of the interrupt lines. The address 63 is not allowed, since it is used by the interrupt receiving module when its internal interrupt queue is full and it therefore has to queue the interrupts directly on the interrupt lines (INR).

When a module will transmit an interrupt, it starts with priority bits and continues with the address bits, as shown in the timing diagram below.

Interrupt, Timing Diagram

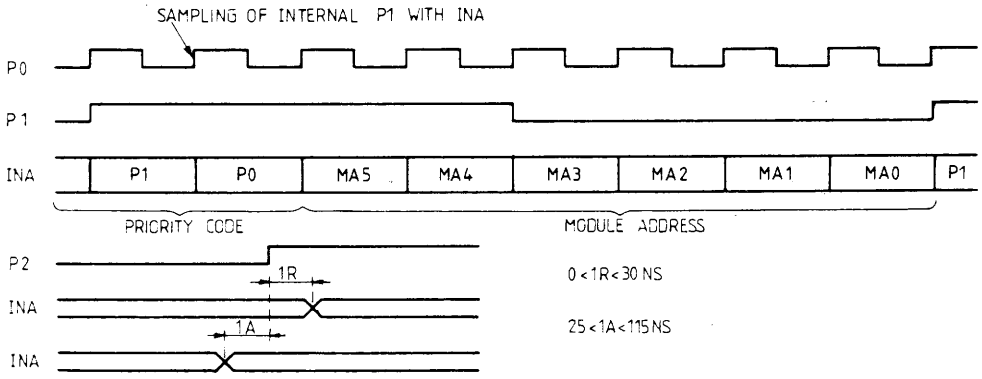


Fig. 3.4.4-3

c. Master Clear MC(H) (B29)

The Master Clear signal is used for resetting the modules to a well-defined state when power is switched on or it is issued from the MAP upon command. The signal sequence during power up and programmed clear is shown in figure below.

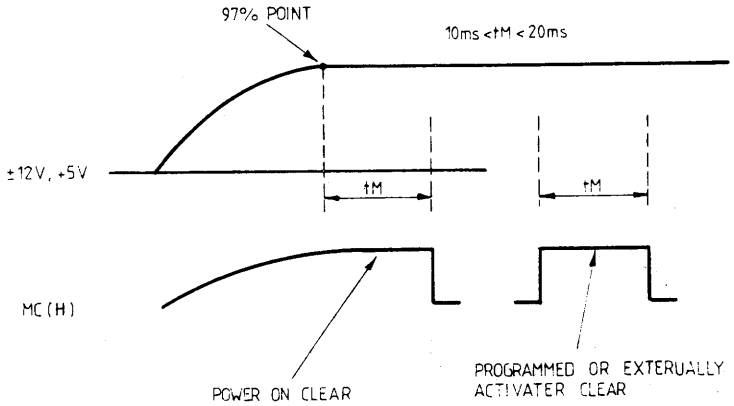


Figure 3.4.4-4

When the modules recognize a Master Clear they have to reset all their internal functions to the start-up condition, and, if implemented, initiate their built-in test (BIT).

The RAM modules do not use the Master Clear signal, but only the power up sequence for resetting.

d) Data Lines DA0-DA15 (A2-B9), LP (28), UP (A28)

The information communicated on the parallel bus is 16 bit + 2 parity bit. There is one parity bit for each of the two 8 bit bytes; UP is parity for the upper byte (DA8-DA15) and LP for the lower byte (DA0-DA7).

During of byte read, both parity bits have to be correct independent of which byte is addressed. As far as memory modules are concerned, all read operations are word-read-operations, leaving it to the address sourcing module to select the relevant part of the data on the bus to be processed. Thus, a memory module does not distinguish between word and byte reads, but always outputs the full word containing the addressed data to the bus.

Odd parity is used for both bytes. DA0 is LSB and DA15 is MSB in the data word.

The data bus is bidirectional, the source being specified by the address lines, LS1, LS0 and R/W signal.

With R/W(L/H) = L, a read operation is performed, meaning that the address sourcing module CPU or DMA receives data from the addressed module.

With R/W(L/H) = H, a write operation is performed, meaning that the address sourcing module (CPU or DMA) transmits data to the data bus.

For the timing specification for the data bus lines, refer to the paragraph on Bus Communication (3.4.4g).

Address destination modules must be ready for operation immediately after falling edge of MC, i.e. they must respond

- either according to timing specs. figs. 3.4.4-8 or 3.4.4-9
- or by not responding at all (if internal MC-operation has not terminated "build in test"), causing a timeout condition to arise.

e) ADDRESS LINES

AD(19:0) (A16 - B19, A21 - B25, A36, B36)
LSO (A26)
LSI (B26)
R/W (L/H) (A34)

The address bus consists of 20 lines, permitting addressing of up to 1 megawords (2 megabytes) and 64 module numbers.

Tables 3.4.4a-c gives a summary of the bus addressing scheme. For timing specifications, refer to "Bus Communication", section 3.4.4g.

TABLE 3.4.4a

ADDRESSING SPECIFICATIONS FOR ADDRESS SOURCING MODULE IN PROCESSOR UNIT CONTAINING A MAP MODULE

LOGICAL ADDRESS FROM "ADDRESS SOURCING MODULE"

R/W	LS1, LS0	AD17	AD16	AD (15:12)	AD (11:6)	AD (5:0)	OPERATION	ADDRESS DESTINATION MODULE
0	00	0	0	CRATE ADDRESS	CM	MA	Privileged READ from Moduel "MA" in Crate "CRATE ADDRESS". Address field "CM" is interpreted by module.	MODULE NO
1	00	0	0	CRATE ADDRESS	CM	MA	Privileged WRITE to Module "MA" in Crate "CRATE ADDRESS". Address field "CM" is interpreted by module.	MODULE NO.
0	00	1	AD(16:10) = LOGICAL PAGE		AD (9:0 = DON'T CARES		The "ACCESS Control Bits" of the logical page designated by AD (16:0) are read from the MAP module.	MAP MODULE
1	00	1					TBD	TBD
0	00	0	1				READ from location in MAP module	MAP
1	00	0	1				WRITE to location in MAP module	MAP
0	01 or 10 or 11	*)	LOGICAL ADDRESS				Read word in location designated by AD (16:0)	MEMORY
1	01	*)	LOGICAL ADDRESS				Write lower byte in location designated by AD (16:0)	MEMORY
1	10	*)	LOGICAL ADDRESS				Write upper byte in location designated by AD (16:0)	MEMORY
1	11	*)	LOGICAL ADDRESS					

*) With AD17 = 0, the access to the memory location is to subject to the control of the "access control code" in the map
AD17 = 1, corresponds to "full access"

For CPU's, AD17 is a replica of the YSSTEM/USER - STATE - bit in the Process Status Word. (1 - SYSTEM STATE)

Table 3.4.4b

Adressing specifications for I/O - module.

R/W (L/H-O/I)	LSI, LSO (L/H - O/I)	AD(19:18) (L/H-O/I)	AD (17)	AD 16	AD (15:12)	AD (11:6)	AD (5:0)	Operation
0	00	X	X	X	X	CM	MA	Privileged read from module "MA". Addressfield "CM" interpreted by module.
1	00	X	X	X	X	CM	MA	Privileged write to module "MA". Addressfield "CM" interpreted by module.

Table 3.4.4c

Addressing specifications for memory module.

19 ← → 0

0	01 or 10 or 11	PHYSICAL ADDRESS	Read word in location designated by AD (19:0). Output full word parity to bus.
1	01	PHYSICAL ADDRESS	Write byte on DA (7:0) into lower byte of location designated by AD (19:0)
1	10	PHYSICAL ADDRESS	Write byte on DA (15:8) into upper byte of location designated by AD (19:0)
1	11	PHYSICAL ADDRESS	Write word on DA (15:0) into word location designated by AD (19:0)

Note: With no map module in processing unit, AD (19:18) are high.

f) Transfer Control Signals

TRQ(L) (A31), AE(L) (A37)
RS(L) (A30) & BD(L) (A29)

Four signals on the Processor Bus are used for controlling of the information transfer on the bus. Three of the signals TRQ(L), AE(L), BD(L) are used as address validity control during read operation, and as address and data validity control during write operation.

The signal TRQ(L) is generated by the CPU or DMA module specifying that the address bus except AD19 and AD18 is valid.

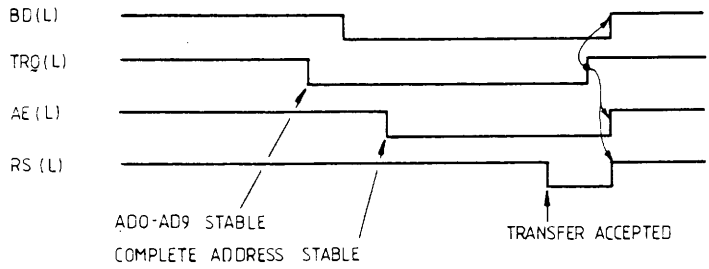
BD(L) is transmitted from the MAP module to the CPU's and DMA to switch off address lines AD10-AD17.

AE(L) is issued from the MAP as validity control for the complete modified address field.

RS(L) is the response signal from the address destination module. During read operations, RS(L) = L indicates that data are being transmitted. During write-operations, the falling edge of RS(L) indicates, that data has been accepted.

Figure 3.4.4-5

Bus Handshaking Procedure



g) BUS COMMUNICATION

In this paragraph, the timing specs. for the following modules are presented:

- Address sourcing modules (CPU, DMA)
- Address destination modules
- The MAP module

The bus timing diagrams are designed to allow use of long busses, as reception of signals is not specified relative to the clock. This does not, however, apply to the interrupt signalling lines INA and INR, which are transmitted and received on positive edges of clock Ø2.
Thus, delay in the INA/INR circuit and signal lines sets a limit to the maximum bus length. This limitation does not apply to systems (or subsystems) not using the INA/INR - lines.

In the timing specs., the following notations are used.

- Numbers in square brackets [e.g. $0<t<40$] are absolute maximum ratings for the propagation delay from the reference signal, to an output measured at the bus).

- Numbers in curled brackets {e.g. $t>50$} specify guaranteed worst case delay between two input signals, assuming a maximum bus length of 2 m (corresponding to a skew of $\leq$ 5 ns).

Figure 3.4.4-6

TIMING SPECS. FOR ADDRESS SOURCING MODULE

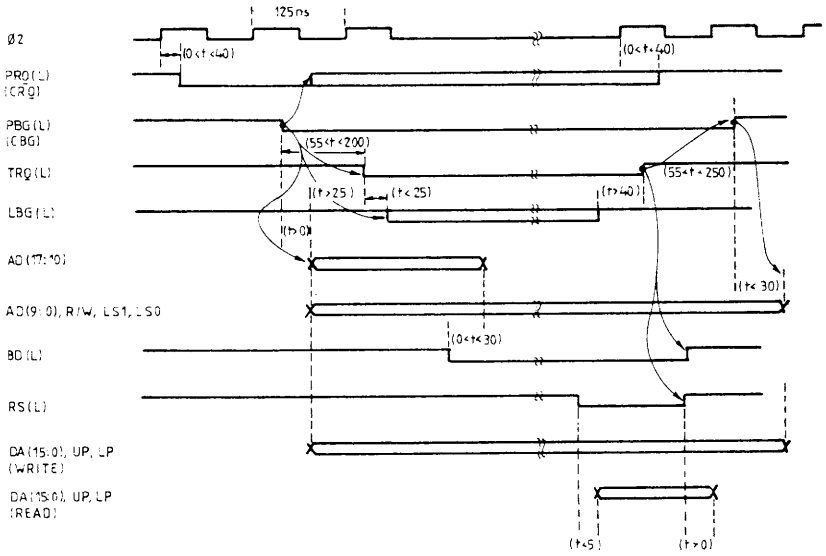


Figure 3.4.4-7

TIMING SPECS. FOR MAP MODULE

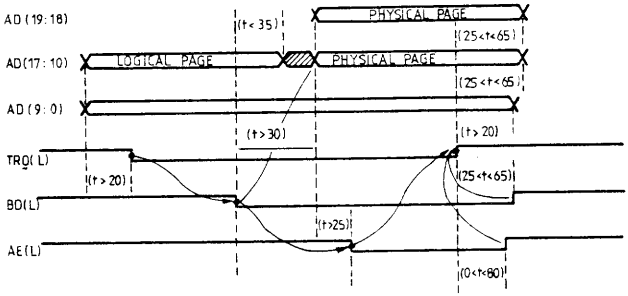


Figure 3.4.4-8

TIMING SPECS. FOR ADDRESS DESTINATION MODULE, WRITE TIMING

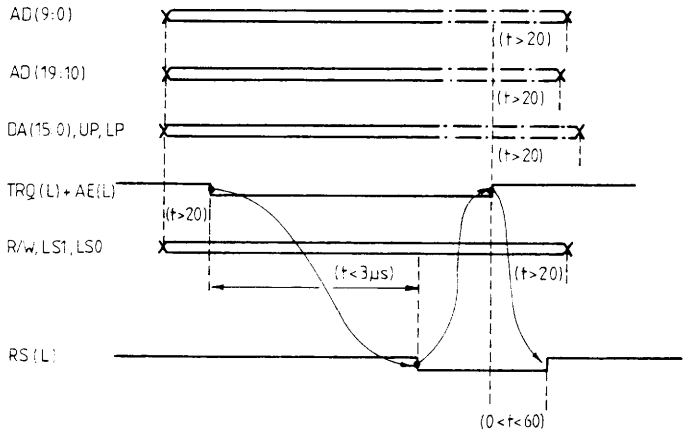
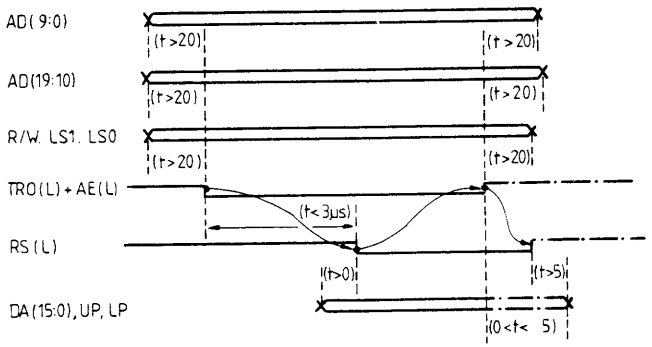


Figure 3.4.4-9

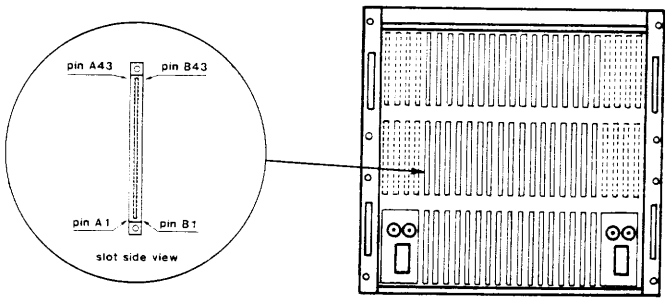
TIMING SPECS. FOR ADDRESS DESTINATION MODULE, READ TIMING



3.5 C-Bus Specification

The C-Bus is located within the PU-Crate in the front magazine as shown below.

Figure 3.5-1
PU-Crate Front View



C-Bus
Connector (J2)

The C-Bus is implemented identically to the P-Bus except for the location within the crate. Refer to section 3.4 P-Bus for detailed specifications.

3.6 Data Bus A and Data Bus B

The transfer buses for the CU (Data Bus A and Data Bus B) are implemented as multilayer printed circuit boards (motherboards) equipped with 86 pin connectors for interfacing to the modules as shown below. The length and number of connectors depends on the selected crate, type as described in the datasheets for CU-crates.

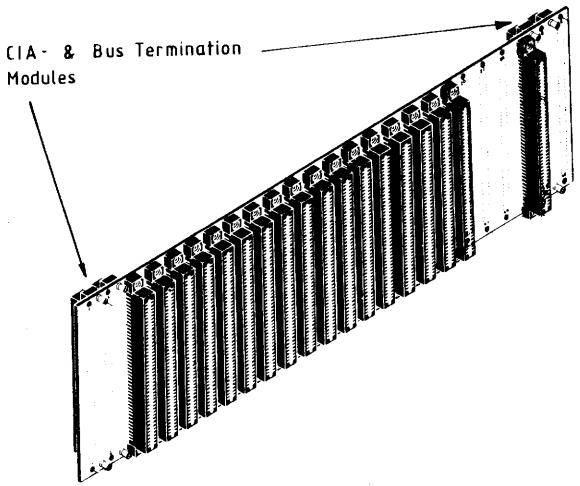


Figure 3.6-1
Data Bus A or Data Bus B Motherboard

The dual buses are implemented in dualized systems while only Data Bus A is available in non dualized CU crates. The physical location of the buses within the crate is defined in figure 3.3.1.
The pin layout and signal electrical and functional specifications are as defined for the P-Bus, (refer to section 3.4) with the following exceptions:

a) In dual bus crates the DC power is used as defined below:

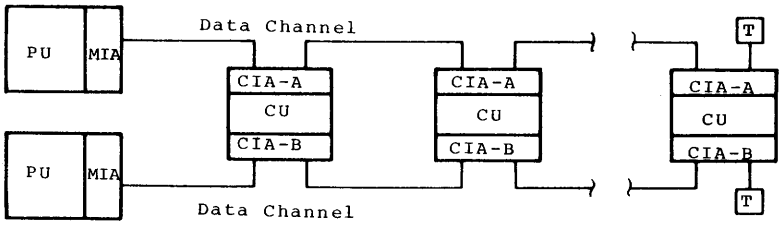
P-Bus, Data Bus A in non dualized bus CU crates	Data Bus A and Data Bus B in dual bus CU crates
5.2V	5.7V
12.2V	12.7V
-12.2V	-12.7V

b) In dual bus crates the master clear signal MC(H) must not be used by the connected dual bus modules.

3.7 CR80 Data Channel

The Data Channel is the Memory extension bus connecting a PU to a number of Data Buses in the attached CU's. It is controlled by the MAP-module via the MAP-adapter (MIA), which communicates with a daisy-chain of Channel I/F Adapters (CIA's) in the CU's.
Below a typical configuration is illustrated, and while the physical data channel and related interfaces are described in the following.

Figure 3.7-1
Typical configuration:



Different lenghts of data channel cables (25 twisted pair cable) are available in the CR80 product line (see datasheets).

3.7.1 Connector Specifications

The Data Channel is a 50-pole flat cable arranged in 25 twisted pairs with the following pin-lay-out:

SIGNAL		SIGNAL PIN	SIGNAL RETURN PIN
Information bit 0	IB0	1	2
Information bit 1	IB1	3	4
Information bit 2	IB2	5	6
Information bit 3	IB3	7	8
Information bit 4	IB4	9	10
Information bit 5	IB5	11	12
Information bit 6	IB6	13	14
Information bit 7	IB7	15	16
Information bit 8	IB8	17	18
Information bit 9	IB9	19	20
Information parity	IBP	21	22
		23	24
Address timing	AT	25	26
Data timing	DT	27	28
Data acknowledge	DA	29	30
		31	32
Interrupt data	ID	33	34
Interrupt timing	IT	35	36
Interrupt acknowledge	IA	37	38
		39	40
		41	42
		43	44
		45	46
		47	48
		49	50

Connector type

Flat cable: female connector
MIA/CIA: male header

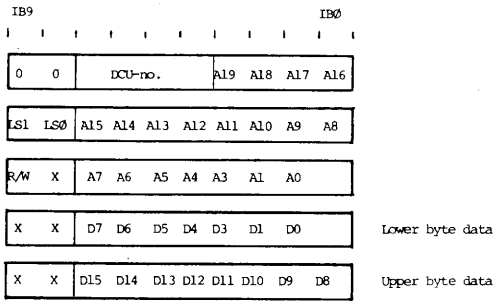
Maximum length from MIA to last CIA in chain: 50m.
The last CIA terminates the channel with a row of resistors instead of a continuing flat cable.

3.7.2 Operation

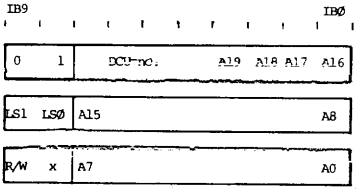
The Data Channel consists of two separate paths:

a) Information path
b) Interrupt path

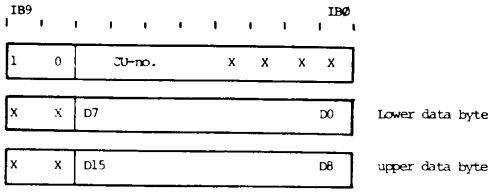
a) The information path is capable of transmitting one byte of data/address at a time. As the data word size is 16 bits and the maximum address field is 20 bits a normal transfer takes 5 cycles.
The control signals used on the Data Bus are transmitted on IB8 and IB9 as described:



In order to speed up sequential access the CIA has a register in which address information is stored. The register is loaded by the MIA with the following procedure:

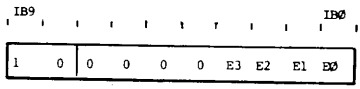


An access in the form of:



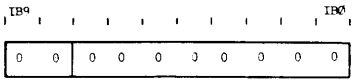
uses the stored address, which is incremented after each memory access.

If the CIA receives an address cycle with parity error it does not respond. Parity error in connection with data, however, results in an error message:



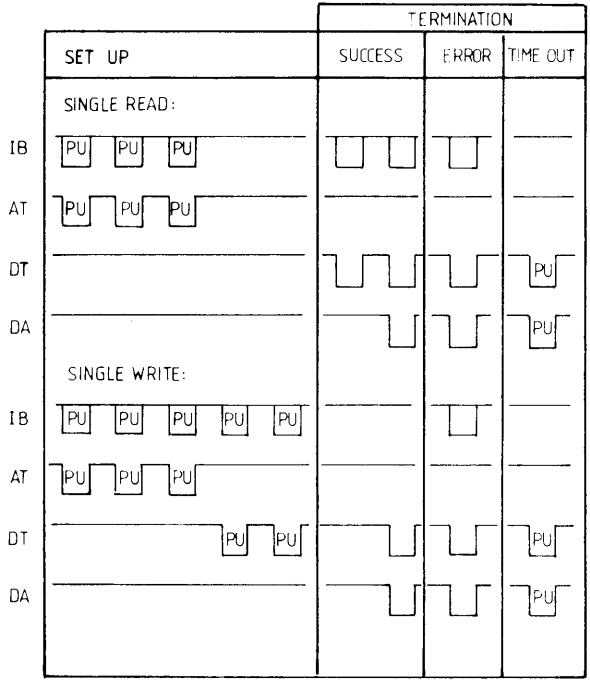
E3	E2	E1	E0	
1	0	0	0	Parity error detected on the Data Bus (read operation)
0	1	0	0	Parity error detected in one or both data cycles (write operation)
0	0	1	0	Stored address overflow. If A_7 -A_0 reaches FF further reduced accesses must be preceded by a new load operation.

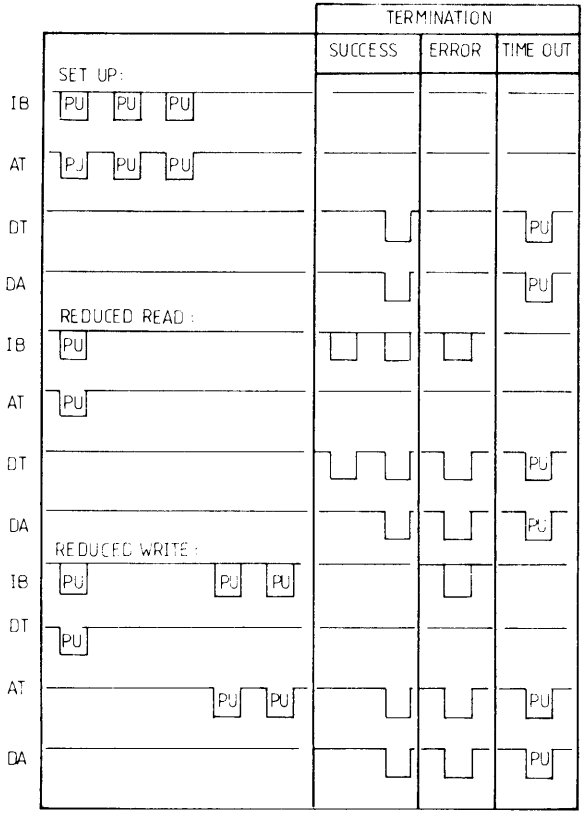
Write operations are terminated by an extra cycle so that the MIA can decide whether a transfer succeeded or not.



Address cycles are accompanied by the Address Timing signal (AT) and the Data Timing signal (DT) controls the other types of cycles (data, error , and acknowledge).

Timing diagrams, figure 3.7.2-1 overleaf, summarize the different transfer types.

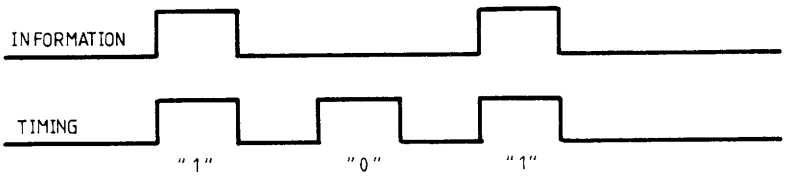

TERMINATION
SET UP
SUCCESS
ERROR
TIME OUT
SINGLE READ:
IB
AT
DT
DA
PU
SINGLE WRITE:
IB
AT
DT
DA
PU


TERMINATION
SUCCESS
ERROR
TIME OUT
SET UP:
IB
AT
DT
DA
PU
REDUCED READ:
IB
AT
DT
DA
PU
REDUCED WRITE:
IB
DT
AT
DA
PU

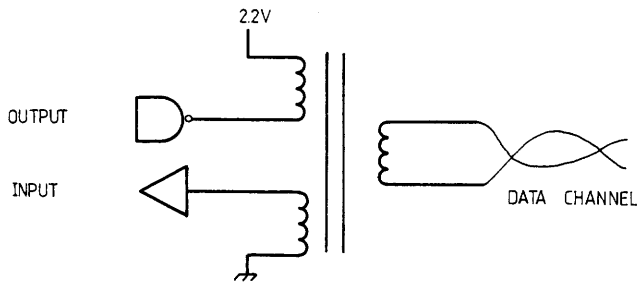
3.7.3 Datachannel Electrical and Timing Interface

The transmission is fully balanced by means of pulse-transformers. These provide galvanic separation, eliminating possible problems with ground currents between racks.

Basic transmission principle:



Nominal transmitter pulse height: 2.1 V
Nominal pulse width: 62.5 ns
Nominal minimum space width: 62,5 ns



3.8 Supra Bus

The Supra Bus is the communication media between the Processing Elements (PE's) in the CR80 system. The Suprabus consists of two coaxial twisted-wire cables of maximum length 50 meters.

As shown below in figure 3.8.1, one of them is the databus for serial interchange of data between any two of up to 16 connected Processing Elements (no. 0-15), the other being the respond bus for acknowledging to the transmitting device when transmission is initiated that the receiving device is ready to receive (receive buffer empty).

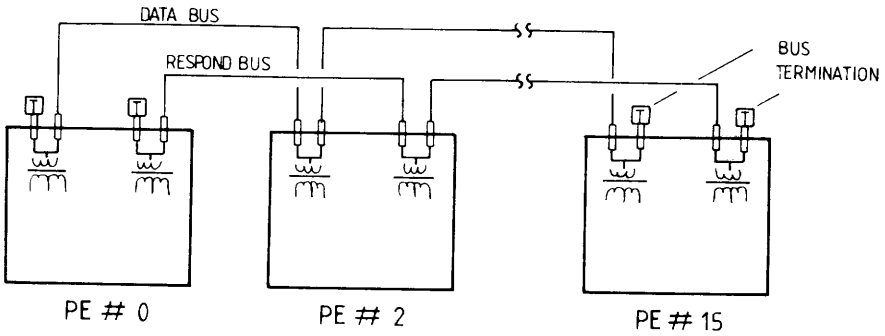
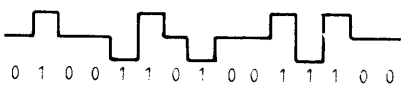


Figure 3.8-1
Typical Supra Bus Configuration

Transfer rate on both the data - and the respond bus is 16 Mbips utilizing Alternate Mark Inverted encoding:



Because no DC-component is associated with the AMI format, changeover between transmitters positioned at different places along the bus will not disturb the normal run in effect.
Devices are transformer-isolated from both buses and the buses are terminated with their characteristic impedance at bus ends.

Multiple Suprabuses may be installed between PE's for expanded throughput and redundancy.

Figure 3.8-1 shows the physical Suprabus and related interface Suprabus adapters (SBA).

Different length of Suprabus cables are available as part of the CR80 product line (see datasheets).

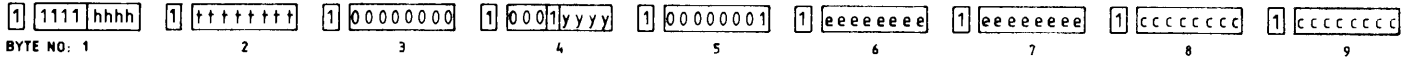
3.8.1 Transmission and Arbitration

The transmission format is shown in figure 3.8.2, overleaf. For detailed information on header content is referred to chapter 2.

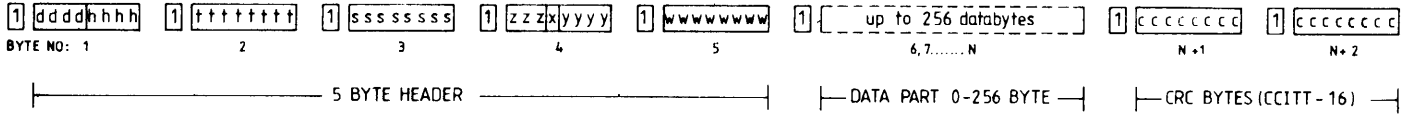
After a maximum of 4us from the beginning of transmission the transmitting device must receive a respond frame on the respondbus indicating that the receiving device is ready to receive, otherwise it will initialize. The correct respond-byte is a copy of the fourth byte in the transmitted frame (with preceeding one). If the respond-byte is correctly received, the transmission is continued until the complete frame has been sent out on the bus. If the respond-byte is in error, the device acts as if no respond were received.

DATABUS:

SIGNAL FRAME:

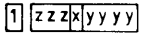


DATA FRAME:



1 : preceeding "one's"
d : datatype (4 bit)
h : destination Processing Element, PE (4 bit)
t : data stream number (8 bit)
s : communication protocol byte (8 bit)
z : cyclic sequence numbering of dataframes (3 bit)
x : dataframe contains data bytes: x = 1 / no data: x = 0 (1 bit)
y : source Processing Element, PE (4 bit)
w : number of following databytes - 1 (8 bit)
e : length of outgoing transfer (16 bit)
c : cyclic redundancy check word (16 bit)

RESPOND BUS:



z,x,y: content as for byte no 4 on the DATA bus

Fig. 3.8.2
TRANSMISSION FORMAT ON
DATA AND RESPOND BUS

3.8.1.1 Interface Specifications

The Suprabus connectors are located on the front panel of the SBA (as shown below) and are of the type: Suhner BNO serie 6 mm.

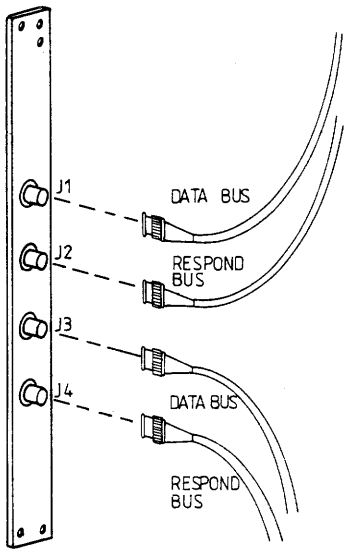


Figure 3.8.1-1 SBA front panel

Electrical Interface

Output amplitude:	2V peak (AMI)
Input amplitude:	>300mV peak

3.9 Adapter Crate, J3 position backplane

The Adapter Crate J3 position, backplane (see Table 3.9 a and Fig. 3.9-1) distributes power and configuration control signals from the MCP subsystem (Chapter 7) to Adapters in PU and CU Crates.

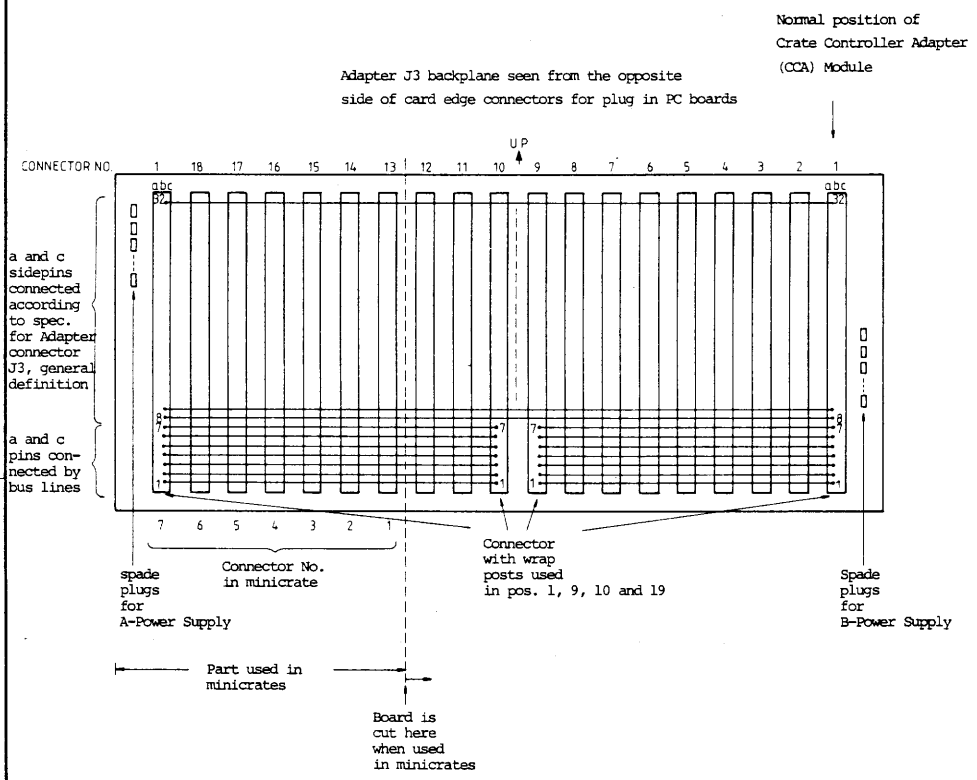
Table. 3.9 a Adapter Crate Backplane Connector J3

	not used		
	a	b	c
	+5V-A	32	+5V-A
	+5V-A	31	+5V-A
	+5V-A	30	+5V-A
	grd	29	grd
	grd	28	grd
	+12V-A	27	+12V-A
	grd	26	grd
	-12V-A	25	-12V-A
	+24V-A	24	+24V-B
	grd	23	grd
	-24V-A	22	-24V-B
	free1	21	free2
	PU adapter enable	20	PU adapter enable
	+5V-B	19	+5V-B
	+5V-B	18	+5V-B
	+5V-B	17	+5V-B
	grd	16	grd
	grd	15	grd
	+12V-B	14	+12V-B
	grd	13	grd
	-12V-B	12	-12V-B
	dev.3 on	11	dev. 4 on
	grd 3	10	grd 4
device switching	dev. 1 on$_1$	9	dev. 2 on$_2$
	grd$_1$	8	grd$_2$
	switch inhibit	7	grd^2
LTU bank switching	grd	6	grd
	A3	5	A3
	A2	4	A2
	A1	3	A1
	A0	2	A0
	switching enable	1	grd

see note

* In systems with single Power Supply, only the A bus power pins carry power.

Note: The lower part of the connector is reserved for the Crate Configuration Adapter (CCA) control outputs (see Chapter 7).



CHRISTIAN ROVSING A/S

Backplane in
Adapter Crate J3 position
Fig. 3.9-1

Scale	–	Approved		Issue	1
Date	80-10-12	E		Date	80-10-12
Drawn	ASM/MM			Approved	
Parts no		30-1143			
Print no		Sheet 1 of 1 sheets			

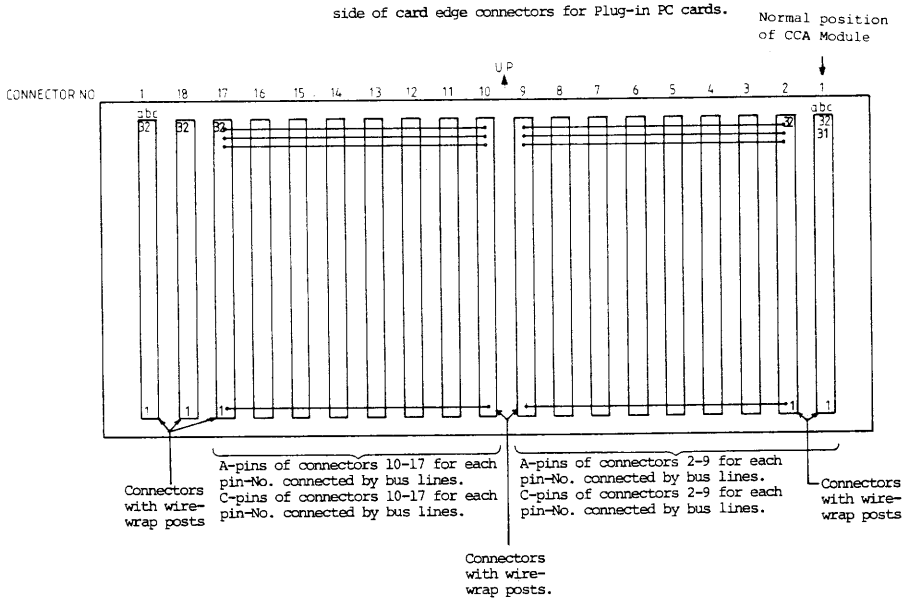
3.10 Adapter Crate, J2 Position Backplane (CU)

The Adapter Crate, J2 position backplane (see Fig. 3.10-1) provides the bus connection of communication lines to one or two spare LTUs with N+1 redundant LTU configurations in a CU-Crate.

For single LTU bank of more than 8+1 LTU's, J2 bus lines are continued by connecting corresponding pins, by wire-wrap, between connectors in position 9 and 10.

Please refer to chapter 7, section 7.7.2 for details on the CCA, J2 connection and to section 3.9 Fig. 3.9-1 for details on backplane cabling.

Adapter J2 backplane seen from opposite side of card edge connectors for Plug-in PC cards.



CHRISTIAN ROVSING A/S

Backplane in
Adapter Crate J2 position
Figure 3.10-1

–			1
80-10-12	E		80-10-12
ASM / MM			
	30-1142		

3.11 RACK Integration of PU and CU Crates

3.11.1 RACK

The PU and CU-crates constituting a CR80 Computer is housed in a single or multiple standard 19" rack, dependent of the actual system configuration. The specifications given here are related to the integration of systems while the datasheets give detailed specifications and dimensions for the available standard racks.

The general construction of the racks is as shown in figure 3.11-1. The width is standard 19", while the height and depth will depend on the related type No.

Installation of the system units in the rack is performed as illustrated in the following figures:

3.11-2 CR80 Rack Front View
3.11-3 CR80 Rack Rear View
3.11-4 Installation of Units in CR80 RACK

The following comments to the figures should be noted:

Notes to figure 3.11-1, General Rack Construction:

1. The rack dimensions correspond to 19" standard.

2. The height, H, varies with the different standard types and is selected in accordance with the actual system requirement, i.e. number of system units to be housed in the rack.

3. The depth, D, varies with the different standard types and is selected in accordance with the actual system requirement.

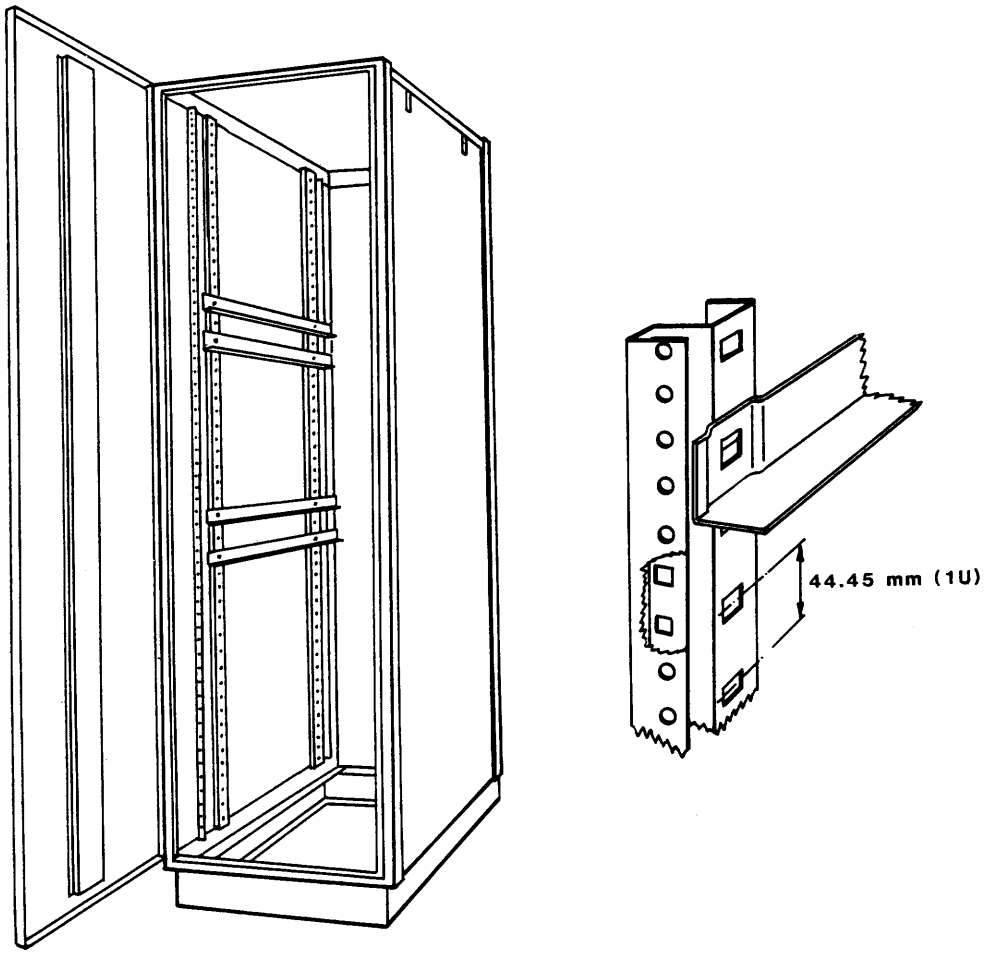


Figure 3.11-1
CR80 Cabinet and Installation of Supporting brackets

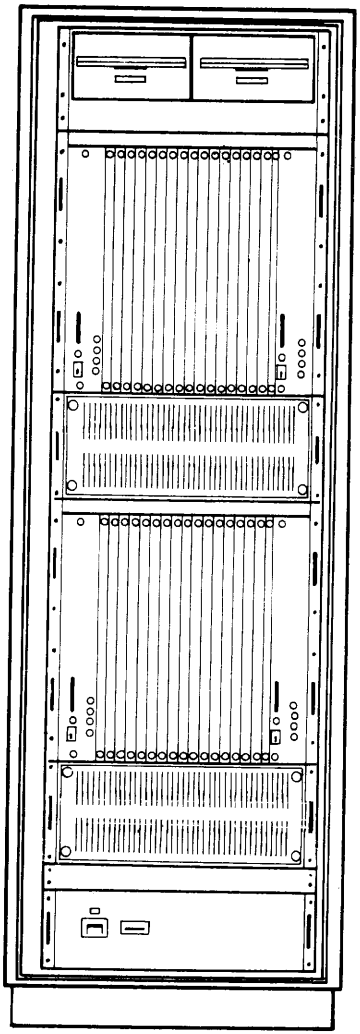


Figure 3.11-2
Typical CR80 Front View

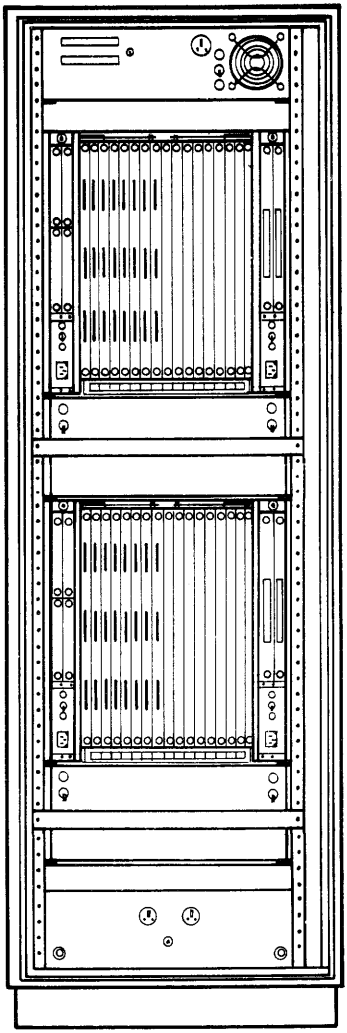
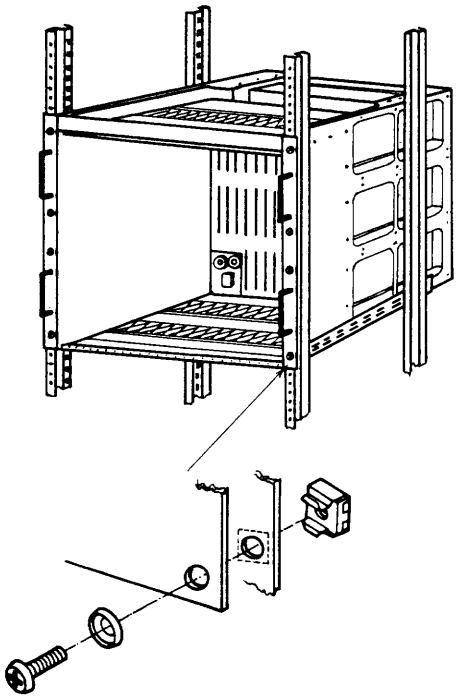


Fig. 3.11.3
Typical CR80 Rear View

Figure 3.11-4.
Crate Installation



4. The supporting brackets are inserted just below the System Unit. One right and one left of each unit.

Notes to figure 3.11-2, CR80 Rack Front View:

1. The system layout is an example and alternative layouts are possible; the only restriction is that a Fan unit be located just below the PU and the CU.

2. The Mains filter is preferably located in the bottom of the rack as shown, but it could alternatively be located at the top.

3. The Mains filter has the front panel towards the front of the rack, but it could as well be turned around so the front panel is towards the rear of the rack. If this solution is chosen, a blank panel is mounted on the front instead of the filter.

Notes to figure 3.11-3, CR80 Rack Rear View:

1. Cables are not included in the figure as the number of cables depends on the system configuration, but it should be mentioned that cables should be supported in accordance with good engineering practice.

2. The shown system configuration example includes two power distribution panels located at the bottom of the two fan units. No special requirements are set on the location of the panels but they should be located to allow access to the system, for example opening of the adapter crates in the PU and CU's.

3. As mentioned in the notes to figure 3.11-2 the Mains filter can be turned around.

3.11.2 FAN Units

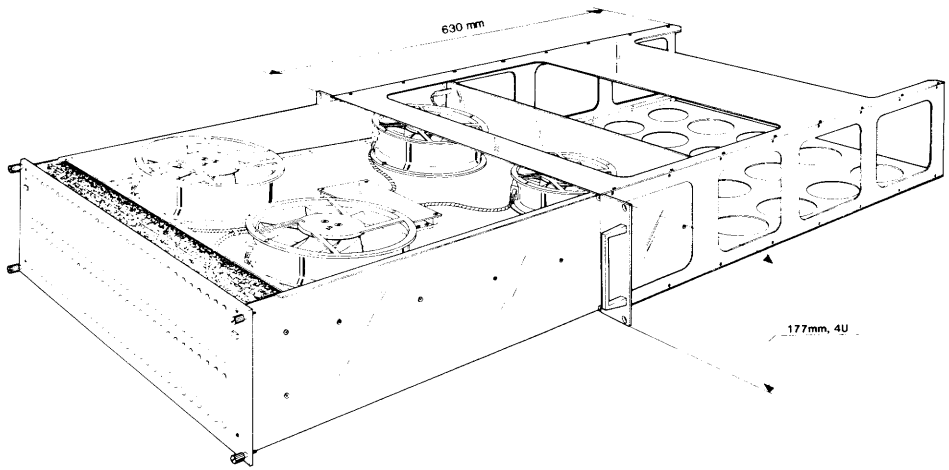
The FAN UNIT consists of a metal frame which is mounted permanently in the 19" rack just below the crate (ref fig. below). The frame contains a upper side clearance allowing the cooling air to dissipate vertically into front and rear magazine of D-crate. The frame contains a drawer carrying electrical circuits, fan motors and filter cushion.

The FAN UNIT applies two electrically independent systems (A and B), each consisting of axial fan motors with ball-bearings. Each system is capable of maintaining sufficient cooling air to the associated crate. The two systems are supplied from different mains phases, in order that a mains failure will not stop removal of dissipated heat from the crate.

For each system a POWER-ON lamp is located on front. Power plug, ON-OFF switch and fuse are located on the rear side of the unit.

CR80M FAN UNIT

CR 8105M BASE LINE



3.11.3 Mains Power Distribution

The mains power is distributed within different types of rack utilizing the principles outlined below.
A typical mains power distribution within a rack is shown, the distribution is composed of three elements, the Mains Filter, the Power Distribution panel and Power Cables.

The mains filter is normally installed in the bottom of the rack, but could as well be located in the top of the rack. Various types are available for different mains voltages (240V, 220V and 110V, please refer to the datacheets). All types are housed in the same mechanical frame and have screw terminals for the mains input and Tussen female connectors for interface to the power distribution panels (J1-J4).

The power distribution panels used in the CR80 systems come equipped with plug and cable for connection to the mains filter and plugs for connection of power cables between the panel and the different units (PU, CU, Fan Units etc.).
Two different types of power cables are available , the S type and the M type as specified in the datasheets.

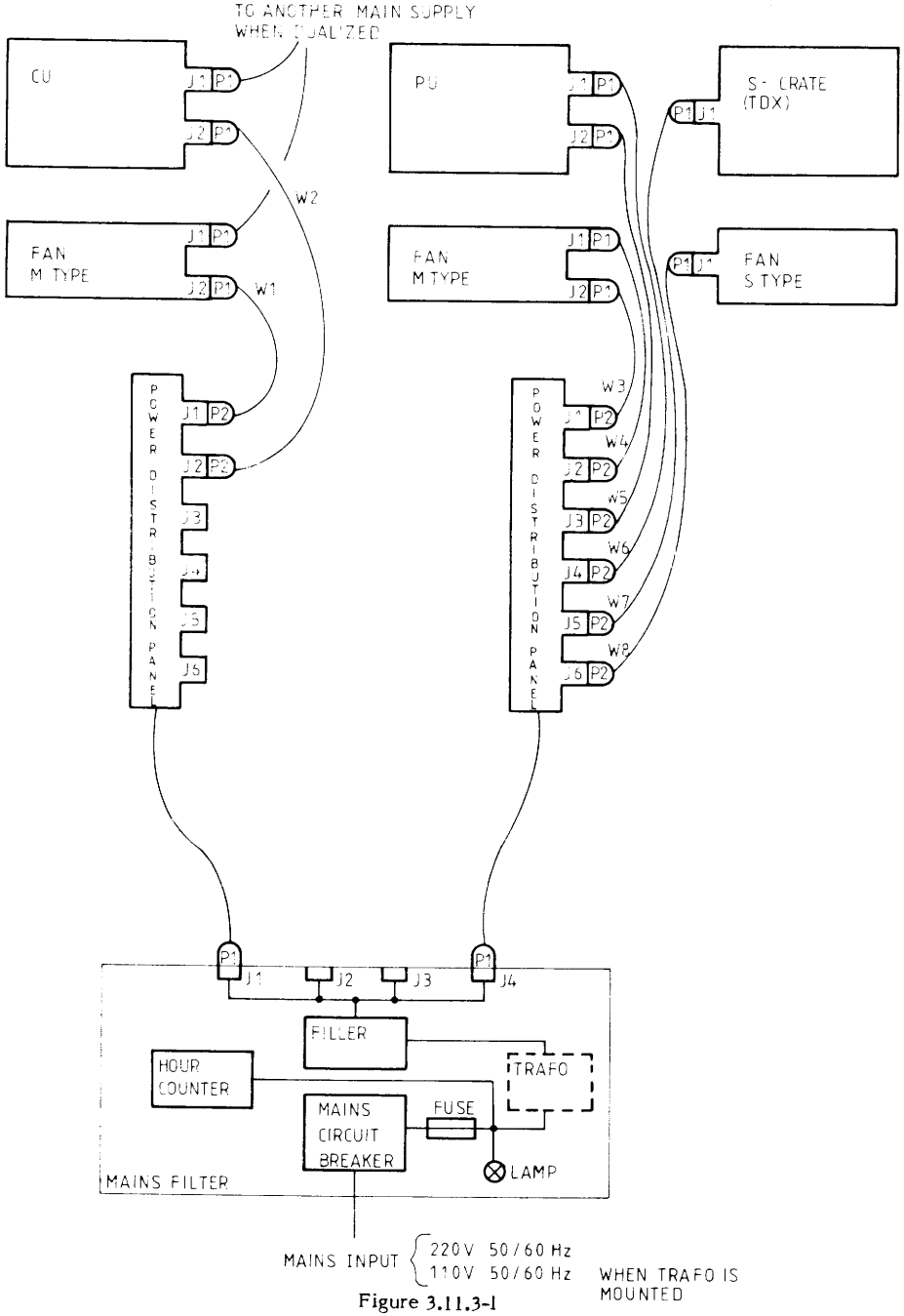
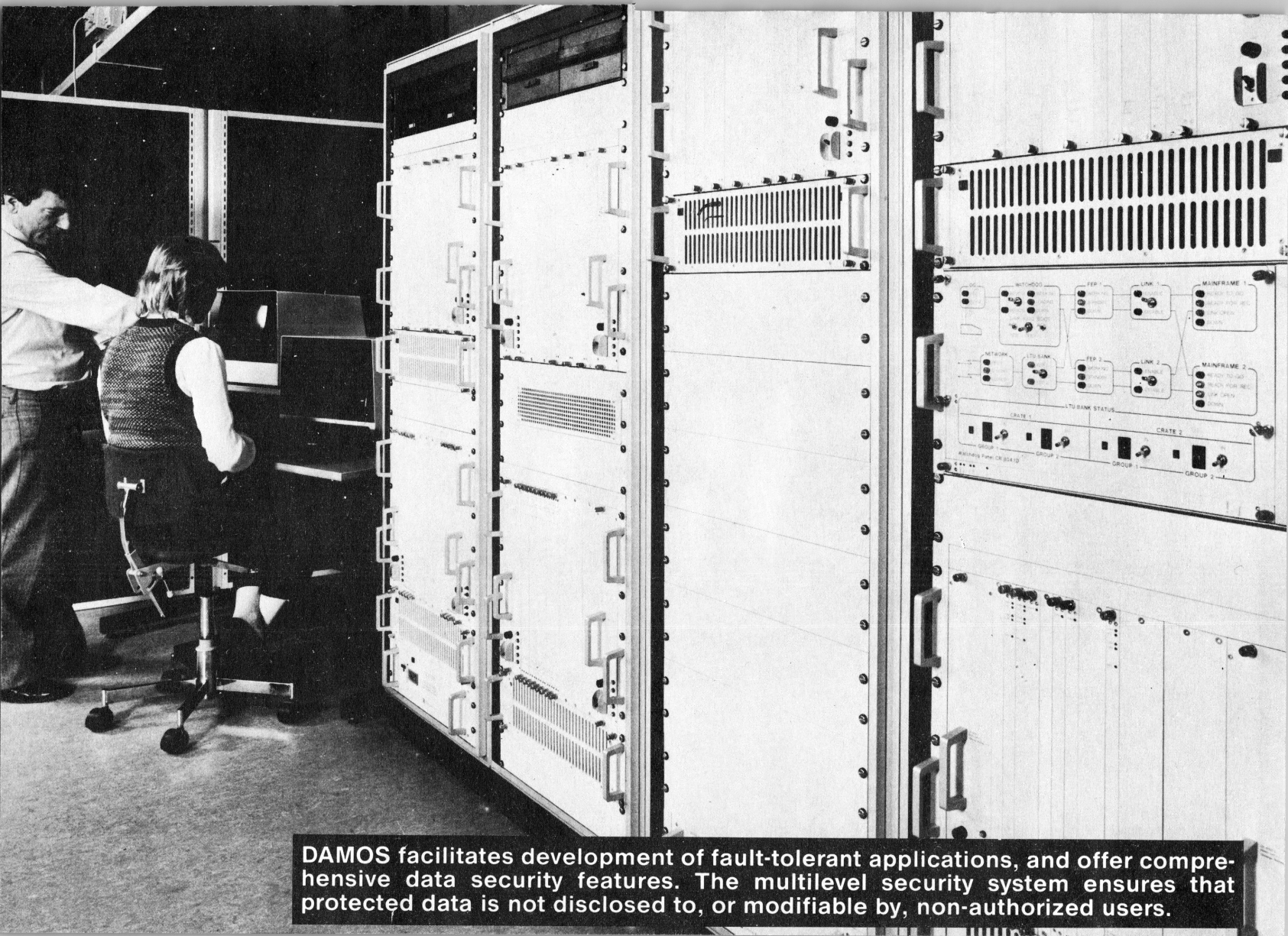


Figure 3.11.3-1

Mains Power Distribution

For One Rack

DAMOS, the virtual memory operating system for the CR80 FATOM and MAXIM Computers, is particulary suited for real-time applications, but supports also batch- and development environments.



DAMOS facilitates development of fault-tolerant applications, and offer comprehensive data security features. The multilevel security system ensures that protected data is not disclosed to, or modifiable by, non-authorized users.

4. DAMOS - CR80 STANDARD SOFTWARE FOR MEMORY MAPPED SYSTEMS

4.1 Introduction

The CR80 Distributed Advanced Multi Processor Operating System, DAMOS, is the standard operating system for memory mapped CR80 FATOM and MAXIM range of computers.

DAMOS includes a virtual memory operating system kernel for the mapped CR80 series of computers.

DAMOS fully supports the CR80 architecture which facilitates fault tolerant computing based on hardware redundancy. DAMOS supports a wide range of machines from a single Processing Element (PE) with 1 CPU and 128 K words of main memory, and up to a maximum configuration with 16 PE's where each PE has 5 CPU's and 16.384 K words of main memory and a virtually unlimited number of peripheral equipments including backing storage.

DAMOS is particularly suited for use in real time systems but supports also other environments like software development and batch. The main objectives fulfilled in DAMOS are: high efficiency, flexibility and secure processing.

DAMOS is built as a hierarchy of modules, each performing its own special task. The services offered by DAMOS include CPU, PE, and memory management. Demand paging is the basic memory scheduling mechanism,. Other levels of DAMOS provide process management and interprocess communication, basic device handling and higher level device handling including handling of interactive terminals, communication lines, and file structured backing storage devices.

DAMOS provides an operating system kernel which integrates supervisory services for real-time, interactive and batch systems. A comprehensive suite of software development tools is available under DAMOS. The following languages are presently available:

- assembler
- SWELL, the CR80 system programming language
- Pascal
- Cobol

The following languages are announced:

- Fortran 77
- Ada

The DAMOS standard software is described in the following sections. The description is divided into the following main sections:

- Overview of DAMOS
- Security,
  which describes the general DAMOS approach to data security
- Resource management,
  which describes the general DAMOS approach to management of system resources
- Kernel,
  which describes the DAMOS operating system kernel components
- DAMOS Input/Output,
  which describes the DAMOS standard interfaces to peripheral I/O equipment, the DAMOS disk file management, magnetic tape file management, and terminal and communication line management systems
- System initialization
- Operating system for software development
- Programming languages
- Utility software

4.2 Overview of DAMOS

DAMOS may be visualized as the implementation of a set of abstract data types and a corresponding set of tools for creating and manipulating instantia tions (objects) of these types.

The major components in DAMOS are the Kernel, the File Management System, the Magnetic Tape File Management System, the Terminal Management System and the Root Operating System.

The DAMOS Kernel exists in one incarnation for each Processing Element (PE). The data types and functions implemented by the Kernel are:

Data Type	Function
CPUs	CPU management and scheduling
processes	process management
virtual memory segments	memory management
PE's	PE management
synchronization elements	interprocess communication
devices	device management and basic device access methods
ports	basic transport service

The Kernel also provides facilities for

- processing of errors
- centralized error reporting
- a data transfer mechanism
- a PE service module

The File Management System (FMS) controls a group of disk units. The FMS superimposes a logical file structure on top of the disk space and provides functions for manupulating and accessing files.

The FMS may exist in several incarnations in each PE where each incarnation controls its own devices.

The Terminal Management System (TMS) is similar to the FMS. It provides functions for manipulating and accessing communication lines and terminals including line printers. The objects accessed via the TMS are called subdevices, a subdevice may be an interactive terminal, a line printer or a virtual circuit. The TMS controls a group of communication devices attached via LTUs, LTUXs or parallel controllers.

The TMS may exist in several incarnations in each PE, each incarnation controlling its own devices.

The magnetic Tape File Management System handles files on magnetic tape units.

A common security policy and hierarchical resource management strategy are used by the Kernel, the FMS and the TMS. These strategies have been designed with the objective of allowing multiple concurrent higher level operating systems to coexist in a PE in a secure and independent manner.

The Root Operating System is a basic low level operating system which initially possesses all resources in its PE.

4.3 Security

DAMOS offers comprehensive data security features. A multilevel security system ensures that protected data is not disclosed to unauthorized users and that protected data is not modified by unauthorized users.

All memory allocatable for multiple users is erased prior to allocation in case of reload, change of mode, etc. The erase facility is controlled during system generation.

The security system is based on the following facilities:

a. Hardware supported user mode/privileged mode with 16 privilege levels. Privileged instructions can be executed only when processing under DAMOS control.

b. Hardware protected addressing boundaries for each process.

c. Non-assigned instructions will cause a trap.

d. Primary memory is parity-protected.

e. Memory bound violation, non-assigned instructions, or illegal use of privileged instructions cause an interrupt of highest priority.

f. The hierarchical structure of DAMOS ensures a controlled use of DAMOS functions.

h. A general centralized addressing mechanism is used whenever objects external to a user process are referred to.

i. A general centralized access authorization mechanism is employed.

Centralized addressing capabilities and access authorization are integral parts of the security implementation. User processes are capable of addressing Kernel objects only via the associated object descriptor table. The following types of DAMOS objects are known only via object descriptors:

a.	Processes	Kernel objects
b.	Synchronization elements	
c.	Segments	
d.	Devices	
e.	PEs	
f.	CPUs	
g.	Ports	
h.	Files	FMS objects
i.	Subdevices	TMS objects

The object descriptor forms the user level representation of a DAMOS object. It contains all the information necessary for DAMOS to locate its low level representation and to ensure its security and integrity, such as:

a. Host PE

b. Object type

c. Object control block index for use by the Kernel to locate the corresponding object control block.

d. A sequence number which must match a number in the object control block (to prevent reallocated blocks from being erroneously accessed).

e. A capability vector specifying the operations which may be performed on the object by the process which has the object descriptor.

The access right information concerning the various DAMOS objects is retained in a PE directory of object control blocks (Kernel objects), and FMS directory (Files) and a TMS directory (subdevices). Each control is associated with a single object.

Authorization of access to an object is based on a general security policy which ensures that the socalled *-property is always fulfilled, and a discretionary access checking.

The security policy is based on a multilevel and -multicompartment security system. Objects are associated with a security classification level for each compartment, i.e., set of data with the same kind of information; subjects (processes) are associated with a security clearance level for each compartment. Both entities are described in a common type - the security profile. Security profiles are not necessarily ordered.

Discretionary access checking is based on identification of access classes of subjects (processes), and statements of access capabilities for explicitly enumerated access classes of subjects vis-a-vis a given object.

Access to an object is authorized if the following conditions are both fulfilled:

- the access operation requested is allowed according to the capability vector in the object descriptor

- the combination of process security profile, object security profile and operation (read or write) agrees with the security policy.

The security policy is:

- A process may read from objects with classification lower than or equal to that of the process. An untrusted process may write to objects with classification higher than or equal to that of the subject.

- A trusted process may write to objects with any classification.

A process can only obtain access rights (i.e., an object descriptor) to a DAMOS object in the following ways:

a. By inheritance from a parent process
b. By creating the object.
c. By successful look-up in a directory.

Similarly, a process can only distribute access rights to objects registered in its object descriptor table. This may be done:

a. By inheritance when creating a child process
b. By entering the object into the PE directory by a symbolic name.

When an object is entered into the directory it is specified by whom it may be looked up and what capabilities they should have vis a vis the object.

The object descriptor table and security profile of a process are kept in a memory which is accessible by that process when it is executing in the privileged mode, but protected against modification by the process when executing in the user mode.

4.4 Resource Management

The goal of DAMOS Resource Management is to implement a set of tools which enables the individual DAMOS modules to handle resources in a coherent way. This makes it possible for separate operating systems to implement their own resource policies without interference. In addition, built-in deadlock situations can be avoided.

The resource management module governs anonymous resources, such as control blocks. Examples of resource types are:

- process control blocks
- segment control blocks
- synchronization elements
- PE directory entries

Each type of resource is managed independently from all other types.

The resources are managed in a way that corresponds to the hierarchical relationships among processes. Two operating systems which have initially got disjoint sets of resources, may delegate these resources to their subordinate processes according to separate and non-interfering strategies. For example, one operating system may give all its subordinate processes distinct resource pools, whereby there is no risk of one process disturbing another. In contrast, the other operating system may let all its subordinate processes share a common pool. This results in a much better resource utilization at the risk for deadlock among these processes.

The principles employed by the DAMOS resource management, for each individual type of resource, can be summarized as follows.

- Resources are allocated and de-allocated on a process basis.

- A process draws its resources from a pool.

- Each process is allowed to use only a specified maximum number of resources from the pool.

- The resources associated with a pool may be used exclusively by one process, or they may be shared by several processes.

- A process may create a new pool for its subordinate processes, drawn from its own pool.

- When a process is created, its parent process specifies the pool from which it can draw its resources. This may be the pool of the parent process, a private pool created for the child process, or a pool shared with other children (siblings).

Handling of Resource Shortage

The resource management module does not implement any policy for handling resource shortage, but just reports the fact to its caller, which is the DAMOS module managing that particular type of resource. This object manager then implements the policy or defers the decisions to the parent of the offending process (i.e. an operating system).

The object manager passivates the process and reports the resource shortage to its parent. The parent may then take the following actions:

- resume the process after some time, hoping that the resource is now available.

- extend the resource pool of the child process, and then resume it.

Resource Management Functions

The following routines are available to the user:

CREATE POOL
which creates a new resource pool for a specific kind of resource. The number of resources for the new pool are drawn from the pool of the caller for this kind of resource.

UPDATE POOL
which transfers a number of resources between a specified pool and the pool of the caller.

GET RESOURCE STATUS
which returns the current parameters for the resource pool of the caller for a specific kind of resource:

- the maximum number of resources, which the caller may allocate
- the number of resources still available for the caller
- the number of processes drawing their resources from the same pool as the caller

GET POOL STATUS
which returns the current status of a specified resource pool:

- the type of resource governed by the pool
- the size of the pool
- the number of free resources
- the number of processes using the pool

4.5 Kernel

The DAMOS Kernel is a set of reentrant program modules which provide the lowest level of system service above the CR80 hardware and firmware level.

The Kernel consists of the following components:

- Directory Functions,
  which provide a common directory service function for the other Kernel components

- Process Manager,
  which provides tools for CPU management, process management and scheduling

- Page Manager,
  which provides memory management tools and implements a segmented virtual memory

- Process Communication Facility,
  which provides a mechanism for exchange of control information between processes

- Device Manager,
  which provides a common set of device related functions for device handlers and a standard interface to device handlers

- Device Handlers,
  which control and interface to peripheral devices

- Error Processor,
  which handles errors detected at the hardware and Kernel level and provides a general central error reporting mechanism

- Real Time Clock
  for synchronization with real time

- PE Manager,
  which provides functions for coupling and decoupling PEs

- PE Service Module,
  which provides service functions for remote PEs

- Transfer Module
  for a hardware based transfer of data in a PE and between PEs

- Basic Transport Service,
  which provides a general mechanism for exchange of bulk data between processes and device handlers.

The DAMOS Kernel components are described in the following subsections.

4.5.1 Directory Functions

DAMOS Objects and Their Denotation

The DAMOS Kernel offers a set of procedures for manupulating objects of different built-in abstract Kernel data types, e.g. processes and synchronization elements. A type is characterized by the operations which can be performed on objects belonging to it.

Any object resides in a PE which is called the host of the object. However, the object may be accessible from all other PEs in the system if certain conditions are fulfilled. These PEs are called remote with respect to that particular object. The set of PEs from which an object is accessible is called its scope.

Each process has an object descriptor table which registers its current environment of DAMOS objects. A process executing in the user state is only capable to operate on a DAMOS object if it is included in its object descriptor table. An object is passed as argument to Kernel procedures by specifying its location (index) in the object descriptor table.

PE Directory

The directory functions provide tools for the exchange of object descriptors between processes by other means than inheritance from parent to child.

The process giving the access rights and the process(es) obtaining them refer to the object by a symbolic name.

A process can only distribute access rights to objects which it has created. In order to distribute access rights, the donor process must enter the object into the PE directory by a symbolic name. Hereafter it can, by calling update procedures, change the capabilities of different user groups vis-a-vis the object and the allowed scope of the object.

Processes wanting to access the object must perform a lookup in the directory by specifying the symbolic name. The main conditions to be fulfilled in order to obtain the object reference are that:

- the user group of the requesting process has been assigned capabilities with respect to the object.
- the PE in which the process executes is included in the allowed scope of the object.

Security Checking

All access to objects is performed via object descriptors containing logical references and not 'hard' addresses. An object descriptor also contains a capability vector which describes the functions which the owner of the descriptor is allowed to perform on the object. This capability vector is checked before any access.

Since the security profiles of objects (including processes) and the capability vector in an object descriptor are static, the secutiry check for a process vis a vis an object is performed once and for all when the object descriptor is inserted in the object descriptor table of the process.

Whenever an attempt to violate security is detected by the directory functions the Error Processor module is invoked.

Functions Performed

The following procedures are available to user processes for manipulating symbolically addressed objects:

CATALOG

This procedure creates an entry for the object in the directory of the PE which is host for the object and associates the symbolic name with it. The scope and access rights may be changed by the

appropriate update procedures provided by the individual object managers.

UNCATALOG

This procedure removes the directory entry corresponding to the specified object from the PE directory where it is located.

LOOKUP

This procedure is used to obtain access to an object via its symbolic name.

IDENTIFY

This procedure will return the symbolic name of the specified object if it is cataloged.

DISMANTLE

This procedure is used by the calling process to indicate that it will not access the specified object any more. If the caller is the creator of the object it will be deleted. Thus, processes still having an object descriptor for the object are no longer able to access it. If the object is deleted it will be uncataloged.

GET OBJECT ATTRIBUTES

This procedure returns the object type, subtype and the logical number of the host PE of the specified object. The object need not be cataloged.

4.5.2 CPU Management

This module provides processor multiplexing for the DAMOS system. It may, thus, be viewed as the "multiprogramming support" module. In addition, the system parameters governing the processor multiplexing may be inspected and changed via routines implemented here.

Processors (CPUs)

A processor is a piece of hardware which is able to perform transformations on data presented to it, governed by a sequence of program instructions.

A CR80 Processing Element (PE) may be configured as a multiprocessor. The DAMOS system allows (as does the CR80 hardware) a maximum of five processors in a Processing Element. The minimum number of processors is, of course, one.

The processors are only known to the system as members of a processor pool; they cannot be controlled individually. The user interface to a processor, consequently, allows only reading the physical properties of the processor; no parameters may be changed.

Processor Pools

A processor pool is a group of from one to five processors. The number of processor pools may also range from one to five.

The processors of a Processing Element are statically allocated to the processor pools at system initialization time; this allocation lasts as long as the system.

The processor pool is the basis for process/processor scheduling (dispatching) in the DAMOS system. A process is for its entire lifetime confined to a specific processor pool. All processes, and all processors of a given pool are scheduled equally by the standard DAMOS dispatching algorithm.

The user interface to a processor pool allows the user (typically the operating system) to read and modify the variables that govern the dispatching.

Dispatching

The processor resources of a processor pool are multiplexed among the attached processes by a time slicing mechanism.

The processes are each assigned one out of sixteen priority levels. The assignment of a priority level to a process is given at process creation time. It may, however, be changed later on by the parent process (the operating system).

Each process is not assigned to a processor for more than the size of the time slice, before the next process at the same priority level is offered the opportunity to execute. It should be noted that this mechanism only takes care of the multiplexing among processes at the same priority level. For each priority level, however, a 'guarantee boolean' controls whether lower priority levels should get CPU resources with certain intervals.

The size of a time slice is a function of the logical priority of the process. This function may be changed dynamically.

The time slicing mechanism is implemented by means of the TIMER registers of the CR80 processors. When a process is assigned to a processor for execution, its time slice size is transferred to the TIMER register. It is decremented at regular intervals (with a size of 250 microseconds) during the execution by the processor firmware. Whenever the TIMER register becomes negative (and timer interrupts are not masked out), a timer interrupt occurs which causes the processor to be rescheduled.

The TIMER register is preserved at calls to blocking and deblocking synchronization routines, thus making time slicing independent from interprocess synchronization.

The dispatching is done independently from processor pool to processor pool. The following description applies to one processor pool:

the processes attached to a processor pool are divided into 16 priority levels. For each priority level a ready queue exists , which holds the processes of that priority, which are eligible for execution;

the scheduling policy of DAMOS is embedded in the routines which insert a process in the ready queue, and which select a process from the system of ready queues.

The scheduling policy follows a definite sequence. The ready queues are considered one at a time, starting with the queue corresponding to the highest priority level:

If the queue is empty, the next lower priority ready queue is considered.

If the guarantee boolean for this priority is true:

If the schedule count is non-zero, the head process is selected, and the schedule count is decremented by one.

If the schedule count is zero, it is reset to the preset value for this priority, and the next lower priority queue is considered.

Otherwise:

The head process is selected.

If the ready queue at the lowest priority level is empty, the scheduling is repeated if there are any processes at higher levels. Otherwise, the "idle"

process for that processor is selected. An "idle" process is a process never blocking itself, thus ensuring that it will always be eligible.

4.5.3 Process Management

Process management is concerned with long term control of processes (e.g. their creation and deletion from the system).

A process may be defined as an incarnation of the data transformations obtained by execution of a program in a given context. A context is taken to mean a set of CPU registers (CPU resident or saved) and the two memory translation tables describing the logical data space seen when the process is executing.

A process is represented by a Process Control Block (PCB). It is identified by an object descriptor as described in Directory Functions (section 4.5.1).

Each process competes with other processes for the system resources such as processor power, physical memory, I/O service, etc.

During the lifetime of a process it undertakes different process states. Figure 4.5.3-1 shows the process states in conjunction with long and medium term scheduling, and the process management functions which cause transitions between them.

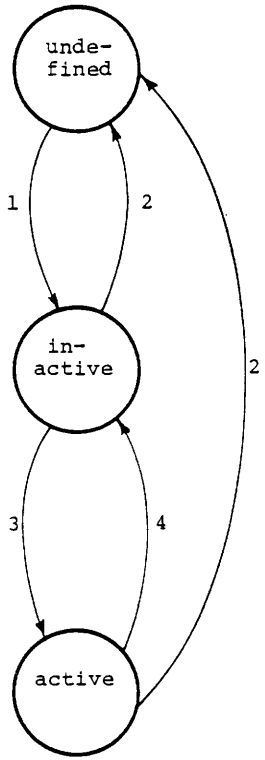


figure 4.5.3-1, Process Management

1: The process is subject to CREATE PROCESS
2: The process is subject to DELETE PROCESS
3: The process is subject to RESUME PROCESS
4: The process is subject to PASSIVATE PROCESS or RETIRE

The process state ACTIVE may be further refined by the short term scheduling (dispatching) as depicted in figure 4.5.3-2.

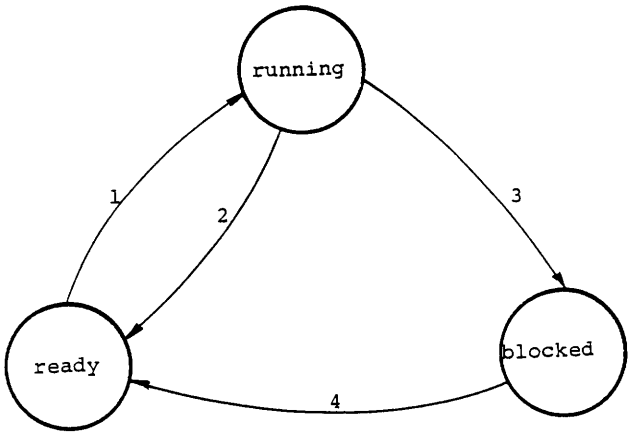


Figure 4.5.3-2, Active Process States

The events which cause transitions between the different active states are:

1: The process is scheduled by the dispatcher
2: The process is preempted by the dispatcher
3: The process is blocked awaiting some event
4: A previously awaited event occurs

Process Hierarchy

Processes are organized in a hierarchical manner as shown in figure 4.5.3-3.

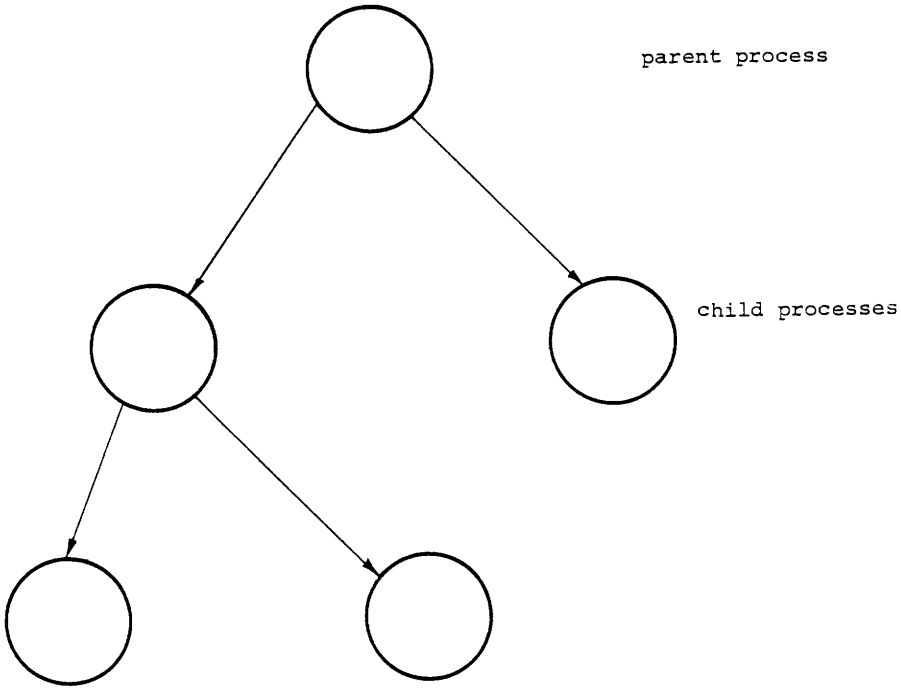


Figure 4.5.3-3, Process Hierarchy

A process may create subordinate processes. These are called child processes in relation to the creator process, which in turn is called their parent process.

The purpose of having a process hierarchy is to enable implementation of multiple operating systems each having supreme control over the processes of its process subtree.

The following process management routines are available to the user:

CREATE PROCESS

which initiates a new process instance to run on a specific processor pool. The new process is given access to specified resource pools, and inherits a set of object descriptors at the discretion of the creator. The new process does not get execution time before its parent process calls Resume Process.

RESUME PROCESS

which activates a process which was either just created or was previously passivated.

PASSIVATE PROCESS

which deactivates a process, that is the process is temporarily withdrawn from the dispatching

CLEAN UP PROCESS

which catches a child process and terminates its activities. Among other things, its possible child processes are cleaned up and deleted, and outstanding process activities (I/O requests, process communication) are cancelled.

DELETE PROCESS

which removes a previously cleaned process from the system. The resources occupied by the process are returned to the caller (its parent process).

RETIRE

which passivates the caller and sends possible error information to its parent process.

GET PROCESS ATTRIBUTES

which returns the current attributes of a child process

CHANGE PROCESS ATTRIBUTES

which changes specified attributes of a child process. The attributes may relate to the dispatching of the process (e.g. its priority), or to the page manager parameters for the process (e.g. the size of its working set).

4.5.4 Memory Management

The Page Manager (PM) implements a segmented virtual memory.

The hardware basis for the PM is a partitioned memory implemented via a memory mapping module (MAP), which performs a translation from logical addresses to physical main memory addresses. The translation is performed by means of a set of Translation Tables (TT). The MAP module contains several TTs. Two TTs are used for each CPU, one for translation of program addresses and one for translation of data addresses. The TTs currently used by a CPU are identified by means of two CPU specific Translation Table Registers (TTR).

In this section the following definitions are used:

Page

A fixed amount (1024) words of information

Page Frame

A physical storage area capable of holding one page

The page size is 1024 words. The TT defines a mapping from a set of logical pages into a set of main memory page frames and a set of access rights.

The domain of access rights has the elements:

- no access (full access in system state)
- read only access (full access in system state)
- full access (read and write) in all states

To cater for the situation where no main memory page frame corresponds to a logical page frame, a nil main memory page frame is defined. A mapping into the nil page frame corresponds to an "absent page".

The PM operates with the following memories:

- a Backing Store (BS)
- a Virtual Memory (VM)
- a Main Memory (MM)

The two first memories may be assumed to fill a contiguous memory space.

The connection between page frames and locations is that the page frame PF contains locations PF*1024 through PF*1024+1023.

The Page Manager implements demand paging as well as functions for programmed transfer of page frames between BS and MM. This support is useable for implementing overlaying and process swapping.

The Page Manager may also be configured to work without Backing Store. In this case the Virtual Memory is always resident in Main Memory.

Shared Memory

The Page Manager makes sharing of memory between processes in the same PE possible. Sharing is subject to security and discretionary access check.

Working Set

The working set of a process is defined as a set of MM page frames currently in use by that process.

Virtual Memory

A process may want to be able to refer to a memory larger than the physical memory of the PE.

In order to do this an abstraction called a virtual memory is defined. A process may possess a very large lump of virtual memory - say, 2 megawords or more.

Pages in the virtual memory correspond to page frames which are either located in the main memory or at the backing store.

Only those page frames of the virtual memory which reside in the main memory are accessible by the CPU for operand/instruction fetching and modification.

The Page Manager keeps track of where the virtual page frames are located, and if the CPU attempts to address a page frame which is on backing store the Page Manager will take actions to transfer the page frame to main memory.

The Page Manager is invoked by a page fault occurring when the CPU addresses the page frame which was absent in the MAP module. However, because of the limited addressing capability inherent in a 16-bit architecture (64 K program locations and 64 K data locations), the process will only be able to see (i.e. address by machine instructions) parts of its virtual memory at any one time.

The virtual memory is therefore segmented and a process has to determine which parts of the virtual memory it can access by composing a logical address space of virtual segments.

The size of the total virtual memory for all processes is limited by the available backing store.

Segments

A segment is defined as an identified, contiguous portion of the virtual memory. A segment may consist of one or more virtual page frames.

Segments are standard DAMOS objects and are associated with a security profile which is defined at segment creation time.

A segment created by a process can only be accessed by the process itself or decendants of the process.
A child-process can inherit a segment from the parent and it can look-up a segment cataloged by an ancestor, and it can also create segments by itself.

Only the creator-process can dismantle a segment so that it disappears, and the segment must then not be referenced by any other process.

The creator of a segment will at creation-time be accounted for the resources needed to build the segment.
When a process which have access to a segment wants the segment or part of it into the main memory, it will be accounted for main memory resources equal to the number of pages it wants in main memory. The working set of a process is defined as a set of MM page frames currently in use by that process.

Resident segments

In the case where two or more processes share a segment, they will each be accounted in the RM for the amount of WS used to hold the pages of the segment.
This means that some of the page frames may be in the working set of more than one process.

This is necessary in order to avoid a coupling between the physical TTs for the different CPUs. However, it means that sharing of segments may reduce the utilization of the main memory.

In order to allow operating systems to take advantage of reentrant programs and data areas without degrading main memory utilization the concept of resident segments is introduced.

Resident segments are always located in main memory and the page frames are only included in the working set of the process which created the segment. A process may only dismantle a resident segment which is created by itself when no other process references (uses) the segment. It is considered a fault on behalf of the creating process (read OS) to try to dismantle a resident segment still used by subordinate processes.

Resource Management

The PM uses four types of resources, which are administered by the Resource Management module (RM). These are:

- Virtual Memory (VM)
  For each virtual memory page frame of a process, there must be a page frame on the Backing Store (BS) for the virtual page to reside on, when it is not in main memory.
  The maximum number of virtual memory page frames is therefore the number of page frames on the BS.

- Working Set (WS)
  The maximum number of pages in Working Sets is equal to the number of main memory page frames.

- Segment Control Blocks (SCBs)
  Blocks for controlling the use and allocation of segments

- List Elements (LEs)
  Blocks for chaining the various PM-data structures together.

For every resource the RM-module keeps account of how many units of the resource each process currently uses, and the maximum amount it is allowed to claim.

Administration of VM and WS Resources

The general principle employed when allocating VM, BS and WS is that the resources of a child process must be donated by its parent process. How the resources are distributed between the child processes of a particular parent process is determined solely by that parent process. However, any resource given to a child process is taken from the resources of the parent process.

The WS and VM used and useable by a process is as mentioned above controlled by the corresponding resource limit in the RM-data. This limit defines the resources that the process and all its subordinate processes may use.

The limit is composed of two contributions:

used which is the number of page frames used by the process for its own use, or granted to child processes.

max claim which is the maximum number of page frames it may use under any circumstances. This limit can, however, be adjusted by the parent process.

It is possible for a parent process to reclaim WS resources from a child process by using the function change_pm_attributes. However, only resources which have not been delegated further down the family tree by the child can be recovered in this way. That is, the reclaiming does not work recursively through the process levels.

Release of WS

When a process requests a segment to be saved at the BS, the pages are not released from the working set of the process until the save operation is complete. This ensures that a process may not covertly occupy more main memory resources than it is allowed to do by its WS_max_claim.

However, this technique also prevents a process from stipulating its number of free page frames at a particular time. In order to relieve application processes from considering this problem, the following strategy is used.

When a function is called which requires more free working set resources than are available, the PM checks if such resources are tied up only during a transfer of pages to BS, and if this is the case, the process will be delayed until those pages become available, i.e. until so many transfers are complete that the necessary page frames are available.

Demand Paging

Demand paging is the mechanism whereby processes may execute in a virtual memory space which is larger than the physical memory space of the machine, and whereby the virtual memory space is automatically composed of physical page frames as required. The virtual page frames which do not correspond to physical page frames will reside on backing store.

The demand paging mechanism is invoked when a process encounters a page fault by addressing an absent page. When this happens, the CPU hardware generates an interrupt.

The processing of a page fault first of all checks whether the page fault occurred in a legal logical page which is part of a mapped in segment. If this is not the case, the page fault is considered a run time error on behalf of the process.

Otherwise, the missing virtual page is determined and actions are taken to bring it into main memory. Firstly, it is checked whether the page is already in main memory in an active page frame (it might have been brought in by another process) or if its contents are still in a page frame which is free or queued for saving.

In both cases, the virtual page can immediately be obtained in main memory and included in the working set of the calling process.

If the page cannot be found in main memory, a free page frame is allocated and the page is brought in from backing store. This may require that another page is saved on BS, namely in the case where the working set cannot be expanded.

Page Managing

A process has a maximum logical memory of 64 K program locations and 64 K data locations. This space is partitioned in 64 logical program pages and 64 logical data pages. To manage and address this logical memory, the PU has two tables of each 128 words for every process:

The first table is the translation table or TTBL. This table defines the mapping from logical to physical memory. Each entry contains a 14-bit field, which may contain an address of a physical page frame corresponding to the logical page frame, and a 2-bit access field which defines the access rights of the process to the page frame (full, read only, no access) or whether the page is absent (not allocated a physical page frame). This table is an exact mirror of the MAP TT for the process.

The second table is the segment table or STBL. This table defines the mapping from logical to virtual memory. Each entry in this table contains a pointer to a segment control block (an internal PM data type) and the access right of the process to the segment (full, read only, no access). If the entry does not correspond to a segment, the segment control block pointer is nil.

Support of Different Memory Types

The PM supports the following types of memory:

- internal RAM, i.e. RAM memory connected to the PU P-Bus and the PU C-bus
- external RAM, i.e. RAM memory connected to the PE Datachannel
- compartmentalized RAM, i.e. RAM which is part of a Peripheral Module
- internal PROM, i.e. PROM memory connected to the PU P-Bus and the PU C-Bus
- external PROM, i.e. PROM memory connected to the PE Datachannel

Internal and external RAMs are the only types of memory in which demand paging is supported. For the other types of memory, segments are always resident in main memory.

Functions Performed

The following functions are provided by the PM for use by user processes:

- Create_segment
  builds up data structures in the page manager to represent a segment.

- Update_segment_access
  changes the access rights which is required of a process to access a certain segment.

- Get_segment
  brings the segment into main memory, and includes it into the working set of the calling process.

- Put_segment
  takes the segment out of the working set of the calling process, and saves the segment on BS.

- Make_segment
  works like get_segment, with the difference that it does not provide any meaningful contents in the segment, but rather allocates space for it.

- Free_segment
  works like put_segment, with the difference that it does not save the segment on BS, but rather releases the main memory occupied by it.

- Map_in_segment
  maps the segment into the logical space of the calling process, either starting from a specified logical address, or where there is free space.

- Map_out_segment
  maps the segment out of the logical space of the calling process, thus enabling other segments to be mapped in.

- Lock_page
  locks a specific page of a segment into main memory. The page may or may not be mapped in.

- Unlock_page
  nullifies the effect of a previous call of lock_page.

- Change_pm_attributes
  changes the available WS or VM resources of a child process of the calling process. This may involve suspension of the child process while the change is performed.

4.5.5 DAMOS Process Communication

Process communication in DAMOS is realized by using synchronization elements. A synchronization element may be thought of as a mailbox, where one process delivers information and another, or the same, process removes information.

From the above it is seen that the DAMOS process communication is a communication between processes and synchronization elements rather than communication between processes.

However, by allowing only one specific process to remove information from a synchronization element, a process-to-process communication can be created as a special case of the more general DAMOS Process Communication Facility (PCF).

Two basic types of communication functions are provided. The send type, which enables a process to deliver information at a synchronization element, and the await type which enables a process to wait until information is available, at one or more synchronization elements, and then remove the information.

Figures 4.5.5-1 through 4.5.5-3 sketch some of the possible send and await combinations.

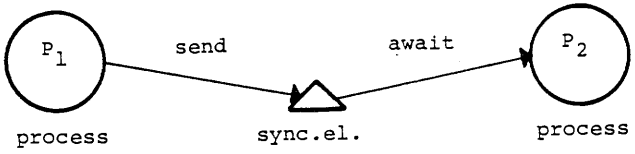


Figure 4.5.5-1

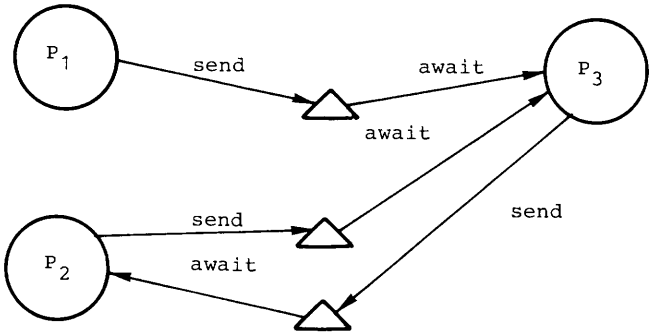


Figure 4.5.5-2

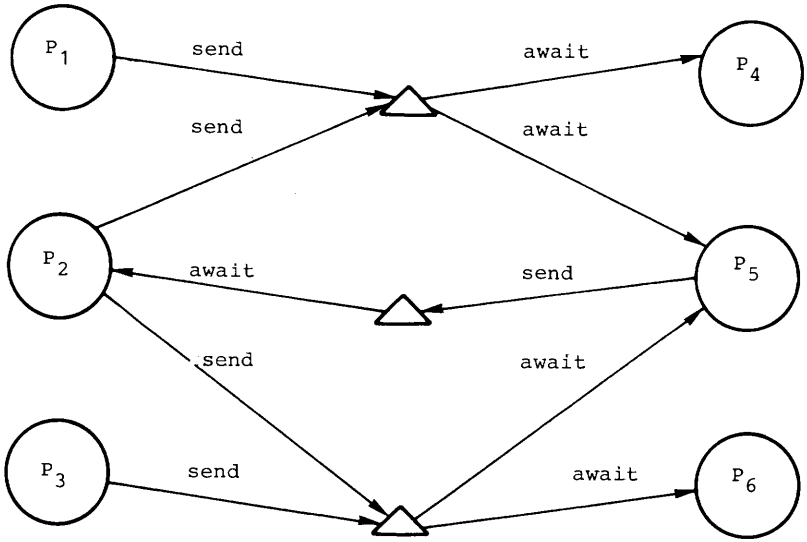


Figure 4.5.5-3

Process Communication Resources

Two basic types of resources are considered in the process communication facility, namely synchronization elements and communication elements. Communication element is a shorthand for the resources which are required when information is delivered to or removed from a synchronization element.

DAMOS provides a pool of unnamed synchronization elements and a pool of communication elements within each PE. Processes may create and delete synchronization elements, and processes may also have different communication element requirements.

The maximum number of resources which are available to a process must be specified when it is created. The resources are taken from a pool of resources. This pool, which is donated by the parent process, may or may not be shared with other processes depending on the policy imposed by the parent process.

Since synchronization elements are DAMOS objects it is possible to have a name associated with them and thereby communicate the existence of a synchronization element to other processes.

A synchronization element is given a symbolic name by using the directory function catalog.

When a synchronization element is catalogged a scope must be associated with the name. This scope defines whether the synchronization element (i.e. its name) shall be known in other PEs than the one in which it has been created.

Synchronization elements which are created in the same PE as processes performing operations on them, are called local synchronization elements, while synchronization elements which are created in another PE than the processes performing operations on them are called remote synchronization elements.

Process Communication Procedures

The following are the main procedures available for process communication:

create synchronization element

This procedure creates a synchronization element object which is fully controlled by the creator, i.e. the creator determines which other processes may be allowed to use the synchronization element and the kind of functions these processes may perform.

dismantle (synchronization element)

This procedure deletes a synchronization element. A synchronization element can only be deleted by the process which has created it.

update synchronization element

This procedure enables the control of which users may be granted access to a synchronization element, as well as their kind of access may be controlled.

send

This procedure delivers information at a synchronization element which may be local or remote. If a process is waiting at the synchronization element, then the information will be given to that process, otherwise, the information will be entered into an information queue at the synchronization element. The sender proceeds immediately upon completion of the call.

await

This procedure delays the calling process until information is available. The caller has the ability to wait for information from more than one synchronization element. The number of synchronization elements, which a process wants to wait for, may be less than or equal to the number of synchronization elements specified in the await call. However, all synchronization elements must be local. For each of the synchronization elements, which the process has to wait at, it is examined as to whether information is available or not. If information is available, then it will be reserved for the process; otherwise, the process will be entered into a queue - at the synchronization element - of waiting processes.

The process is blocked, if it does not obtain information from as many synchronization elements as specified in the await call. When the process is allowed to proceed, it is provided with the available information.

Figure 4.5.5-4 shows the transitions of processes between the ready, running and blocked states.

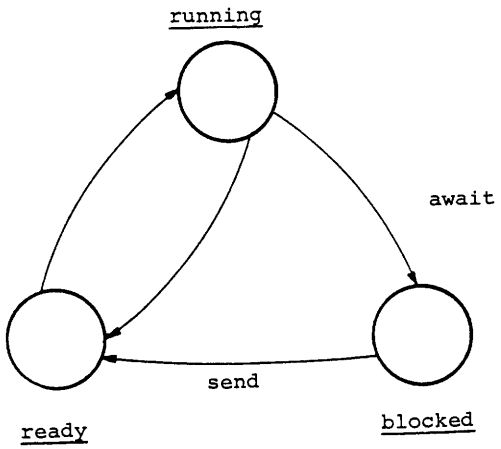


Figure 4.5.5-4

4.5.6 Device Management

The Device Manager (DVM) module is the layer of software which provides the lowest level interface to physical input/output devices.

The DVM consists of a number of device handlers - one for each kind of device attached or attachable to the CR80 system and a device control block for each physical device actually attached.

Devices as DAMOS Objects

Devices are included in the DAMOS object types and are thus addressed by object descriptors. A process is only able to access a physical device directly if it possesses an object descriptor for the corresponding device.

An object descriptor for a device can be obtained by creation, lookup in the PE directory, or by heritance from the parent process.

When a logical device is created the interrupt vector (IV) for the corresponding physical device is associated with the object descriptor and other processes cannot create a device with the same IV until it has been dismantled by the creator.

Before the physical device can be accessed the device control block (DCB) must be initialized by the creator.

When the device has been initialized it may be accessed by any process having an object descriptor for the device. This is possible until the device is shut down by the creator.

Functions Performed

CREATE_DEVICE

This procedure is used to prepare the DVM part of a device control block (DCB).

HANDLE_DEVICE

This procedure is the only entry point for users to communicate with a physical device. The call parameters are:

- device identification
- wanted function
- function dependent parameters

The DVM interprets the device identification and invokes the corresponding device handler. If the device is initialized and access checks are passed, the wanted function will be performed.

The functions supported are specific for the different kinds of devices.

Normally the handler will report completion of the requested function by "sending" to a synchronization element specified by the caller.

4.5.7 Device Handlers

A device handler is a set of pure procedures for performing operations on a device and a device control block. A device handler has three entry points:

- handle_notification, which is used by the CPU interrupt firmware
- enter_handler_P, which is the common entry point for all device handler functions available for processes. This entry can only be reached via DVM
- enter_handler_H, which is the common entry point for all device handler functions available for other device handlers.

Device handlers have different characteristics and functions which are specific for the kind of device in question. However, some functions are always provided for a device:

- ACCESS, which is invoked when a process wants access to the device. A security check of the caller vis-a-vis the device is performed.
- INITIALIZE, which initializes a device control block and sets up the connection from an interrupt vector to a device handler and a device control block.
- SHUTDOWN, which cleans up the device control block for a device and breaks the connection from the associated interrupt vector to the device handler and the device control block.
- DISMANTLE, which is invoked when a process wants to dismantle the connection to the device.
- ERROR, which is invoked when an error has been detected by CR80 hardware and it can be related to a specific device.
- STATUS, which returns statistics and detailed information about errors on the device.
- TEST, which will activate the built-in self testing program in the physical device.

Device Failures

As seen from a device handler point of view a physical device may be unreachable for several reasons, e.g.:

- Data channel failure
- CIA failure
- Crate Power Supply failure
- Device failure

Apart from failures possibly reported by the device itself via its status register all errors will be reported by the CR80 hardware. This is done by vectored interrupts.

When pertinent the error indication will be passed to the device handler which may take a relevant action, e.g. stimulate a switch over to a back up device.

A power failure concerns all devices in the crate. The PE receiving the interrupt can no longer access these devices via its Data channel. However, the CR80 architecture provides a power supply for each of the two separate Data busses in a CU-crate. Thus the device can be accessed via a partner PE with Data channel to the same device.

The device handlers available in DAMOS are described in the following subsections.

4.5.7.1 Disk Handler

The DAMOS Disk Handler is able to manage one CR8044 disk controller with a maximum of

- 4 disk units in daisy chain
- 64 K words of buffer RAM

The disk handler makes an internal scheduling of commands concerning separate disk units in order to utilize the ability of the controller for concurrent operations on separate units, and for minimizing head movements.

The disk handler performs a mapping from logical sector numbers to physical sector addresses.

If a read or write operation fails, the disk handler attempts to recover the error if possible. In case of a failed read operation the recovery may include read with servo and/or strobe offset.

The following Disk Handler specific commands are supported:

INCLUDE,
which introduces a subunit, i.e. a disk drive or a separately addressable part of a disk drive as, for example, the fixed head part of a drive.

EXCLUDE,
which again removes a subunit.

GETSECTORS,
which reads a number of sectors from a logically addressed place on the volume on a particular subunit.

PUTSECTORS,
which writes a number of sectors to a logically addressed place on the volume of a subunit.

PROTECTSECTOR,
which writes and writeprotects a logically addressed sector.

MARKSECTOR,
which marks a logically addressed sector as bad.

CHECKSECTOR,
which activates a CRC-check of a sector.

FORMATSUBUNIT,
which formats a volume on a subunit.

4.5.7.2 Magnetic Tape Handler

The DAMOS Magnetic Tape Handler is able to manage one CR8045 magnetic tape controller with a maximum of:

- 8 tape transports in daisy chain
- 64 K words of buffer RAM

The magnetic tape handler performs an internal scheduling of commands concerning separate tape transports to utilize the ability of the tape controller to transfer data on one transport while another is rewinding or skipping.

The following MT handler specific commands are supported:

INCLUDE,
which introduces a tape unit into the control of the tape handler

EXCLUDE,
which again removes a tape unit from the control of the handler.

MOUNTTAPE,
which winds the tape until the first block.

REWIND,
which rewinds the tape and subsequently turns it off line.

READBLOCK,
which reads a block of data from a particular unit.

WRITE BLOCK,
which writes a block of data to a particular unit.

WRITE_EOF,
which writes an end_of_file mark on the tape of the specified unit.

ERASE,
which erases the tape from its current position and on.

LOCATE,
which will locate a specific block of a specific file. Blocks are numbered absolutely within each file. Files are numbered absolutely within each tape. The tape handler internally keeps track of the current position.

4.5.7.3 LTU Handler

The DAMOS LTU Handler manages one CR8066 LTU (Line Terminating Unit) with a maximum number of

- 4 communication lines

The following LTU specific commands are supported:

LOAD,
which loads the LTU with programs and data

INCLUDE,
which defines the dynamic properties of the line connected to one of the LTU channels.

EXCLUDE,
which removes a channel from the control domain of the LTU handler.

GETBLOCK,

which delivers the next block of data from a particular channel.

PUTBLOCK,
which outputs the next block of data to a particular channel.

GETCONTROLBLOCK,
which delivers the next block of control information from a particular channel.

4.5.7.4 TDX Handler

The interconnections (LOG_LINES) supported by the TDX system may be created, changed and removed dynamically. Though the low level protocol of all interconnections is the same (using the TDX frame formats), different higher level protocols may be implemented on the LOG-LINE level.

As a consequence, the TDX Handler manages:

- LINE CONTROL,
  i.e. creation and removal of LOG-LINES

- PROTOCOL ADAPTION,
  i.e. adapting a LOG-LINE protocol to the IO system used by the application process. This may include assembly/disassembly of data units, data conversion etc.

- DATA TRANSFER

The following TDX specific commands are supported:

CREATELOGLINE,
which establishes a connection between the host interface and a controller.

REMOVE,
which removes the connection between the host and the LTU.

GETBLOCK,
which reads the next record from a particular line.

PUTBLOCK,
which writes a record to a particular line.

4.5.7.5 Operator Console Handler

The Operator Console Handler supports the serial communication interface of the CR8020 MAP module.

The following Operator Console specific commands are supported:

GETBLOCK,
which delivers the next block of data from the channel.

PUTBLOCK,
which outputs a block of data on the channel.

4.5.7.6 Line Printer Handler

The Line Printer Handler supports the CR8046 parallel controller which may interface up to 4 printers.

The following Line Printer specific commands are supported:

ASSIGN,
which sets up a line printer channel.

DEASSIGN,
which removes a line printer channel.

PUTBLOCK,
which outputs a block of data on a line printer channel.

4.5.8 Error Processor

The Error Processor module is invoked by error interrupts from the CPU firmware (e.g. parity error) or from other DAMOS kernel modules when an error has been detected (e.g. security violation).

When the error can be associated to a process this process is forced to retire and the parent process is informed. For certain types of error the parent may be able to recover the situation. It is then possible to let the child resume processing.

All errors considered harmful to the system are reported to a Central Error Synchronization Element (CESE). The error reports can only be retrieved from the CESE and handled by a process having special capabilities.

No error Processor functions are available to user processes.

4.5.9 Real Time Clock

The DAMOS real time clock facility offers the following functions:

- Reading and setting the system clock
- Tying the execution of processes to time
- Conversion between different formats of a time variable.

The setting of the system clock is restricted to the DAMOS root process.

Hardware Support

The RTC module is implemented by means of the real time clock of the CR80 MAP module. This clock is an accumulating timer with an alarm. The timer counts milliseconds from a certain point in time (Midnight the 1st January 1980). The size of the counter is 48 bits, allowing for a total run of 8919 years. This clock can be read and set by the procedures of the RTC module. The alarm is implemented by means of another 48-bit variable. If the timer and alarm variable hold the same bit pattern, a vectored interrupt is issued. This interrupt is received by the RTC module, and the action which has to take place at this point in time, is performed. The alarm is set by the RTC module to the next time an event should be scheduled.

Functions Performed

The following routines are available to the user of the RTC module:

SET TIME,
which specifies a new value of the 48-bit time variable.

READ TIME,
which returns the current value of the system time in the internal 48-bit format.

SCHEDULE TIME SIGNAL,
which requests the RTC module to schedule a SEND operation on a semaphore synchronization element at some time(s) in the future. The time(s) may be specified as:

- A certain point in time
- After the elapse of a certain period of time
- At regular intervals from now on

CANCEL TIME SIGNAL,
which cancels a previously scheduled time signal

CONVERT TIME TO INTEGER,
which converts a 48-bit string in the internal time format to a sequence of integers representing: year, month, day, hour, minute, second, and millisecond.

CONVERT TIME TO ASCII,
which converts a 48-bit string in the internal time format to an ASCII string.

CONVERT INTEGER TO TIME,
which converts a sequence of integers to a 48-bit string in the internal time format.

4.5.10 PE Management

This DAMOS kernel module provides facilities for high level management of the logical connections between individual Processing Elements (PE's) attached to a Supra Bus.

Two PE's are not able to communicate until a logical connection has been established. The connection procedure is accomplished only if both PE's agree.

The PE's to which a PE has established a logical connection are known as the partners of that PE. New partners may dynamically be included in the acquaintance of a PE, and current partners may be excluded as well.

The PE module is able to manage from 1 to 16 PE's.

Function Performed

The functions provided by PE Management are:

CREATE_PE,
which allocates and initializes a control block for a partner PE (in spe)

INCLUDE_PE,
which establishes a logical connection with another PE if agreed upon by the latter.

EXCLUDE_PE,
which breaks the logical connection to a partner PE.

4.5.11 PE Service Module

The PE service module is responsible for performing "remote kernel functions".

Remote kernel functions are DAMOS kernel functions, which are performed in a PE different from that of the calling process.

The PE Service Module provides interface to the following DAMOS kernel modules:

- PE Management
  The PE service module is used when two PE's must establish a connection.

- Directory Functions
  The PE service module is used when a process in one PE wants to have access to an object in another PE.

- Process Communication
  The PE service module is used when a send function refers a remote synchronization element.

The support of "remote kernel functions" is performed by using a kernel process (KP). A KP is a process which is able to perform DAMOS functions on behalf of other processes, and is also able to communicate with KPs in other PEs.

When a DAMOS kernel function is activated with a reference to a remote object the kernel process will be invoked. The KP will then take over the processing of the function until it has been completed in the PU which is the host for the object.

The invocation of the KP will not be visible to the process which performed the DAMOS function on the remote object.

Figure 4.5.11-1 below illustrates the interface between the Directory Functions and the PE Service Module.

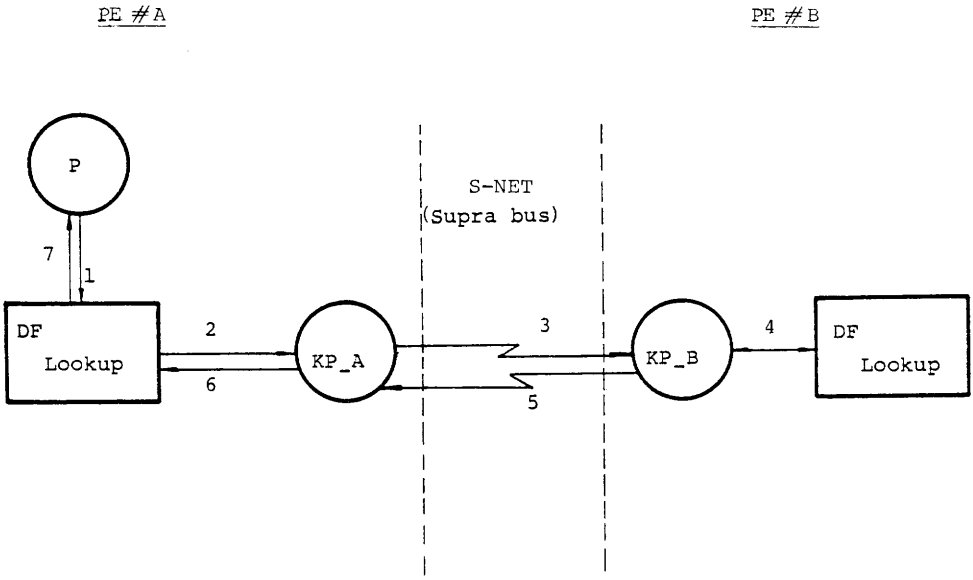


Figure 4.5.11-1

The interface is shown by a walk thru of the steps which are performed when the process P, in PE A, performs a lookup function in the directory of PE B.

Step Description:

1. P calls the DF procedure LOOKUP to obtain the object descriptor for the object named "mysync" residing in PE B.

2. LOOKUP requests KP_A to perform a remote lookup in PE B. P is blocked while this request is processed.

3. KP_A sends a request for a lookup of object "mysync" to KP_B. This request is transmitted to PE B by means of the supra bus.

4. KP_B receives the request and recognizes it as a request for lookup of the object named "mysync".

5. KP_B returns an answer to KP_A with the result of the lookup.

6. KP_A receives the answer and returns it to the process P.

7. Process P may now continue with the result of the remote lookup.

4.5.12 Transfer Module

The Transfer Module offers other DAMOS system modules the possibility to transfer a block of data inside the same PE, or from one PE to another.

The data may reside in the total PE address space.

When a block of data has to be moved, two transfer users are involved - a primary and a secondary user.

To be able to make use of the transfer facility, the two users have to allocate a set of transfer ports.

The primary user (the promoter) is allocating a primary transfer port using the Transfer Module function

alloc_prim_port

The PE number of the secondary user is given as parameter.

The PE number is used by the Transfer Module to decide whether to use the DMA or the SUPRA handler during the following transfer.

The Primary user conveys the allocated primary port number to the secondary user, which in turn, allocates a secondary port number, using the Transfer Module funtion

alloc_second_port

The primary port number is given as parameter.

Once a transfer port is allocated, the two users may transfer data accross this connection again and again.

When the users want to transfer a block of data from a source to a destination area, each user has to use the Transfer Module function

init_xfer

giving the following parameters:

- local port number
- flow direction
- pointer to local Transfer List

The Transfer List contains a description of the source/destination area.

When the transfer is completed the users are informed.

The users are capable to cancel all transfer requests, associated with a given port number, by using the Transfer Module function

cancel_xfer

The associated port number is given as parameter.

When the transfer users no longer want to make use of the transfer facility, the transfer ports are deallocated. This is done by using the Transfer Module function

dealloc_port

The local port number is given as parameter.

4.5.13 BASIC Transport Service

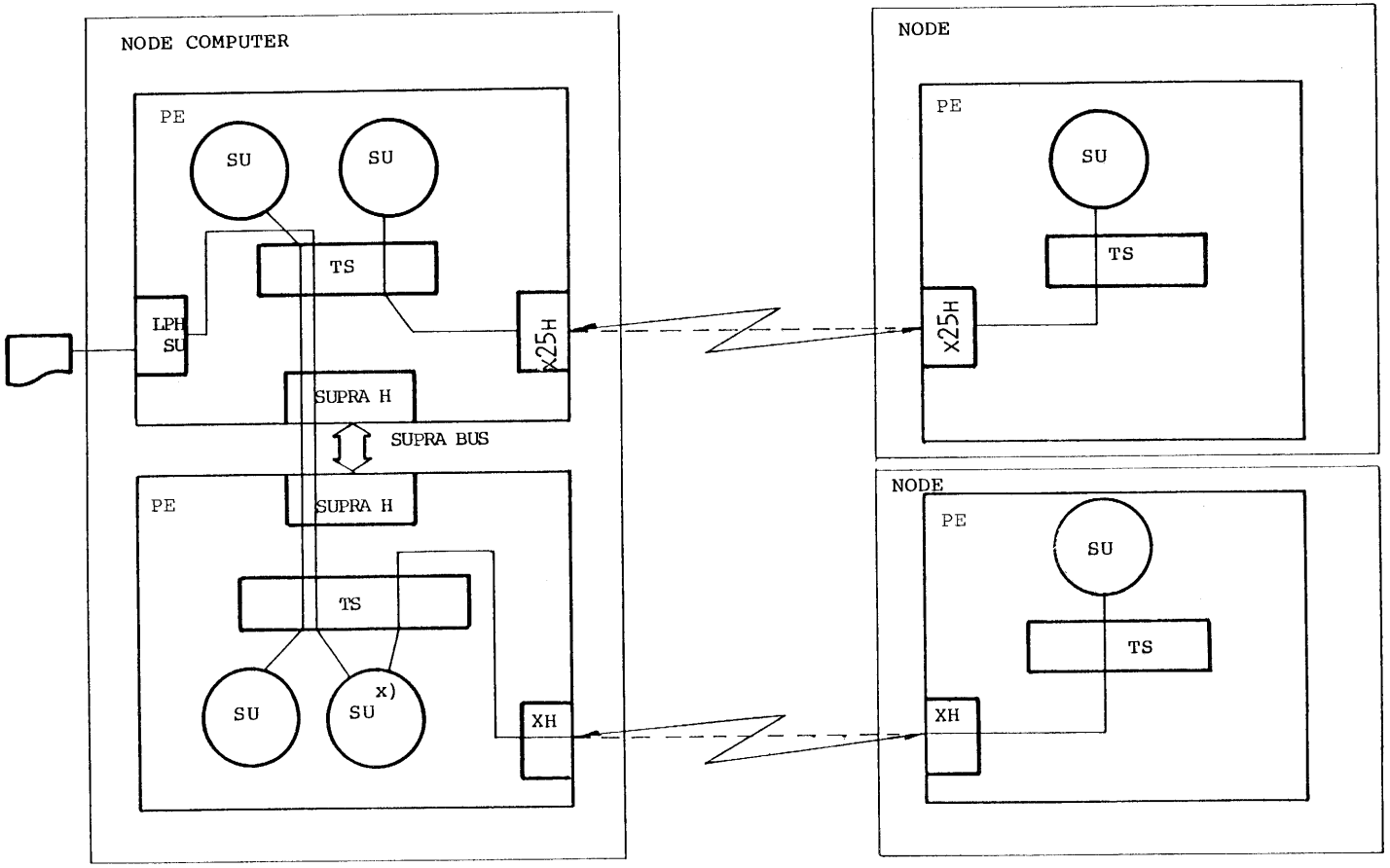
The Basic Transport Service (BTS) offers DAMOS processes and device handlers the possibility to communicate with other remote or local processes and device handlers.

Processes and device handlers - in the following description called Service Users (SU) -may be addressed indirectly via ports.

A service user can dynamically be tied to a port. When a service user wants to communicate with another service user, the former service user requests a connection to be established between (one of) his own port(s) and a port to which the remote service user is tied.

Once such a connection has been established, the two service users may exchange data and control information across the connection.

The figure overleaf depicts the possible connections within a multiple PE node.



x) Gateway process, not part of BTS

Service Types

The BTS offers two different types of service:

- stream service and
- message service

In the first type of service data flows as a logically continuous stream from one SU to the other. The blocking of data into buffers performed by the transmitting SU is not seen by the receiving SU.

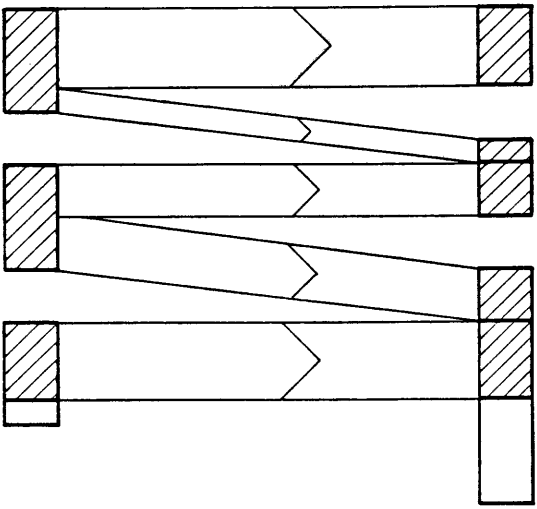
In the second type of service the buffers are treated as semantic entities and transmitted as such: i.e., a homomorph correspondence between transmit and receive buffers is enforced.

The two different situations are depicted overleaf.

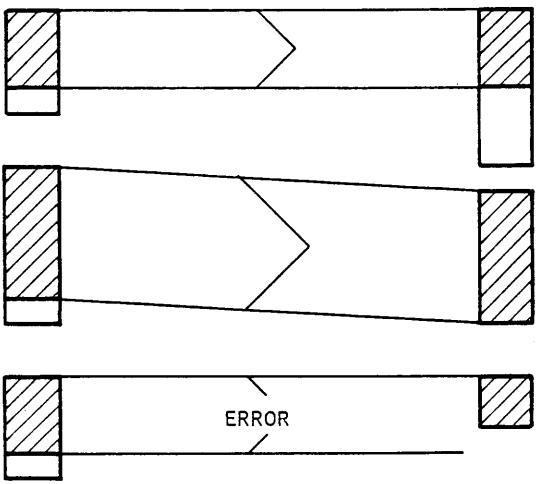
STREAM SERVICE

SENDING SU

RECEIVING SU



MESSAGE SERVICE



The BTS SU interface consists of the procedures listed below.

create_port
This procedure creates a port for the caller

delete_port
This procedure deallocates a port

connect
This procedure is called when the SU wants a connection to be established between his port and a remote port. The process or handler which is tied to the remote port is informed about the connect request.

connect_accept
The caller accepts an incoming call request.

connect_reject
The caller rejects an incoming call request

disconnect
The caller request a port connection to be deleted.

reset
The caller requests all data and control information which is in the process of being transferred across a connection to be discarded.

transmit
The caller requests data to be transmitted (sent or received) across the connection

interrupt
The caller requests a short, high priority data block to be transmitted across the connection.

port_wait

The caller requests to await events on a specified set of ports.

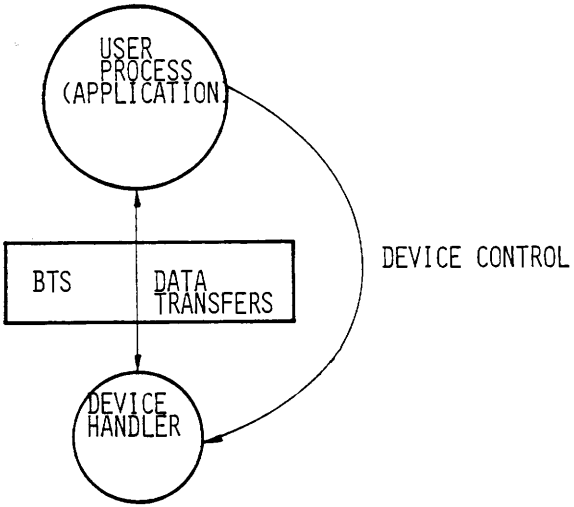
The events may be:

- interrupt received
- reset by remote caller
- disconnect by remote caller
- data transferred

4.6 DAMOS Input/Output

DAMOS supports input/output (I/O) from user programs at different levels.

At the lowest level user programs can interact with device handlers directly and transfer blocks of data by means of the Basic Transport Service module. This interface is illustrated in the figure below.



Device control is exercised via the Device Manager functions (cf section 4.5.6). Data is transfered between the user process and the device handler using a port in the user process and a port in the device handler (cf section 4.5.13).

At a higher level DAMOS offers a more structured I/O facility under the DAMOS I/O System (IOS).

The IOS provides a uniform, device independent interface for user processes to

- disk files
- magnetic tape files
- interactive terminals
- communication lines
- line printers

The IOS is a set of standard interface procedures through which a user communicates with a class of DAMOS service processes known as General File Management Systems. General File Management Systems include:

- the File Management System
  which implements disk files

- the Magnetic Tape File Management System
  for magnetic tape files

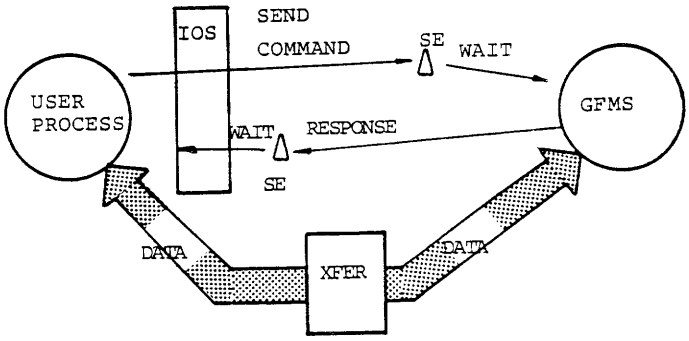
- the Terminal Management System
  for communication lines, interactive terminals and printers.

The General File Management Systems provide functions which are classified as:

- device handling
- user handling
- file handling
- file access

The communication between user processes and General File Management Systems (GFMS) is based on the DAMOS Process Communication Facility using synchronization elements (SE) for exchange of commands and responses. The DAMOS Transfer Module is used for exchange of data between the user process and the GFMS.

User processes and GFMS may be located in different PEs.



The information element sent to a GFMS must contain a specification of the command and an identification of a synchronization element via which the completion of the command is signalled.

The individual GFMS's are described in more detail in the following sections.

The common file access functions provided by the IOS are readbytes for input and appendbytes and modifybytes for output.

These basic functions are used for transfer of blocks of data.

On top of these functions the IOS provides a stream I/O facility where the IOS handles the blocking and buffering of data. A comprehensive set of functions are available for stream I/O including:

connect
Connects a stream to a file. Mode must be specified as input or output.

disconnect
Disconnects a stream from a file.

get_position
Returns the current position of the stream.

set_position
The stream is adjusted so that the next byte will have the specified position on the file to which the stream is connected.

inbyte
The next byte from the stream is delivered.

backspace
Next byte to be delivered will be the same as the last delivered

in rec
Delivers a record with a specified number of bytes.

inline
Delivers a record terminated by a New Line character (Linefeed).

inelement
Delivers a number, an identifier or a special character.

outbyte
Outputs a byte on the stream.

outnl
Outputs a New Line character on the stream.

outrec

Outputs a record with a specified number of bytes on the stream.

outline

Outputs a record with a specified number of bytes and a New Line character on the stream.

outtextp

A text string embedded in the program part is output on the stream.

outtextb

A text string embedded in the data part is output on the stream.

outhexa

A word is output on the stream as four hexadecimal digits and preceeded by a specified character

outinteger

A 16 bit word is output on the stream as decimal digits. The number may output as

signed

unsigned

The minimum number of digits output may be specified.

Leading zeroes may be suppressed.

outlonginteger

A double word (32 bit) is output on the stream. Format control is similar to outinteger.

flush

Currently buffered data are output to the file connected to the stream.

4.6.1 File Management System

The File Management System (FMS) is responsible for storing, maintaining, and retrieving information on secondary storage devices (disks).

The number and kind of devices attached to the FMS is dynamically reconfigurable.

The FMS internally consists of a set of coroutines. The user interface is handled by a coroutine which distributes the commands to internal coroutines. These coroutines are of different kinds; one kind for each access method implemented. The internal coroutines all access the disks through a disk cache coroutine. This coroutine manages a pool of sector buffers according to a priority scheme, and accesses the physical disks through device handlers only when necessary.

Device and Volume Handling

Each device known to the FMS is represented by a Device Control Block (DCB). A DCB contains the information which makes it possible for the file system to use the device. This information includes the identification of the command synchronization element of the corresponding device handler, and the kind of device.

When a volume is mounted on the device, its description is included in the DCB. Both devices and volumes may be referenced by symbolic names. The device control block is linked to all open files on the volume mounted on that device.

The file system may be given commands concerning:

- Management of peripheral devices.
  Devices may be assigned to and deassigned from the file system

dynamically. Instances of device handlers are at the same time created or deleted.

- Management of volumes.
  Volumes may be mounted on and dismounted from specific devices.

User Handling

Each process using the backing store file management system is within the file system represented by a User Control Block (UCB). The User Group Identification (UGI) is contained in the UCB. Several processes may be active under the same UGI, thus having the same access rights. The UCB contains the classification of that particular process. The UCB is linked to the file control blocks for files which are open to the user.

The file management system may be given commands concerning:

- Creation and Removal of user control blocks.

File Types

The file system supports two different organizations of files on disk. A contiguous file consists of a sequence of consecutive sectors on the disk. The size of a contiguous file is fixed at the time the file is created and cannot be extended later on. A random file consists of an index giving the addresses of areas scattered on the volume. Each area consists of a number of consecutive sectors. The number of sectors per area is determined at creation time, whereas the number of areas may increase during the lifetime of the file. The number of areas is limited to 126.

Directories

The file system uses directories to implement symbolic naming of files. If a file has been entered into a directory under a name specified by the user, it is possible to locate and use it later on. Temporary files do not need to be named. A file may be entered into several directories, perhaps under different names. Since a directory is also considered a file, it can itself be given a name and entered into another directory. This process may continue to any depth, thus enabling a hierarchical structure of file names.

File Handling

Each open file is within the file system represented by a File Control Block (FCB). The FCB contains information which makes it possible to access the corresponding file. This information includes the address of the file on the volume, its security profile, its size and its type. The FCB is linked to user control blocks for all users who operate on the file indicating at the same time their access rights to the file.

File Commands

The commands given to the file system concerning files may be grouped as:

- Creation and removal of files.
  A user may request that a file is created with a given set of attributes and put on a named volume.

- Naming of files in directories.
  A file may be entered into a directory under a symbolic name. Using that name it is possible to locate the file later on. The file may also be renamed or removed from the directory again.

- Change of access rights for a specific user group (or the public) vis a vis a file. The right to change the access rights is itself delegatable.

Security

The protection of data entrusted to the file management system is handled by two mechanisms:

The first mechanism for access control is based on the use of Access Control Lists (ACL). There is an ACL connected to each file. The ACL is a table which describes the access rights of each individual user group (one being the public) to the corresponding file. Whenever a user tries to access a file, the ACL is used to verify that he is indeed allowed to perform this access.

The second mechanism for access control is based on a security classification system. Each user and each file is assigned a classification. The user classification is recorded in the user control block and the file classification is recorded on the volume. An access to a file is only allowed if the classification levels of the user and the file match each other. (cf section 4.3).

Redundant Disks

The FMS allows use of redundant disk packs, which are updated concurrently to assure that data will not be lost in case of a hard error on one disk.

The FMS allows exclusion of one of the two identical volumes, while normal service goes on on the other one. After repair it is possible to bring up one volume to the state of the running volume, while normal service continues (perhaps with degraded performance).

The bringing up is done by making a raw copy of the good disk to that which should be brought up. While the copying takes place all read operations are directed to the good disk, whereas all write operations are directed to both disks.

Bad Sectors

The FMS is able to use a disk pack with bad sectors, unless it is sector 0.

The bad sectors are handled by keeping a translation table on each volume from each bad sector to an alternative sector.

Access Methods

The file management system implements two access methods to files:

Unstructured Access

For access purposes a file is considered simply a consecutive string of bytes. It is, therefore, a byte string which is transferred between a file and a user buffer. The user can directly access any substring in a file.

The commands which are implemented by this access methods are:

READBYTES - Read a specified byte string

MODIFYBYTES - Change a specified byte string

APPENDBYTES - Append a byte string to the end of the file.

Indexed Sequential Access

CRAM is a multi-level-index indexed sequential file access method. It features random or sequential (forward or reverse) access to records of 0 to n bytes, n depending on the selected block size, based on keys of 0-126 bytes. The collating sequence uses the binary value of the bytes so that character strings are sorted alphabetically.

CRAM supports variable key sizes as well as variable record size within the same file (or subfile, see below).

CRAM works on normal contiguous FMS files which are initialized for CRAM use by means of a special CRAM operation.

The CRAM updating philosophy is based on the execution of a batch of related updatings, which together form a consistent status change of the CRAM file, being physically updated as a single update by means of a LOCK operation. That is, after such a batch of updates, all these updates may either be forgotten (by means of the FORGET operation) or locked (by means of the LOCK operation). Both operations are performed without critical regions, i.e. without periods of CRAM data base inconsistency.

For convenience, CRAM supports subdivision of the CRAM file in up to 255 subfiles, each identified by a subfile identifier of 0-126 byte (as a key).

CRAM keeps track of the different versions of the CRAM data base by means of a 32 bit version number, which is incremented every time CRAMNEWLOCK (the locking operation) is called. This version number can only be changed by CRAMNEWLOCK (and CRAMINIT), but if the user intends to use it for some sort of unique update version stamping, it is delivered by the operations CRAMNEWOPEN,CRAMNEWLOCK, CRAMFORGETand CRAMNEWVERSION.

The file access commands supported by CRAM are:

CRAMINIT

Initializes a contiguous FMS file (which must be FMS open) to use as CRAM data base.

CRAMNEWOPEN

Opens an open FMS file for CRAM use. The files must previously have been initialized by CRAMINIT

CRAMREAD

Reads a CRAM record on base of a key and a direction, which must be one of: LT, LE, EQ, GE, GT. The direction gives the relation between the searched record compared to the given key.

CRAMUPDATE

Updates an already created CRAM record. Neither key size, key contents nor record size can be changed, only the record contents.

CRAMDELETE

Deletes the record with the given key from the data base.

CRAMPURGE

Deletes a span of records between two given keys from the data base.

CRAMCREATE

Creates a new CRAM record with a given key (and Key size) and a given record contents (and record size).

CRAMCREATESUBFILE

Introduces a new subfile with a given name to the CRAM data base. The new subfile is left closed and empty. The subfile entry includes a user defined file option record with format as a normal CRAM record.

CRAMDELETESUBFILE

Deletes a subfile from the data base. The subfile must be empty, i.e. without records.

CRAMOPENSUBFILE

Opens the subfile for CRAM operations and delivers the option record to the user (see CRAMCREATESUBFILE). A direction must be specified as for CRAMREAD.

CRAMLOOKUPSUBFILE

Get the option record of a subfile (as CRAMOPENSUBFILE), but does not open the subfile.

CRAMFORGET

Forgets a batch of CRAM updates.

CRAMBEGINLOCK

Reserved for future extension with queueing facility of update requesting processes. Presently working as CRAMFORGET.

CRAMNEWVERSION

Delivers the present version of the CRAM data base as a LONG INTEGER (32 bits).

4.6.2 Magnetic Tape File Management System

The Magnetic Tape File Management System (MTFMS) is responsible for storing and retrieving information on magnetic tapes. It is able to handle one magnetic tape controller with a maximum of 8 tape transports in daisy-chain. The driver is logically split into 3 parts:

- I/O-SYSTEM interface
- Main Processing
- Magnetic tape controller interface

Commands for the MTFMS are received by the I/O-SYSTEM interface while the controller interface implements a number of (low level) commands for handling a tape transport.

Symbolic volume names and file names are implemented through use of label records which comply with the ISO 1001 standard.

The functions of the driver can be separated into four groups:

- Device functions
- Volume functions
- File functions
- Record functions

Device functions

The following functions are defined:

- Assign a given name to a given unit of the controller.
- Deassign a given device.

Volume functions

- Initiate the tape on a given device assigning a name to it by writing a volume header label.
- Mount a given volume on a given device.
- Dismount a given volume.
- Rewind a given volume.

File functions

- Create a file on a given volume. The following information must be supplied by the caller and is written onto the tape in file header label records:
  1. File name
  2. Fixed/variable length record specification
  3. Record size.

  The file is opened for output and the given volume is reserved for the caller.

- Find a file with a given name on a given volume. The file is opened for input and the given volume is reserved.

- Skip a given number of files (backwards or forwards) on a given volume. The file at the resulting tape position is opened for input and the volume is reserved.

- Get information about the currently open file on a given volume. Information like file sequence number, record size and type (fixed/variable length) can be retrieved.

- Close currently open file on a given volume. Volume reservation is released.

Record functions

- Skip a given number of records (forwards or backwards) in a given file.
- Read a record in a given file.
- Write a record in a given file. The MTFMS performs recovery from writing errors by
  1. backspacing over the record in error
  2. erasing a fixed length of about 3.7 inches (thus increasing the record gap).
  3. attempting the writing once more.

  This procedure will be repeated maximally 10 times.

4.6.3 Terminal Management System

The TMS is a service process which provides a uniform interface for user processes to terminals attached to a CR80 system. By 'terminal' is meant a device which is basically sequentially accessed. Examples of terminals are:

- interactive terminals (screen or hardcopy)
- virtual circuits (via data communication equipment)
- line printers

Terminals may be attached to LTUs, LTUXs (via the TDX system), the MAP serial communication I/F, and parallel interface controllers.

The functions performed by the TMS fall in the following main categories:

- management of device controllers
- management of terminals
- management of users
- control of access to terminals
- transfer of data between user processes and terminals

The basic TMS data types are:

- users
- devices
- channels
- terminals and
- connections

The characteristics of these elements are described in the following.

Channels and terminals are called subdevices.

Devices

The TMS data type 'device' corresponds to a physical device controller.

The TMS supports terminals and communication lines which are interfaced to the PE via the following device controllers:

a. LTUXs attached to the PE via a TDX bus and a STI

b. LTUs attached to the PE Data channel

c. Parallel interface controllers

d. The serial MAP communication chennel

LTUXs are accessed via the STI device handlers.

LTUs are accessed via the LTU device handler.

LTUs are special in the sense that their initialization require loading of a program into the LTU.

The TMS provides an interface function for loading of an LTU, but the load module has to be provided by a user process external to the TMS.

LTUXs may require set-up of TDX connections.

Parallel interface controllers (PIC) are accessed via the LP device handler.

The MAP serial communication channel is accessed via the OC device handler.

Status and Error Reporting

Status and error information concerning a device (LTU, LTUX, PIC) is sent asynchronously to the creator of the device via a Status Synchronization Element.

Generic Device Structure

The devices supported by the TMS have similar structures. Each device provides a number of channels which are physical interface connections for external peripheral devices or internal 'devices' (like for instance LTU debuggers). Each channel may interface to data communication equipment (DCE), a single physical terminal (interactive terminal, printer) or several terminals.

Examples on device structures are shown in figures 4.6.3-1 and 4.6.3-2.

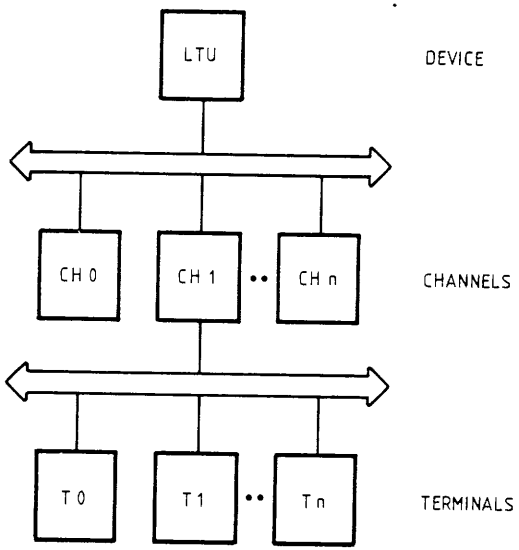


Fig. 4.6.3 - 1 LTU device structure

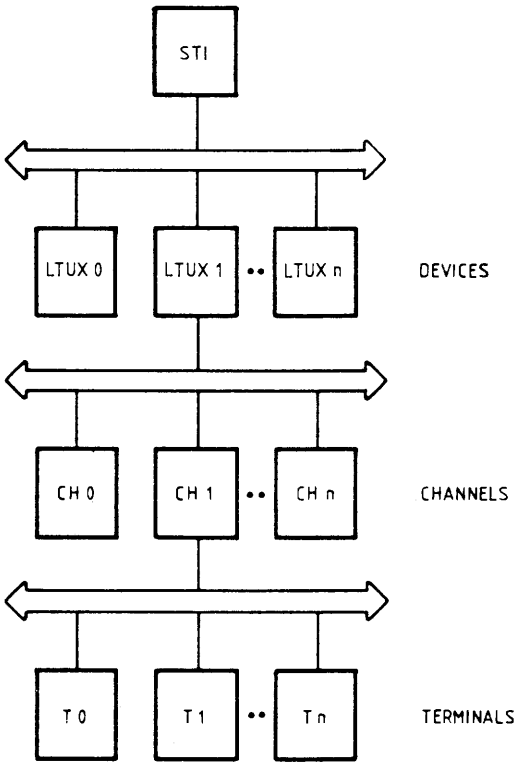


Fig. 4.6.3 -2 TDX device structure

Device States

Devices may be in one of the following states:

unassigned: the device is not accessible for other commands than assign.

assigned: the device has been assigned, but has no channels open

channels open: the device is assigned and has channels open

Channels

Channels may be single channels or multiplexed channels. Multiplexing may correspond to a physical multitude of terminals as in the case of multidropped terminals on a single line or to logical subchannels as in the case of virtual circuits on an X25 communication line.

Status and Error Reporting

Status and error information concerning a channel is sent asynchronously to the creator of the channel via a Status Synchronization Element.
The Status Synchronization Element may be common for several channels and/or devices.

Terminals

Terminals are the ultimate addressable objects which act as the source of input data to users and sink of output data:

TMS terminals correspond to subdevices in the device handlers.

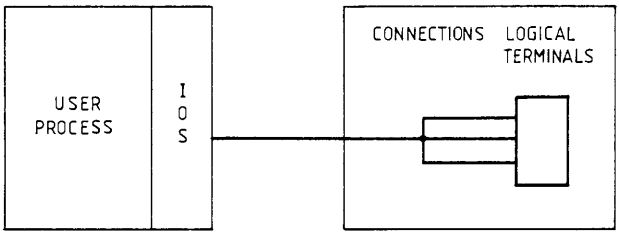
The TMS assumes a uniform interface to the device handlers supported by it. The device handlers are assumed to implement an error free, block oriented sequential link to the terminals.

The input from a terminal is a sequence of blocks of bytes as is the output. The TMS is transparent to the possible dependency between input blocks and output blocks.

Terminals may be shared by several user processes at the same time.

Connections

A user process communicates with channels and terminals via connections. From a user's point of view, a connection is syntactically equivalent to an FMS file. The connection is the object referred by the user process when it f.i. outputs data to the terminal.



A connection is obtained when a channel or a terminal is created. Further connections can be obtained by 'lookup' of an already open channel or device or by an inheritance mechanism whereby the parent 'offers' a connection to the child process who in turn 'accepts' the connection.

The connection is used for exchange of protocol information with the channel or terminal or for exchange of input/output data with the terminal.

Selection of Connections

Only one connection can talk to a terminal at a time. In order to be able to exchange data with the terminal a connection must be selected.

Two different schemes may be employed for controlling the selection of connections. The scheme is defined individually for each terminal when the terminal is created.

In the first scheme the creator of the terminal is in complete control of which connection is currently selected.

This control is exercised by means of a select command. The basis for selecting a connection may be status information from the terminal received via a status synchronization element.

In the second scheme the TMS controls which connection is selected. This scheme may be used for TTY compatible scroll mode terminals.

The selection may be triggered from the terminal by means of a break status or triggered by the combined state of pending input and output requests for the terminal:

1. The operator at the terminal may select a connection which exists and which has pending input requests. This is done by inputting a 'break' from the terminal. In response to the break the TMS outputs a connection select interrogation message (TO:) and waits for a connection select message (<user name>). This message defines the connection to be used.
   The connection is identified by the symbolic name of the corresponding user process. This name was defined when the user was logged on to the TMS. (If the user has several connections to the same terminal, the choice is arbitrary).

2. The currently selected connection has placed no input or output requests for the last X seconds, and other connections to the terminal have pending output requests. In this case the TMS selects a connection with a pending output request and preambles the output with a connection select header (FM: <user name>).
X is defined when TMS is initialized.

Status and Error Reporting

Status and error information concerning a terminal is sent to the creator of the terminal via a status synchronization element.

The status input is defined by the terminal and is device dependent.

The status type includes:

- break (only if the creator is in control of connection selection)
- off line

Whenever status input is received from the terminal, the terminal is disabled for all connections. The creator of the terminal will have to explicitly enable the terminal (if the TMS is in control of connection selection) or to select a connection (if the creator is in control of connection selection).

If the generation of the status input is associated with a disruption of input or output, the terminal may return a completion code to that effect to the TMS, which in turn will pass on the completion code to the user whose input or output was disrupted.

Terminal States

TMS terminals may be in the following states:

not created:	the terminal is only accessible to a create_terminal command
disabled:	the terminal is created but not accessible for input/output
enabled:	the terminal is created and is accessible for input/output

Flow Control

Flow control is exercised by the user process. Input data from the terminal is not accepted until a user process is prepared to receive it.

Users

The users of the TMS are user processes requesting service by the TMS via the IOS.

Before a user can have requests served by the TMS, the user must have been logged on to the TMS. This is performed by another user who is already logged on.

The TMS recognized two categories of users:

- system users, and
- ordinary users

System users are allowed to

- log on other users to the TMS,

When a user is logged on, a symbolic name is defined for the user, and a number of TMS resources are allocated for the user.

Access Control and Access Mechanisms

The objects which may be accessed via the TMS are:

- devices
- channels
- terminals
- connections

Devices are referenced by logical device numbers while terminals and channels are referenced by symbolic names. Connections are only referenced via connection descriptors.

A device is accessed in order to

- delete the device, or
- create channels on the device

Only the creator of the device is allowed to delete it.

A channel is accessed in order to

- delete the channel, or
- control the channel, or
- create terminals on the channel

Only the creator of the channel is allowed to delete and control it.

A terminal is accessed in order to

- delete the terminal, or
- control the terminal, or
- obtain a connection to the terminal for data input and output.

Only the creator of the terminal is allowed to delete and control the terminal whereas all other users may access the terminal in order to attempt to obtain a connection to it.

Obtaining a connection to a terminal is subject to an access check, which is based on:

- a security check and
- a discretionary access check

The discretionary access check employs an ACL associated with the terminal. The ACL enumerates the access rights of specific user groups identified by theri UGI.

Access rights include the right to:

- input from the terminal,
- output to the terminal,
- define the access rights of other users (protect),

A user process may obtain a connection to a terminal

- by looking up the terminal based on its symbolic name
- by accepting an offered connection

Security

Access to terminals under the regime of the TMS is subject to a security check.

Terminals and user processes are associated with a security profile, and the security policy employed is:

1. a user may input from a terminal with a profile lower than or equal to that of the user

2. a user may output to a terminal with a profile higher than or equal that of the user

3. a trusted process may output to any terminal

The security profile for the terminal may be changed dynamically by the creator of the terminal.

If the current profile is changed, all connections to the terminal are reevaluated.

Recovery

The HW supported by the TMS may be redundant in the following senses:

1. a TDX bus may be dualized, each bus interfaced via a separate TIA to the same STI

2. a group of LTUs may have a common spare LTU which may replace any of the other LTUs in the group in case of a single LTU failure.

The dualization of the TDX bus is transparent to the TMS as it is handled at the STI handler level.

Spare LTUs are supported via the TMS. However, switching to a spare LTU must be requested explicitly by the creator of the failing LTU.

The decision to switch is normally but not necessarily based on status information from the LU handler.

After the switch, four recovery actions must be performed:

1. recovery of the LTU firmware
2. recovery of the LTU configurational information
3. recovery of the input in progress
4. recovery of the output in progress

Recovery of LTU Firmware

This is the responsibility of the user process that originally assigned and loaded the failing LTU.

The spare LTU must be boot loaded with the same boot load module which was used for loading the previously active LTU. This boot module must therefore have been saved by the creator.

Recovery of LTU Configurational Information

The LTU is reconfigured, i.e. channels and terminals are recreated based on the TMS data structures, which describe the previous LTU.

This is a TMS responsibility.

Recovery of Input in Progress

This is a TMS responsibility.

The input requests which were pending in the LTU when the switching occurred, are redirected to the spare LTU.

Recovery of Output in Progress

This is a TMS responsibility.

Output requests which were pending in the previously active LTU are redirected to the spare LTU.

4.7 System Initialization

When a CR80 memory mapped PE is master cleared, a boot strap loader is given control.

The boot strap loader is contained in a programmed read-only memory which is part of the MAP module. Having initialized the translation tables of the MAP module, the boot strap loader is able to fetch a system load module from a disk connected to the PE.

The selection of the disk and the load module is determined by a load command which the operator may enter on the operator's console.

The system load module is placed in the main memory of the PE by the boot strap loader and given control. An initialization module which is part of the load module initializes the DAMOS kernel and the DAMOS Root process.

The Root process possesses all the PE resources. At system generation time the Root Process is given a task list which may direct it to perform the following initialization actions:

- create objects like synchronization elements and device handlers.
- catalog objects
- load and create processes, including
  - File Management Systems
  - Terminal Management Systems
  - Magnetic Tape File Management Systems
  - load and create an operating system with a specified set of resources

4.8 Operating System for Software Development

TOS is an operating system which supports interactive terminal users in a program development environment.

The functions performed by TOS are invoked by two types of requests:

Operator commands, which are messages typed at terminals and sent to TOS. The functions which may be performed in response to these requests are:

- assign/deassign disk devices
- mount/dismount volumes on disk drives
- include terminals in the system/remove terminals from the system
- log on to the system
- remove processes from the system
- broadcast messages to terminals
- manipulate a "news" message facility
- present status information
- run a task
- close the system

Programmed requests are sent from processes to TOS. The functions which may be performed in response to these requests are:

- allocate resources (memory) for a task, load a program, and create a process to execute the program
- start a process
- stop a process
- restart a process
- logout from the system
- reserve a print queue file semaphore
- release a print queue file semaphore
- start a printer task

TOS operates in two phases: a system initialization phase where parameters like the identity of the system disk may be (re)defined and a production phase where users may log on to the system and have tasks executed.

Initialization phase commands are:

SYSDIR	<file id>
SYSDEV	<disk assignment params>
MODE	M
	S
OPEN	

The production phase commands are:

ASSIGN-TERMINAL	<terminal assignment params>
ASSIGNDISK	<disk assignment params>
DEASSIGNDISK	<device name>
RESERVEDISK	<device name> FOR <process name>
RELEASEDISK	<device name>
MOUNT VOL	<volume name> ON <devicename>
DISMOUNT	<volume name>
UPDATE	<volume name>
RUN	<file id>
REMOVE	<process name>
STATUS	
DO	<file name>
NEWS	<text>
CLEARNEWS	
BROADCAST	<text>
LOGIN	<user name> <old password> <new password>
RELEASE	
CLOSE	

When a user logs on to the system, a Command Interpreter process is created to handle his commands.

For further information: CSS/380/USM/0026.

Command Interpreter(CMI)

The CMI reads input lines from the current input file and scans the line for commands.

When a non-recognized command is found it is assumed to be a file name for a program to be loaded and the remaining portion of the command line is assumed to be parameters to be passed on to that program. The file is looked up, and if found, a request is sent to the TOS for loading the program and for creating and starting a process to execute the program. This process is hereafter referred to as a task.

If the program file cannot be found, the remaining portion of the line is skipped.

The CMI will repeatedly process command lines as long as it has no active tasks. When one or more tasks are active, the CMI stops processing command lines unless an attention is sent to is (by pressing the break key of a terminal and typing the name of the CMI process). When an attention signal is received by the CMI, the CMI will read and process one more command line.

The reading and processing of command lines is resumed when the last active task invoked by the CMI terminates.

The CMI accepts the following commands from the current input file:

USE	<file id>	changes the current directory file
DO	<file id>	changes the current input file
WHAT		

		types the name of the directory file
LOAD	<file id>	load a task
START	<process name> <program params>	initiate a loaded task
STOP	<process name>	suspend a task
CONTINUE	<process name>	resume a task
LOGOUT		terminate session
<file id> <program params>		load and initiate a task using the file as a load image

The character string <program params> are essentially passed on to the task without modifications. However, the following constructs will be processed as described:

construct		processing
>		the character is deleted and the file id for the current output file is inserted
<		the character is deleted and the file id for the current input file is inserted
CIN:	<file id>	the current input file for the task is redefined
COUT:	<file id>	the current output file for the task is redefined

For further information: CSS/381/USM/0037.

4.9 Programming Languages

The following programming languages are available for the CR80 computers:

- SWELL, which is an intermediate level systems programming language
- PASCAL, which is a general purpose high level programming language
- COBOL, which is a high level language for administrative application programming

In addition, the new DOD standard programming language ADA will be available for the CR80 computers from 1984.

The programming languages are briefly described in the following sections.

4.9.1 SWELL

SWELL is the system programming language for the CR80 series of computers. In the design of SWELL, the programming language PASCAL, has been used as a model, because of the facilities PASCAL offers for systematic and structured programming. However, in SWELL, program execution is not supported by a run time system, and no facilities that might hide the architecture of the computer are implemented. Therefore, only constructs that have a direct and efficient implementation on the CR80 are included in the language.

In the following sections the elements of SWELL are presented. The language is mainly illustrated by examples. The intent is to give a general idea of the level and the facilities of SWELL. A precise definition of the language is given in the SWELL reference manual.

Constants and Type Definitions

Constant and type definitions are abstract ways of describing the data structures on which a program is to operate. They result in no code generation and no allocation of data space in the object program. SWELL has almost directly adopted these program parts from PASCAL.

One difference from PASCAL is that an expression, involving constant operands only, is considered a constant and may be used to define other constants.

Examples of constants and constant definitions:

```
CONST
    LASTINDEX = n-1;
    AREA      = LENGTH*WIDTH;
    NL        = '(:10:)';
    VERSION   = 'VERSION 1';
```

The standard data types of SWELL represent the units accessible in one instruction on the CR80:

BYTE is an unsigned integer, which can be held in a byte (8 bits)

CHAR is equivalent to BYTE

INTEGER is an integer, which can be held in one word (16 bits). There is no standard interpretation of this data type. The interpretation depends on the operations performed in variables of this type. But normally INTEGERs are used as unsigned integers or as signed integers using 2s-complement arithmetic.

LONG is an integer which can be held in a double word (32 bits). As only a few operations on double word operands are supported by the CR80 directly, the type is 'predeclared' as:

```
LONG=
  RECORD
    LEAST, MOST: INTEGER
END;
```

Scalar types and subrange types are defined by using the same notation as in PASCAL, e.g.

```
TYPE
  BOOLEAN= (FALSE, TRUE);
  ALFA = 'A'..'Z';
```

The rules for type compatibility are much more liberal in SWELL than in PASCAL. This is due to the explicit use of registers. Very few operations may be carried out without involving registers, so it would be very inconvenient if the type of a register did not match all other scalar and subrange types.

The following definitions are therefore equivalent:

1)
```
TYPE
    WORKDAY = (MONDAY, TUESDAY, WEDNESDAY, THURSDAY,
    FRIDAY);
```

2)
```
CONST
    MONDAY    = 0;
    TUESDAY   = 1;
    WEDNESDAY = 2;
    THURSDAY  = 3;
    FRIDAY    = 4;

TYPE
    WORKDAY = MONDAY .... FRIDAY;
```

Array types and record types may also be defined in SWELL.

Examples:

```
TYPE
    HOURSWORKED = ARRAY [WORKDAY] OF INTEGER;

    DAYTYPE=
        RECORD
            DAY:   1..31;
            MONTH: 1..12;
            YEAR:  INTEGER
        END;

    PERSON =
        NAME: ARRAY [1...NAMELENGTH] OF CHAR;
        BIRTHDAY: DAYTYPE
    END,
```

Pointer types are not included in SWELL. The reason is that in low level languages pointers are often used in arithmetic expressions. Besides, the pointers are usually held in general purpose registers, to be used for register indirect addressing (index addressing) of record components, etc. A distinction between INTEGER and pointer types would therefore be inconvenient, and consequently pointer variables must be declared as INTEGER.

To support the use of pointers (e.g. for dynamic memory allocation), SWELL includes an ADDRESS operator giving the address of a variable and a SIZE operator giving the size of a data type.

Examples:

```
SIZE (DAYTYPE)
ADDRESS (HASHTABLE)
```

Variables

Variables may be declared and initialized in SWELL. But as no run time system exists, allocation of variables is not performed dynamically. Space for all declared variables is allocated at compile time.

Example of variable declarations:

```
VAR
    I, J: INTEGER;
    L: LONG;
    OPERATOR: (PLUS, MINUS, TIMES);
    HASHTABLE: ARRAY [0..HASHSIZE-1] OF INTEGER;
```

The general purpose registers are predeclared as variables of type INTEGER, with the names:

R0, R1, R2, R3, R4, R5, R6, R7

The same variables may be used as variables of type LONG. In this case two registers are concatenated and used as one variable. The names of these standard variables are:

R01, R12, R23, R34, R45, R56, R67, R70

A declared variable is denoted by its name, e.g.:

```
I
HASHTABLE
```

A component of an array variable is denoted by the variable followed by an index:

```
HASHTABLE [R4] ;
HASHTABLE [17];
```

Correspondingly a component of a record is denoted by the variable followed by a field identifier, e.g.:

```
L.LEAST;
```

In SWELL, the notation of a variable referenced by a pointer differs from the notation used in PASCAL. This is so because a pointer is not associated with variables of a specific type. In SWELL, pointers must therefore be qualified explicitly when they are used. This qualification is denoted by writing a type identifier after the pointer variable like this:

```
R4@ DAYTYPE.MONTH;
P@ INTEGER;
```

If a register is used as a pointer variable, a referenced variable may be addressed in a very efficient way on a CR80, as the addressing may be performed by an index register.

Expressions

The notation for expressions aims at supplying the programmer with a tool that has the same expressive power as the corresponding constructs of high level languages. In SWELL, no part of the data space is 'hidden' from the programmer to store intermediate results. But since arithmetic and logic instructions on a CR80 change one of the source operands, an expression involving variables is also an implicit assignment. The expression:

```
R0+7
```

takes the value in R0, adds 7 and stores the result in R0.

If an expression has more than two terms, the destination operand is used as source operand in a continued expression. Thus the construction:

```
R0+R1+R4@ INTEGER
```

is equivalent to

```
R0+R1;
R0+R4@ INTEGER;
```

The scheme for expressions has some disadvantages. Only operations directly supported by the computer may be carried out.

An expression like:

```
I+J;
```

is not allowed, if I and J are simple variables declared as type INTEGER, because CR80 does not support that kind of memory to memory addition.

On the other hand the notation used in SWELL is superior to the notation used in assembly language programming, and this is achieved without losing control of the instructions generated by the compiler.

Since all expressions have embedded an implicit assignment operation, SWELL makes no sharp distinction between an expression and an assignment. If the contents of a variable V are to be moved to R0 the construction below will apply:

```
V=> R0;
```

As mentioned above, the CR80 has no instruction that adds two simple, memory resident variables. If such an operation is to be performed, a register must be used as a work location. The operation may be expressed:

```
I=> R0+J=> I;
```

The '=> ' is used instead of the wellknown ':=' to indicate that all operations are carried out from left to right.

Relations

A relational expression in SWELL is implemented by means of skip and jump instructions. Therefore, the result is not explicitly accessible and cannot be used as an operand. It may only be used as a controlling condition within a statement (conditional or repetitive). A relational operator is included in SWELL only if a corresponding skip instruction exists on the CR80. This means that the following relational operators are allowed:

=	is equal to
<>	is not equal to
>=	is greater than or equal to, signed
<	is less than, signed
>>=	is greater than or equal to, unsigned
<<	is less than, unsigned.

The relations may be combined by logical operators (not, logical and, logical or), and nesting may be specified. The facilities offered are thus far beyond

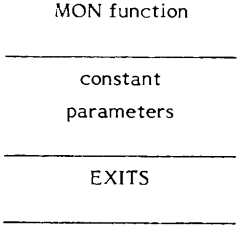
the facilities of assembly languages.The code becomes very compact and efficient and corresponds to the code of an assembly language program, written by a careful programmer.

Examples

```
R0 = 1 LOGAND R0 < 10
(V=> R0) = TRUE LOGOR (U=> R0) = TRUE
```

Procedures

On a CR80 a process interfaces to other parts of the system by means of a special supervisor call instruction (MON). Traditionally the format of a system call looks like this:



The constant parameters are call information, and the exit table is used for signalling the result of the system call. In addition, call and return information may be exchanged via registers. A procedure call in SWELL has the same format as that of a system call, and as no runtime system is present, no parameter transfer facility beyond this level may be implemented. What can be done further is a compile time check of the parameter conventions. When a SWELL procedure is declared the constant parameters, the exits, and the use of the registers are defined in the procedure heading, as in the example below:

```
PROCEDURE BYTES_EQUAL
        (NO_OF_BYTES: INTEGER;
        R4;      " ADDRESS OF FIRST RECORD
        R5;      " ADDRESS OF SECOND RECORD
        R6)      " LINK REGISTER
        : BOOLEAN; " EXITS (RETURN):
                "   TRUE   BYTE STRINGS EQUAL
                "   FALSE  BYTE STRINGS NOT EQUAL
```

A procedure is activated by a procedure statement. The parameters in the call must correspond to the conventions defined in the procedure declaration:

```
BYTES_EQUAL(SIZE(FILE_NAME_TYPE),
        ADDRESS(FILE_NAME1)=> R4,
        ADDRESS(FILE_NAME2)=> R5,
        R6);
```

If the actual procedure exit is significant (as in the case of BYTES_EQUAL), the procedure may be activated by a switch statement. A switch statement corressponds to a case statement, but the selector is functionally replaced by the call of a procedure with multiple exits:

```
SWITCH BYTES_EQUAL (10, R4,R5) TO
 FALSE:
  BEGIN "EXCHANGE POINTERS"
   R4=> R6; R5=> R4; R6=> R5;
  END;
 TRUE;
END;
```

No implicit exit code is generated at the end of a procedure body. It is the programmer's responsibility that a procedure is left properly. This is done by means of an exit statement, where the contents of the link register (after procedure activation) and the exit number are specified. This may appear as in the example below:

```
EXIT(R6, FALSE);
```

Statements

Apart from the expressions and the statements concerning procedures, the following kinds of statements exist in SWELL:

1) Compound statement
2) If statement
3) While statement
4) Repeat statement
5) Case statement
6) Goto statement
7) Dedicated statement

The 'Compound statement', the 'If statement', the 'while statement', and the 'repeat statement' are adopted syntactically and semantically from PASCAL.

Examples:

```
BEGIN
 IF R0 = 7 THEN 1=> R7;

 TOP=> R4; 0=> R0;
 WHILE R4-1 >= FIRST DO
  R0=> TABLE [R4];

 REPEAT GETBYTE(R0, R6) UNTIL R0 = NL;
END;
```

As in PASCAL, the case statement consists of an expression (the selector) and a list of statements, each being labelled by a constant or a constant identifier. In addition (as in some PASCAL dialects) an OTHERWISE part may be specified, where all cases not explicitly mentioned in the label list are trapped.

Example:

```
CASE R0 OF
  SUNDAY: GOTO CHURCH;
  SATURDAY: GOTO SHOPS;
OTHERWISE GOTO WORK;
```

A 'goto statement' performs an unconditional jump to a label. In SWELL a label is named by an identifier. The 'goto statement' need only be used as rarely in a SWELL program as in programs written in high level languages.

A 'dedicated statement' is a statement dedicated for the execution of one specific CR80 machine instruction. Such a statement may be used to gain access to special features of the CR80 hardware, e.g. a control operation on an external device. The syntax of a dedicated statement is very similar to the syntax of a procedure statement, except for the missing link register.

Example:

```
SI0(R1, DEV= R0); "SENSE DEVICE STATUS"
```

The Compiler

The SWELL compiler consists of three main passes and a service pass. All passes are executed in the same program and data areas, using an overlay technique. The compiler might be characterized as a single-pass compiler for a virtual CR80 (executing pseudo instructions, using logical references), followed by a two pass code generator. The main duties of each individual pass are listed below:

Pass 1: lexical analysis
syntax analysis
name analysis
semantic analysis

Pass 2: preliminary code selection
worst case address allocation
error reporting

Pass 3: final code selection
code optimizing
final address allocation

Pass 4: generation of cross references
generation of debug information

During pass 1 the program source text is transformed into an intermediate form. The intermediate code includes operators and symbolic operands, leaving it to the subsequent passes to find CR80 instructions performing the operations on operands of the specified kinds (registers, constants, etc.). All declarations are totally consumed by pass 1 and all structures are broken down into simple context-free pseudoinstructions. It should be mentioned that the compiler produces link modules, and the final code assembly is performed by a linkage editor.

4.9.2 CR80 PASCAL

A CR80 PASCAL program consists of two essential parts, a description of the actions to be performed, and a description of the data which are manipulated by these actions. Actions are described by statements, and data are described by declarations and definitions.

The data are represented by values of variables. Every variable occurring in a statement must be introduced by a variable declaration which associates an identifier and a data type with that variable. The data type essentially defines the set of values which may be assumed by that variable. A data type may in CR80 PASCAL be either directly described in the variable declaration, or it may be referenced by a type identifier, in which case this identifier must be described by an explicit type definition or be one of the standard type identifiers BOOLEAN, INTEGER, CHAR og LONG_INTEGER.

Enumerated types are defined by indication of an ordered set of values, i.e. by introduction of identifiers standing for each value of the type.

A type may also be defined as a subrange of one of the types INTEGER, CHAR or BOOLEAN.

Structured types are defined by describing the types of their components and by indicating a structuring method. The various structuring methods differ in the selection mechanism serving to select the components of a variable of the structured type. In CR80 PASCAL there are three basic structuring methods available: array structure, record structure, and set structure.

In an array structure all components are of the same type. A component is selected by an array selector, or index, whose type is indicated in the array type definition. Given a value of the index type, an array selector yields a value of the component type. Every array variable can therefore be regarded as a mapping of the index type onto the component type.

In a record structure the components (called fields) are not necessarily of the same type. Each component has attached to it an identifier (declared in the record type definition) which is used when the component is selected.

A record type may be specified as consisting of several variants. This implies that different variables, although said to be of the same type, may assume structures which differ in a certain manner. The difference may consist of a different number and different types of components. The variant which is assumed by the current value of a record variable is indicated by a component field which is common to all variants and is called the tag field.

A set structure defines the set of values which is the power set of its base type, i.e. the set of all subsets of values of the base type.

Variables declared in explicit declarations are called static. The declaration associates an identifier with the variable which is used to refer to the variable. In contrast, variables may be generated by an executable statement. Such a dynamic generation yields a pointer, which subsequently serves to refer to the variable. This pointer may be assigned to other variables, namely variables of type pointer. Every pointer variable may assume values pointing to variables of the same type T only, and it is said to be bound to this type T. It may, however, also assume the value NIL, which points to no variable.

The most fundamental statement is the assignment statement. It specifies that a newly computed value be assigned to a variable, or a component of a variable. The value is obtained by evaluating an expression. Expressions consist of variables, constants, sets, operators, and functions operating on the denoted quantities and producing new values. CR80 PASCAL defines a fixed set of operators, each of which can be regarded as describing a mapping from the operand types into the result type. The set of operators is subdivided into groups of arithmetic operators (addition, subtraction, sign inversion, multiplication, division, and computing the remainder), boolean operators (negation, union, and set difference), and relational operators (equality, inequality, ordering, set membership, and set inclusion).

The procedure statement causes the execution of the designated procedure (see below). Assignment and procedure statements are the components or building blocks of structured statements, which specify sequential, selective, or repeated execution of their components. Sequential execution of statements is specified by the compound statement, conditional or selective execution by the if statement and the case statement, and repeated execution by the repeat statement, the while statement, and the for statement. The if statement serves to make the execution of a statement dependent on the value of a boolean expression, and the case statement allows for the selection among many statements according to the value of a selector. The for statement is used when the number of iterations is known beforehand, and the repeat and while statements are used otherwise.

A statement can be given a name, and be referenced through that name. The statement is then called a procedure, and its declaration a procedure declaration. Such a declaration may additionally contain a set of variable declarations and type definitions. The variables and types thus introduced can be referenced only within the procedure itself, and are therefore called local to the procedure. Their identifiers have significance only within the program text which constitutes the procedure declaration and which is called the scope of these identifiers. Entities which are declares in the main program, i.e. not local to some procedure, are called global. A procedure has a fixed number of parameters (if any), each of which is denoted within the procedure by an identifier called the formal parameter. Upon an activation of the procedure statement, an actual quantity has to be indicated for each formal parameter. This quantity is called the actual parameter.

A function is declared analogously to a procedure. The only difference lies in the fact that a function yields a result the type of which must be specified in the function declaration. Functions may therefore be used as constituents of expressions.

Separate Compilation

The CR80 Pascal language includes a facility for separate compilation. Compilation units are submodules or main modules. A Pascal main module may be linked with Pascal and/or SWELL.

4.9.3 CR80 COBOL

The CR80 Cobol compiler is an industry compatible efficient two-pass compiler, fulfilling American National Standard X3.23-1974 level 1, with nearly all features of level 2.

The CR80 Cobol system consists of three main modules:

- compiler interpreter
- runtime interpreter
- subprogram linker

The compiler interpreter interprets the compiler phases thus compiling a source text into a pseudo object module to be processed by the runtime interpreter.

The linker allows separately compiled Cobol modules to be linked into an executable object module.

CR80 Cobol supports the following level 2 features:

- level numbers 77, 01-49, 88, 66
- value series of range, level 88 conditions
- logical AND, OR and NOT in conditions
- algebraic relational symbols
- implied subject or both subject and relation in relational conditions
- ACCEPT DATE/DAY/TIME
- STRING and UNSTRING statements
- all-digit procedure names
- multiple receiving fields in COMPUTE statements
- PERFORM statements with all formats
- mnemonic names for ACCEPT or DISPLAY devices
- qualification of names
- sign test

- nested IF statements
- paranthesis in conditions
- continuation lines
- all formats of ADD, SUBTRACT, MULTIPLY, DIVIDE including REMAINDER and multiple receiving fields
- variable COLLATING SEQUENCE
- exponentiation in COMPUTE
- data substitution in DATA-COMPILED
- figurative constants of the form ALL "character" and ALL figura tive constants
- multiple operands in ALTER statements
- GO TO statement with omitted operand
- ADD, SUBTRACT and MOVE CORRESPONDING

Description

The CR80 Cobol system includes the standard ANSI level 1 and 2 features as specified above, and further supports:

SEQUENTIAL I/O for access of data files in an established sequence containing all of level 2, except BLOCK clause, LINAGE clause, RERUN, MULTIPLE FILE TAPE, REVERSED OPEN and mnemonic-name ADVANCING are not supported.

RELATIVE I/O for access of relative files, including dynamic access containing all of level 2, except BLOCK clause, RETURN and DELETE statements are not supported. Relative text files are also not supported.

INDEXED I/O for access of index sequential files through use of the File Management System CRAM function.

TABLE HANDLING for defining tables of data and access to elements therein. All of level 2 except DEPENDING in OCCURS is available.

SORT providing the capability of sorting a file, including procedures for manipulating the file before and after sorting, and containing all of level 2, except the SAME clause and COLLATING SEQUENCE clause are not supported.

DEBUG providing options for insight into program flow and selected data item value, except that qualifiers and subscripts are not automatically placed in DEBUG items.

SUBPROGRAM capability (inter-program communication) for combining separately compiled Cobol programs into a single executable load module (level 1).

LIBRARY for copying source text into COBOL source program.

4.9.4 ADA

ADA is a high order programming language suited for a wide application domain, e.g. numerical applications, system programming applications and embedded computer system applications including real-time applications.

ADA was designed on behalf of US Department of Defence at CII-Honeywell-Bull. The language was named to honour the world's first programmer Ada Augusta, Lady Lovelace, colleague of Charles Babbage, and the daughter of Lord Byron.

The language belongs to the Algol family by being block structured, but extends the program structuring by providing 'modules', which are collections of data types, constants, variables, subprograms and modules. Modules and subprograms can be compiled separately without loss of security and made available to more users through a common library. One kind of module provided is the 'task' which allows a program to be divided into more processes running in parallel on muti-processors or in quasi-parallel on single-processors.

Combined with the facilities in ADA for handling hardware and software exceptions (interrupts) as well as for reading the clock, the tasking facilities make the language well suited for embedded (and/or) real-time systems.

The language provides simple data types such as integer, character, enumeration, real, and access (reference), as well as composite such as arrays and records with elements of simple and composite types. Furthermore, the programmer can define his own enumeration types as well as subtypes of any of the types. As ADA is strongly typed a large amount of checking can be done at compile time.

Representation specifications can be used for specification of the mapping between data types and the concrete representation, e.g. how the components of a record are to be placed in storage.

ADA supports hierarchical (top-down) as well as configurative (bottom-up) program development through its module concept and the facilities for separate compilation. For example, within a subprogram or module where a subunit (subprogram or module) is declared, the corresponding body can be represented by a body-stub. The body itself can then be developed later and compiled separately.

An ADA compiler hosted on and targetted for the CR80 is currently under development. It is scheduled for release in the first half of 1984.

4.10 Utility Software

The DAMOS Support Software Library contains a variety of on-line and off-line tools. The library includes:

- language processors
- text preparation programs
- program development and test tools
- file manipulation programs
- directory maintenance utilities
- system maintenance utilities
- hardware test and diagnostic programs

4.10.1 Introductory Comments

This section gives a brief description of the utilities at the disposal of the user. For each program, a short description is given of its function, the way it is called (parameters), and a reference to the manual describing the program.

A utility program is started by entering a command line to a terminal. The command line generally consists of the program name followed by a number of parameters which are program specific. These parameters are described for each program in the subsequent paragraphs.

When a program is started by the operating system it is given information describing its environment:

- The name of the current file system.

- A handle to the current directory
- A handle to a current input file, which is normally the terminal from which the program was activated.
- A handle to a current output file which is normally the terminal from which the program was activated.

The current input file and the current output file may, however, be associated with any file at load time.

This is done by entering

either

CIN: <file id>

or

COUT: <file id>

respectively (or both) in the program activation line as a supplement to the program specific parameters.

The syntax of program invocations is described by means of a BNF-syntax.

The following meta-symbols are used:

Repetition: {<something>} max / min

<something> must occur
minimum "min" times, and
maximum "max" times. If
omitted: min = max = 1.

Alternatives: { <either> / <or> } or

<either> | <or>

The first line of the invocation syntax gives the program name and a general parameter description. This is elaborated on in the sections which follow. The sections are formatted in three columns:

- A syntax description
- A description of defaults
- A description of use

In the default column the following symbols are used:

<:current input file
>:current output file
-:no default

In key words the significant letters are underlined.

The syntax of a file identification: <file id> may be given as:

<file id> :: = @{<file system name>}$_0^1$ * {<volume name>}$_0^1$
|<file name>
|<file id> * <file name>

<file system name> :: = <name> - <name>

<volume name> :: = <name>

<file name> :: = <name>

<name> :: = <letter> {<letter> | <digit>}$_0^\infty$

<letter> :: = A..Z | _ | <null>

<digit> :: = 0..9

The <file id> is the hierarchical identification of a file, starting either from the current directory (the first term being a <file name>), or from <file system name> identifying a particular file system followed by the name of volume under that file system.

Examples

@FILSYS - PRI1 * VOLØØ2 * DIREC1 * MYDIREC * MYFILE
@** DIREC1 * MYDIREC * MYFILE
MYFILE

4.10.2 Language Processors

The list of DAMOS language processors includes:

- Micro Assembler
- Assembler
- SWELL Compiler
- PASCAL Compiler
- COBOL Compiler
- PASCAL cross reference generator
- Assembler cross reference generator
- Parsing System
- PASCAL pretty printer
- SWELL pretty printer
- Linkage editor

4.10.2.1 Micro Assembler

Invocation syntax:

MICAS $\{\text{<user params>}\}_0^\infty$

			Default:	Use:
<user params> ::=	$\{\underline{S}:\}$ $\{\underline{I}:\}$	<file id>	<	(source text)
	|O:	<file id>	>	(PROM output)
	|B:	<file id>	-	(binary object)
	|P:	<file id>	>	(listing)

The program translates symbolic assembly language into a binary object and an object text file which can be used as input to a DATA I/O PROMMER for programming of PROM chips.

The target machine and the physical format of the PROM chips must be specified in the source text.

For further information: CSS/006/USM/0019

4.10.2.2 Assembler

Invocation syntax:

AS $\{\text{<user params>}\}_{0}^{\infty}$

			Default:	Use:
<user params> :: =	$\left\{\begin{matrix}\underline{S}\\ \underline{I}\end{matrix}\right\}$:	<file id>	<	(source text)
	O :	<file id>	-	(object code)
	P :	<file id>	>	(listing)
	N :	<file id>	-	(names output)

The program translates symbolic CR80 assembly language into a directly loadable object module.

All files must exist before program invocation. An object file id must be specified.

For further information: CSS/401/USM/0042.

4.10.2.3 SWELL Compiler

Invocation syntax:

SWELL <source> $\{$<user params>$\}_{0}^{\infty}$

			Default:	Use:
<source> ::=	I:	<file id>	-	(source text)
<user params> ::=	O:	<file id>	-	(object module)
|	P:	<file id>	-	(listing)
|	L:	<integer>	0	(list level)
|	X:	<yes/no>	no	(cross ref.)

The program compiles SWELL source texts into object modules, which may be linked together with other object modules to form a loadable module.

If not specified, object module and listing will be directed to non-retrievable temporary files. A source file must be specified.

For information on the SWELL language: CSS/415/RFM/0002

For further information on the SWELL compiler: CSS/415/USM/0047.

4.10.2.4 PASCAL Compiler

Invocation syntax:

PASCAL {<user parameters>}$_{0}^{\infty}$ {<control>}$_{0}^{3}$

		Default:	Use:
<user parameters> ::=	I: <file id>	<	(source text)
	S:		
	O: <file id>	PASCAL_OBJECT	
	P: <file id>	>	(listing)
<control> ::=	/L	-	(force list)
	/N	-	(force nolist)
	/T	-	(test output

The program compiles PASCAL source texts into object modules, directly loadable. If the source file contains $ <file id> at the beginning of a line, input will continue from the specified file, and return to the previous source file upon exhaustion.

If an object is specified, it must exist before program invocation.

For further information on the PASCAL language: CSS/006/RFM/0001

For further information on PASCAL compiler: CSS/006/RFM/0001.

4.10.2.5 COBOL Compiler

Invocation syntax:

COBOL $\{\text{<user params>}\}_{0}^{\infty}$

			Defaults:	Use:
<user params> ::=	I:	<file id1>		(source text)
|	O:	<file id2>	file id1 .0	(object code)
|	P:	<file id3>	file id3 .0	(list)
|	X:	{RW / AW}		(cross-reference listing)

The CR80 COBOL compiler is an industry - compatible, efficient two-pass compiler, which translates source programs into object code, and produces a list on request.

CR80 COBOL is based upon American National Standard X3.23-1974 level 1 with nearly all features of level 2.

For further information on the COBOL language: COS/710/RFM/0001
For further information on the COBOL compiler: COS/710/USM/0008
COS/710/PSP/0071

4.10.2.6 PASCAL Cross Reference Generator

Invocation syntax:

PCROSS <input file> <output file>

			Default:	Use:
<input file> ::=	I:	<file id>	<	(source text)
<output file> ::=	O:	<file id>	>	(cross reference)

The program produces a crossreference listing of a CR80 PASCAL source text.

Both files must exist before program invocation.

For further information: CSS/133/USM/0038.

4.10.2.7 Assembly Language Cross Reference Generator

Invocation syntax:

ACROSS <input file> <output file>

			Default:	Use:
<input file> ::=	I:	<file id>	<	(source text)
<output file> ::=	O:	<file id>	>	(cross reference)

The program produces a crossreference listing of a CR80 assembly language source text.

Both files must exist before program invocation.

For further information: CSS/134/USM/0039.

4.10.2.8 Linker

Invocation syntax:

LINK $\left\{\begin{array}{l}\text{<link parameter>}\\ \text{<program option>}\\ \text{<command switch>}\end{array}\right\}_{2}^{\infty}$

<link parameter> ::=

	Default:	Use:
MAIN: <file_id>	-	main module
I: <file_id>	-	main module
SUB: <file_id> $\{/\text{<file\_id>}\}_{0}^{\infty}$	-	sub modules
O: <file_id>	-	object file
P: <file_id>	>	print file
L: <integer>	0	list level
X: <boolean>	NO	cross ref. flag
TARGET: <integer> $\{/\text{<integer>}\}_{0}^{\infty}$	2/3	target computers

<program option> ::=		Default	Use
PRIORITY:	<integer>	1	
CAPABILITY:	<integer>	0	
EXECLEVEL:	<integer>	2	
FDS:	<integer>	4	file descriptor
IOCBS:	<integer>	3	io control blocks
STREAMS:	<integer>	2	
TLES:	<integer>	9	transfer list elements
MESSAGES:	<integer>	4	
WORKAREA:	<integer>	0	free data space
VERSION:	<integer>	0	
OVERLAY:	<integer>	0	min. prog space
DATA:	<boolean>	YES	process generation flag
PROGHEADER:	<boolean>	YES	program header generation flag
PROGFILL:	<letter>.<integer>	-	program fill specification
PROCFILL:	<letter>.<integer>	-	process fill specification
REENTRANT:	<boolean>	YES	
RESIDENT:	<boolean>	NO	
PERMANENT:	<boolean>	NO	
MONITOR:	<boolean>	NO	
UTILITY:	<boolean>	YES	
PROGNAME:	<id(6)>	-	
PROCNAME:	<id(6)>	-	
CPUNAME:	<id(6)>	-	
USERID0:	<integer>	-	
USERID1:	<integer>	-	

<command switch> ::=		Default	Use
COMMANDS:	<file_id>	-	command file

The program creates a directly loadable object module from one or more independently translated modules.

For further information: CSS/416/USM/0048.

4.10.2.9 Pretty Pascal

The program is activated by entering:

PRETTYPASCAL I: <file id> O: <file id>

<Input file>: a file of characters, presumably a Pascal program or program fragment.

<Output file>: the prettyprinted program. (Note that merge directives $ <file id> may have been indented, and therefore no longer accepted by the Pascal compiler, which must have the "$" in column one).

PRETTYPASCAL takes as input a Pascal program and reformats the program according to a standard set of prettyprinting rules. The prettyprinted program is given as output. The prettyprinting rules are given below.

An important feature is the provision for the use of extra spaces and extra blank lines. They may be freely inserted by the user in addition to the spaces and blank lines inserted by the prettyprinter. No attempt is made to detect or correct syntactic errors in the user's program. However, syntactic errors may result in erroneous prettyprinting.

General Prettyprinting Rules

1. Any spaces or blank lines beyond those generated by the prettyprinter are left alone. In addition, comments are left where they are found, unless they are shifted right by preceeding text on a line.

2. All statements and declarations begin on separate lines.

3. No line may be longer than 80 characters long including newline. Any line longer than this is continued on a separate line.

4. The keywords "BEGIN", "END", "REPEAT" and "RECORD" are forced to stand on lines by themselves (or possibly followed by supporting comments).
   In addition, the "UNTIL" clause of a "REPEAT-UNTIL" statement is forced to start on a new line.

5. A blank line is forced before the keywords "PROGRAM", "PROCEDURE", "FUNCTION", "CONST", "TYPE" and "VAR".

6. A space is forced before and after the symbols ":=" and "=". The latter only in declarations. Additionally, a space is forced after the symbol ":".

Indentation Rules

1. The bodies of "CONST", "TYPE", and "VAR" declarations are indented from their corresponding declaration header keywords.

2. The bodies of "BEGIN-END", "REPEAT-UNTIL", "FOR", "WHILE", "WITH" and "CASE" statements, as well as "RECORD-END" structures and "CASE" variants (to one level) are indented from their header keywords.

3. An "IF-THEN-ELSE" statement is indented as follows:

```
IF expression
  THEN
        statement
  ELSE
        statement
```

4.10.2.10 Pretty SWELL

The program is activated by entering:

PRETTYSWELL I: <file id> O: <file id>

<Input file>: a file of characters, presumably a SWELL program or program fragment

<Output file>: the prettyprinted program.

PRETTYSWELL takes as input a SWELL program and reformats the program according to a standard set of prettyprinting rules.
The prettyprinted program is given as output. The prettyprinting rules are given below.
An important feature is the provision for the use of extra spaces and extra blank lines. They may be freely inserted by the user in addition to the spaces and blank lines inserted by the prettyprinter.
No attempt is made to detect or correct synctactic errors in the user's program.
However, synctactic errors may result in erroneous prettyprinting.

General Prettyprinting Rules

1. Any spaces or blank lines beyond those generated by the prettyprinter are left alone.
   In addition, comments are left where they are found, unless they are shifted right by preceeding text on a line.

2. All statements and declarations begin on separate lines.

3. No line may be longer than 80 characters long including newline. Any line longer than this is continued on a separate line.

4. The keywords "BEGIN", "END", "REPEAT" and "RECORD" are forced to stand on lines by themselves (or possibly followed by supporting comments).
   In addition, the "UNTIL" clause of a "REPEAT-UNTIL" statement is forced to start on a new line.

5. A blank line is forced before the keywords "PROGRAM", "PROCEDURE", "INIT", "CONST", "TYPE", "LABEL" and "VAR".

6. A space is forced before and after the symbols "=> " and "=". The former only outside relational expression and parentheses, and the latter only in declarations. Additionally, a space is forced after the symbol ":".

Indentation Rules

1. The bodies of "CONST", "TYPE", "LABEL" and "VAR" declarations are indented from their corresponding declaration header keywords.

2. The bodies of "BEGIN-END", "REPEAT-UNTIL", "SWITCH", "WHILE", "WITH" and "CASE" statements, as well as "RECORD-END" structures and "CASE" variants (to one level) are indented from their header keywords.

3. An "IF-THEN-ELSE" statement is indented as follows:

```
IF expression
  THEN
        statement
  ELSE
        statement
```

4.10.2.11 Parsing System

The parsing system is a set of tools for table driven text scanning and syntax analysis.

The parsing system consists of three modules:

- a PARSE TABLE GENERATOR
- a PARSER to be used in a PASCAL milieu
- a PARSER to be used in a SWELL milieu

The PARSE TABLE GENERATOR reads a syntax description (Backus Naur Form), checks its validity and generates parse tables to be used to drive the PARSER.

PARSERGEN $\{$<user param>$\}_{3}^{6}$

<user param>::=		Default	Use:
	I: <file id>	-	(BNF input file)
	|O: <file id>	-	(object (table) file)
	|F: {ABS / REL}	-	(PASCAL parser format) (SWELL parser format)
	|P: <file id>	>	(print file)
	|L: {BNF / NO / ALL}	NO	(list syntax) (no listing) (list syntax and parse actions)
	|V: {YES / NO}	NO	(input verification)

The PARSER performs:

- scanning of input text (recognizing input symbols like constants, identifiers, etc.)
- syntax analysis
- activation of (user written) semantics procedure, when a syntactical unit is recognized (corresponding to a production of the input syntax to the PARSE TABLE GENERATOR).

For further information: CSS/210/USM/0051.

4.10.3 Text Processors

The list of DAMOS text processors includes:

- an Interactiv Editor
- a Batch Editor
- a Text Formatting Program
- a File Merge Program
- a File Concatenator
- a Makelines Program
- a Translate Program for converting between capital letters and minuscules

4.10.3.1 Interactive Editor

Invocation syntax:

EDIT

The editor is line-oriented; all commands concern one or more lines. The editor keeps track of a current line position.

Line numbers are denoted by means of

- a numeric value
- a text string, which is part of the line
- a symbol (. means current line; ≠ means last line)
- one of the above +/- a displacement

Below is given a short list of commands to the editor. The bracket contents indicate the number of line numbers applicable for each command, and their defaults. The brackets are not part of the command language. <q> means any quote character.

(.) A	Append text after line (text follows)
(.) C	Change line
(.,.)D <q><pattern><q>	Delete line(s)
(.) I	Insert text before line (text follows)

(.,.) K	Copy line(s) to after current
L(.) (commands)	
L	Loop
(.,.) M	Move line(s) to after current
(.,.) P	Display line(s)
Q	Quit
(.,.) R <file id>	Read file, appending after line
(.,.) S <q><pattern1><q><pattern2><q> [G][P]	Substitute pattern2 for first occurrence of pattern1 in each line. (G implies all occurrences), (P implies display of result)
(1, ≠/) W <file id>	Write line(s) into file
(.) X <q><pattern><q>[D]	Split line before pattern (D implies deletion of second line)

For further information: CSS/102/USM/0021.

4.10.3.2 Batch Editor

Invocation syntax:

CEDIT I:<file-id> O:<file-id> C:<file-id> V:<file-id>

Default for all files is the users terminal.

CEDIT is a sequential character editor, i.e. it is not possible to go backwards in the input file.

It is intended for batch use with a command file (C:), but can also be used interactively.

The verification file (V:) receives verifications of the editing, the verification type is specified by a bit mask.

The editor has facilities to deletion of, searching for, insertion of and substitution of text strings.

Commands may be given a repeat command, and several commands may be enclosed in parentheses to form a compound command.

CEDIT is especially suited for large overall substitutions in a file or when the line format of EDIT imposes too many restrictions.

The Editor Commands are summarized below:

	EDITOR COMMANDS
I/<string>/, I.<string>., IL, IP	INSERT the string or the character
/<string>/, .<string>., C, L, P, E	SEARCH the string or the character
D/<string>/, D.<string>., DC, DL, DP, DE	DELETE until the string or until the character
S/<string1>/<string2>/, S.<string1>.<string2>.	SUBSTITUTE string1 by string 2
B	BACK to before last character read
X	EXIT from program
F	FINISH by E, X
M(<value>)	MASK specification of verification
H	HOME positioning of the source file pointer
<times><simple command>	REPEATED COMMAND execution

For further information: CSS/137/USM/0059.

4.10.3.3 Text Formatting Program

Invocation syntax:

FORMAT $\{\text{<user params>}\}_2^2$

	Defaults:	Use:
<user params> ::= I : <file id>	-	(input file)
O : <file id>	-	(output file)

The program transforms an input file with embedded commands to an output file.

The formatting consists of:

- margin justification
- underlining key words
- addition of page headings
- table of contents generation

For further information: CSS/180/USM/0044

4.10.3.4 File Merge Program

Invocation syntax:

MERGE <user params>

	Default:	Use:
<user params> :: = I : <file id>	<	(input)
O : <file id>	>	(output)
D : <file id>	-	(secondary directory)

The program merges text files together onto an output file.

The input file may contain commands to the merge program. A command is a dollar sign ($, Å, ascii 36) followed by a file-id. The input file is copied onto the output file until a command (or end of file) is reached.

If a command is encountered, the command itself and the rest of that line is skipped and the specified file is merged to the output file. The specified file is, in turn, scanned for commands where new files can be merged. This process may continue to a maximum level of 10.

When the specified file is merged the copying is resumed, and continued until another command (or end of file) is reached.

The files are sought in current directory. If a secondary directory was specified, and a file could not be found, it is sought in the secondary directory.

For further information: CSS/142/USM/0024.

4.10.3.5 File Concatenator

Invocation syntax:

CONCAT F:<file-id> O:<file-id>

The files listed in the file after F: are concatenated to the O: file.

The F: file must contain one <file-id> on each line, possibly preceeded by a '$' character.

The execution of this program is very fast, since it uses direct, double-buffered I/O.

4.10.3.6 Makelines

Activation

MAKELINES I: <file id> O: <file id>

Both files must exist beforehand and consist of text only.

The program adds line numbers to a file in a manner equivalent to the EDITOR's.

4.10.3.7 Translate

Activation

To translate every occurrence of small letters to capital letters use:

MAKELARGE I: <file id> O: <file id>

To do the reverse:

MAKESMALL I:<file id> O:<file id>

Both files must exist beforehand and consist of text only.

The program converts small letters to capital letters or vice versa.

4.10.4 Test Tools

The list of DAMOS test tools includes:

- a Patch program
- a Disassembler
- an On-Line Test Output facility
- an Off-Line Log Editor
- an interactive symbolic Debugger

4.10.4.1 Patch Program

Invocation syntax:

PATCH <input file> {<output file>}

	Defaults:	Use:
<input file> ::= I : <file id>	-	(input)
<output file> ::= O : <file id>	<input file>	(output)

The program copies an input file to an output file. The user is then able to dump and patch the output file interactively.

Commands are read from current input, and messages are displayed on current output.

The interactive commands are:

D <first> <last>	Dump
P <first> <word>	Patch
C <first> <file id> <skip> <max>	Copy another file into the output file
S <first> <word>	Search for pattern
Q	Quit

<first>, <last> are hexadecimal address.
<word> is hexadecimal data words.
<skip>, <max> are hexadecimal counts.

For further information: CSS/155/USM/0030.

4.10.4.2 CR80 Disassembler

Invocation syntax:

DISASM $\{$<user params>$\}_{2}^{4}$

		Defaults:	Use:
<user params> ::=	I : <file id>	-	(input file)
	O : <file id>	-	(output file)
	F : <integer>	0	(start address)
	L : <integer>	32767	(stop address)

The program converts a binary input file to a symbolic assembly language representation on an output file.

Both files must be specified and must exist before program invocation.

4.10.4.3 On-Line Test Output Facility

The Test Output Facility is a tool to be used for trouble shooting and faults finding in a real-time environment. It allows a selective logging of user defined trace output on backing storage for subsequent off-line analysis.

On-Line Log Control Program

Invocation syntax:

CONTROLLOG

The program is used to allocate/deallocate log files and to control on a process basis the ability to generate test output to the current log file.

The program is intended for interactive use. It prints and writes messages on current output and accepts the following commands from current input:

CLOSE
EXCLUDE <process name> ,<process name>
INCLUDE <process name> ,<process name>
OPEN <log file id><no. of blocks>
QUIT
RESET <process index> ,<process index>
SET <bit mask> ,<bit mask>
STATUS

For further information: CSS/341/PSP/0015.

4.10.4.4 Off-Line Log Editor

Invocation syntax:

EDITLOG <number of records wanted>
<qualifier>
{<user labels>}$^{1}_{0}$
<log file>
<output file>

			Default:	Use:
<qualifier>:		refer to CSS/341/PSP/0015	TRUE	qualifies input
<user labels>	::=	USE : <file id>	-	label definitions
<log file>	::=	I <file id>	-	input file
<output file>	::=	O : <file id>	>	output file

The program EDITLOG is intended for editing and printing a log file generated by the Test Output Facility. By specifying a qualifier, it is possible to search for specific records in the log file. It is also possible to have application dependent labels on output by providing a user label file.

For further information, e.g. on the qualifier syntax:
CSS/341/PSP/0015.

4.10.4.5 Debugger

Invocation syntax:

DEBUG

The SWELL DEBUGGER is a tool for detecting logical errors in SWELL programs.

The debugger provides functions for the following:

- Insertion, deletion and inspection of breakpoints
- Listing of current values of variables referenced by symbolic name
- Assignment of new values to variables referenced by symbolic name
- Maintenance of a cyclic log of the last 16 procedure calls
- Process communication, to/from synchronization elements identified by symbolic names in the SWELL program
- Low level features, such as access to registers and dumping and changing locations referenced by address

The program to be debugged is loaded and executed by the debugger, which acts as a small operating system.

The necessary information for referencing the objects in the SWELL program by symbolic names is provided by the SWELL compiler and the LINKER.

4.10.5 File Manipulation Programs

The list of DAMOS file manipulation programs includes:

- a File Copy program
- a Directory Copy program
- a File Display program
- a File Compare program
- a Line Printer Support program
- File Conversion utility programs (Binary/Hexadecimal)

4.10.5.1 File Copy Program

Invocation syntax:

COPY <input file> <output file>

	Default:	Use:
<input file ::= I : <file id>	<	(input)
<output file> ::= O : <file id>	>	(output)

The program makes a data transparent copy of the input file into the output file.

Both files must exist before program invocation.

For further information: CSS/110/USM/0032.

4.10.5.2 Directory Copy Utility

Invocation syntax:

DCOPY I: <directory file id> 0: <directory file id> M:<mask>

The program DCOPY copies the contents of some or all non-directory files from one directory to another. Output files will be created if necessary. A mask can be used to select a subset of files for copying. The file names must then contain the mask which is a pattern of up to sixteen characters.

Messages are written on current output. For the selected files in the input directory a message is written so the exact amount of copying can be determined.

Success:

<filename> COPIED

Failure:

<filename> NOT COPIED: <self explanatory text>

4.10.5.3 File Display Program

Invocation syntax:

LISTF <input file>

	Default:	Use:
<input file> ::= <file id>	-	(input)

The program displays the contents of the file 23 lines (one page) at a time.

4.10.5.4 Compare

Activation:

COMPARE <file id> <file id>

Function:

The program compares the two specified files, word for word, and reports:

a) that the two files are identical, or

b) that one file is shorter than the other (meaning that they are identical over the size of the shorter file), or

c) that they differ at word number <word numb>, where <word numb> represents the first word that was different (the words in a file being numbered 0, 1, 2, ...) in decimal notation.

4.10.5.5 Line Printer Support Program

Invocation syntax:

PRINT { <file> { C: <no> } 1 } ∞
                { 0 }   0

	Default:	Use:
<file> ::= <file id>	-	file to be printed
<no> ::= <integer>	1	number of copies

The program enters 0, 1 or more files into the print queue for subsequent printing on the line printer. The program checks that the user has the right to look-up and read the file.

For further information: CSS/006/USM/0033.

4.10.5.6 File Conversion Utility Programs

The file conversion utility programs are used to change the format and/or representation of a file.

4.10.5.6.1 Binary to Hexadecimal Conversion Program

Invocation syntax:

BINHEX <input file> <output file>

	Default:
<input file> ::= I : <file id>	>
<output file> ::= O : <file id>	<

The program BINHEX converts a binary input file into a hexadecimal output file. From the input file, 16-bit words are read and converted to four-digit hexadecimal numbers and written onto the output file. The conversion continues until the input file is exhausted.

For further information: CSS/114/USM/0027.

4.10.5.6.2 Hexadecimal to Binary Conversion Program

Invocation syntax:

HEXBIN <input file> <output file>

	Default:
<input file> ::= I : <file id>	>
<output file>::= O : <file id>	<

The program HEXBIN converts a hexadecimal input file into a binary object file.

For further information: CSS/113/USM/0028.

4.10.6 Directory Maintenance Utilities

The DAMOS directory maintenance utility programs are a suite of programs for inspecting and modifying the contents of backing store file structure. Programs for the following directory operations are available:

- create a file
- delete a file
- rename a file
- enter an existing file into a directory
- list the contents of a directory
- list the attributes of a file
- change the protection of a file.

4.10.6.1 File Creation Program

Invocation syntax:

$$\text{CREATE}\left\{\text{<direc id>}*\right\}_{0}^{1}\ \text{<file name>}\ \left\{\begin{matrix}\underline{\text{D}}\text{IRECTORY}\\ \underline{\text{C}}\text{ONTIGUOUS}\\ \underline{\text{R}}\text{ANDOM}\end{matrix}\right\}_{1}^{1}\ /\text{<integer>}$$

Creates a new file on the volume specified in <direct id>, and enters the file with the name <file name> in the directory identified by <direc id>. The directory must be an existing file. If the directory is not specified current directory is used.

The organization of the file is stated in the third parameter. If the organization of the file is DIRECTORY or RANDOM, integer specifies the area size of the file in sectors, else <integer> specifies the number of sectors to be allocated.

For further information: CSS/932/USM/0036

4.10.6.2 File Deletion Program

Invocation syntax:

$$\text{REMOVE}\left\{\text{<direc id>}*\right\}_{1}^{1}\ \text{<file name>}$$

Clears the entry with the name file name in the directory specified. If the directory is not specified current directory is used.

For further information: CSS/932/USM/0036

4.10.6.3 File Renaming Program

Invocation syntax:

RENAME $\{$<direc id>*$\}_0^1$ < old file name> / <new file name>

Clears the entry with the old file name in the directory specified. An entry with the new file name is inserted instead. If the directory is not specified current directory is used.

For further information: CSS/932/USM/0036

4.10.6.4 File Include Program

Invocation syntax:

ENTER <file id> $\{$ <direc id>*$\}_0^1$ <file name>

Creates a new entry in a directory, which maps the file name into a reference to the file corresponding to <file id>.
If a directory is not specified current directory is used.
The directory must be an existing file.

For further information: CSS/932/USM/0036

4.10.6.5 Directory List Program

Invocation syntax:

LIST {<direc id>}$_{0}^{1}$

The program lists the BFD-entries for all files entered in the directory stated as parameter. For each entry is listed:

- name
- type
- size in bytes
- number of allocated areas
- areasize

If a directory is not specified the current directory is used as default.

For further information: CSS/932/USM/0036

4.10.6.6 List Directory

Function

The program is activated by entering:

LISTDIR {D: <directory>} {L: <maxlevel>} { P: <file>}

D: <directory>

Specifies the directory to be listed.
DEFAULT is current directory.

L: <maxlevel>

Specifies the maximum level of directories to be listed.
<maxlevel> must be an integer in the range 1..10.
DEFAULT is 1.

P: <file>

Specifies a printfile.
DEFAULT is current output file.

The program makes a list of all the entries in a specified directory. If one of the entries is a directory itself, this directory is also listed.
This process continues until the user specified level is reached.

4.10.6.7 File Attribute Display Program

Invocation syntax:

ATTR {<direc id>*}$_{0}^{1}$ <file name>

The program lists the BFD-entry of the file specified as parameter. In addition to the attributes obtained using the LIST utility program this program will also provide information on the users having access to the file and their access rights.

For further information: CSS/932/USM/0036

4.10.6.8 Get File Information

The program is activated by entering:

GETINF <file id>

The program outputs information about a specified file. The format of the information can be seen from the example below. The meaning of the information is described in (1).

4.10.6.9 File Protect Program

Invocation syntax:

PROTECT <file id> <user> <access type>{/<access type>}$_0^\infty$

<user> ::= <number> | PUBLIC | SYSTEM

<access type>::= READBYTES | MODIFYBYTES
| APPENDBYTES | ENTER
| LOOK UP | RENAME
| REMOVE | RESET
| PROTECT | GETFILEINFORMATION
| OFFER | <number>

The specified user is allowed to access the file as defined by the access types stated. (If a number is specified a set of access types is defined, namely the access types corresponding to the number interpreted as a bit pattern). The access rights of the calling user must include the right to protect the file.

For further information: CSS/932/USM/0036

4.10.7 System Maintenance Utilities

The list of DAMOS system maintenance utilities includes:

- a Disk Initialization program
- a Disk Salvation program
- a Print Queue Rectification program

4.10.7.1 Disk Initialization Program

Invocation syntax:

DISKINIT { <file system> }$^{1}_{0}$

<device>
<volume>
<sectors>
{ <asize> }$^{1}_{0}$

{ <isize> }$^{1}_{0}$

{ <fmode> }$^{1}_{0}$

{ <dmode> }$^{1}_{0}$

		Default
<filesystem>	::= FSN : <filesystem>	Current file system
<device>	::= DEV : <device name>	-
<volume>	::= VOL : <volume name>	-
<sectors>	::= SECTORS : <sector count>	-
<asize>	::= BFDASIZE: <area size>	1
<isize>	::= BFDISIZE: <initial no. of entries>	3
<fmode>	::= FORMAT: (NO/YES)	YES
<dmode>	::= DETAILS: (NO/YES/ALL)	NO

The functions performed by the disk initialization program make a disk suitable for storage of information under the regime of the File Management System. The program will format the specified volume according to parameters and handle bad sectors. An initial "empty" file structure will be created.

For further information: CSS/930/USM/0034.

4.10.7.2 Disk Salvation Program

Invocation syntax:

SALV <file sys> <device> <volume>

		Defaults:
<file sys>::=	FSN :<file system name>	current file system
<device> ::=	DEV :<device name>	-
<volume> ::=	WORKVOL : <volume name>	current volume

For a file structured volume the disk salvation program is able to check the readability and validity and to rebuild the external file system data structures. The program also provides the ability to list all entries in the basic file directory and other directories on the volume.

When started SALV reads commands from current input and writes messages on current output. The commands are:

CHANGESECTOR	<sectornbr.><rel.word><new contents>
CHECKBM	
CLEANBM	
CHECKFILES	
CLEANFILES	
CHECKGARBAGE	
CLEANGARBAGE	
CHECKHB	
CLEANHB	
CHECKNAMES	
CLEANNAMES	
LISTFILES	
LISTNAMES	
LISTSECTOR	<sector nbr.>
MARK	<from sector nbr.><to sector nbr.>
UPDATE	
STOP	

For further information: CSS/931/USM/0035

4.10.7.3 Print Queue Rectification Program

Invocation syntax:

CHECKPQ

This program is used to check, rectify or initiliaze the print queue.

If the print queue file does not exist, the program creates and initializes one, and enters it into the main directory under the name: PRINT-QUEUE-FILE. This function of the program is used to initialize the print queue before any other operations on the print queue are performed.

If the print queue file does exist, the program checks the consistency of the file. If the file is inconsistent, the program tries to reconstruct it. This function of the program is used when the print queue for some reason, for instance a system crash, has become inconsistent.

For further information: CSS/006/USM/0033.

4.10.8 Diagnostic Programs

The Maintenance and Diagnostic (M&D) package is a collection of standard test programs which is used to verify proper operation of the CR80 system and to detect and isolate faults to replaceable modules.

The off-line M&D software package contains the following programs.

- CPU Test Program
- CPU CACHE Test Program
- Memory Map Test Program
- RAM Test Program
- Supra Bus I/F Test Program
- LTU Test Program
- Disk System Test Program
- Magtape System Test Program
- Floppy Disk Test Program
- TDX-HOST I/F Test Program

4.10.8.1 CPU Test Program

The Central Processing Unit Test Program tests proper operation of a CPU with a standard CR80 instruction set. The control logic of the CPU is tested by the special test instruction and by verification of all possible instructions in the instruction set, except for a few privileged instructions. The different addressing schemes are tested for each instruction.

Execution in user and system mode is tried to test mode switch and priviliged instructions. Memory protect and page fault are tested if a memory MAP module is available. The single step facility is tested.
All registers, data busses, arithmetic circuits, and data transfer ways are tested by the program.

4.10.8.2 CPU CACHE Test Program

The Central Processing Unit is tested as a CPU without CACHE. The CACHE is tested by special test instructions (microprogrammed built in test routine), by checking the CACHE status registers, and by automatic time measurement during execution of program parts with high and low hitrate.

4.10.8.3 Memory MAP Test Program

The Memory MAP Test Program verifies proper operation of the memory MAP-and the MAP Interface Adaptor- (MIA) module.
The control logic of the MAP is tested by activation of the Built In Test (BIT) routine, and by verification of the specified function.
The test contains the following subtests:

Memory MAP subtest

The MAP function is tested by change of the segment registers and segment tables such that all 64-segment tables are tested. Memory absent and protect are also tested in this subtest.

DMA subtest

Memory to memory DMA transfers are checked in this subtest.

Interrupt subtest

Write and read in the RAM memory associated with interrupt handling are tested (interrupt vectors, CPU pool, and CPU priority). Interrupt on different levels (from INTRA MEMORY DMA) are tested.

V24 Communication Port

The communication port is tested by output and input of a special test pattern. This subtest requires manual input and visual inspection by the operator at the console.

4.10.8.4 RAM Test Program

The Random Access Memory Test Program verifies proper operation of RAM memory modules.

The RAM test program is capable of testing all RAM modules in a CR80 System, both in the PU's and on the Data channel.

The following elements are tested: Bus interface, RAM internal addressing circuitry and storage circuitry. The test contains the following subtests:

a. Checker board pattern test.

   A checkboard pattern (hex 5555, AAAA, 5555, AAAA, ...) is generated and stored in the RAM module under test. Verification of the stored pattern is performed.

b. The possibility of changing single bits independently is tested by setting and clearing each bit in any storage location while keeping the remaining bits in the location cleared, i.e. the hexadecimal patterns 0001, 0002, 0004, ..., 8000 are stored and verified in all locations, one location at a time.

c. The internal RAM addressing circuitry (word address mode) is tested by storing the internal address of each RAM location in the location and subsequently verifying the simultaneous presence of all relevant address combinations. I.e. for a 64K word RAM, the following pattern is generated, stored and verified.
   hex 0000, 0001, 0002, 0003, ..., FFFE, FFFF.

d. The byte address mode is tested by storing and reading bytewise in all locations of the RAM module under test.

e. Long term stability (refresh) is tested by storing a pattern and verifying the pattern after 1 sec.

f. Move multiple. The lower half of the RAM section under test is filled with a checker-board pattern and then transferred to the remaining portion of the RAM by using the MOVM machine instruction.

4.10.8.5 Supra Bus I/F Test Program

The Supra bus test program verifies proper operation of a supra bus I/F module.
The program tests all of the modules set up and control commands whereby also the PU P-Bus and supra bus I/F interconnection lines are tested. A built in test is used to verify proper transmitting and receiving operation on the supra bus cable. The data transfer on the PU C-bus and memory addressing are thereby also tested.
The test program can be used for systems with or without the memory MAP module.

4.10.8.6 LTU Test Program

The LTU test program verifies proper operation of the Line Terminating Unit, LTU CR8066.
The LTU test includes test of all command instructions, interrupt control, DMA control, RAM's, PROM, the serial interface part, and the Z80-processor.

The test of the Z80-processor parts of the LTU module is performed by transmitting a Z80-program to the shared RAM. The Z80-processor is commanded to execute this program as a self test. The Z80-program stores a test result code in the shared RAM. Test result code is interpreted by the LTU test program in the CR80 processor and a test response is produced.

4.10.8.7 Disk System Test Program

The Disk System Test Program verifies proper operation of the Disk System: Disk Controller and RAM, Disk Controller Adaptor, and Disk Drive.
The test is based on prerecorded data and special "work track" on a reserved disk area. The complete Disk System test requires mounting of a special test disk package and manipulation of the write protect switch on the disk drive by the operator.
The program includes facility for generating and writing of test pattern for later use in tests.

The following elements are tested: Data bus interface, Disk addressing, RAM addressing, Cyclic Redundancy Code (CRC) generator and checker, interrupt circuit, unit selection, write protect, bad marking and all specified commands and status codes.

The test contains the following subtest:

Verification of test pattern directory

The test pattern directory is inspected and checked to ensure that the test patterns which are necessary for the following subtests are available.

Data pattern read

A set of different data patterns are read from the disk and checked to ensure proper function of disk and RAM addressing, data transfers, CRC code checker, status word and interrupt. Seek time and data transfer rate are controlled in this subtest.

Data pattern write and format

A set of different data patterns are written and checked. Track and sector format are checked. This subtest uses tracks reserved specially for this purpose to protect other data patterns.

The RAM memory on the controller board is part of the CR80 PE memory and may be tested separately by the RAM test program

4.10.8.8 Magtape System Test Program

The Magtape System Test Program verifies proper operation of the Magtape System: Magtape Controller, RAM, Magtape Adaptor, and Tape Transports.

The Magtape System test requires mounting of a special test tape and manipulating of the switches on the tape transport.
The program includes a facility for generating and writing of test patterns for later use in tests. The following elements are tested: Data bus interface, RAM addressing, parity bit generator and checker, interrupt circuit, unit selection, all specified commands and status codes.

The test contains the following subtest:

Verification of test pattern directory

The test pattern directory is inspected and checked to ensure that the test patterns which are necessary for the following subtests are available.

Data pattern read

A set of different data patterns are read from the tape and checked to ensure proper function of tape and RAM addressing, data transfer, parity checker, status word and interrupt. Transport speed and data transfer rate are controlled in this subtest.

Data pattern write and format

A set of different data patterns are written and checked. This subtest uses a reserved part of the tape to protect other data patterns.

The RAM memory on the controller board is part of the CR80 PE memory and may be tested separately by the RAM test program.

4.10.8.9 Floppy Disk Test Program

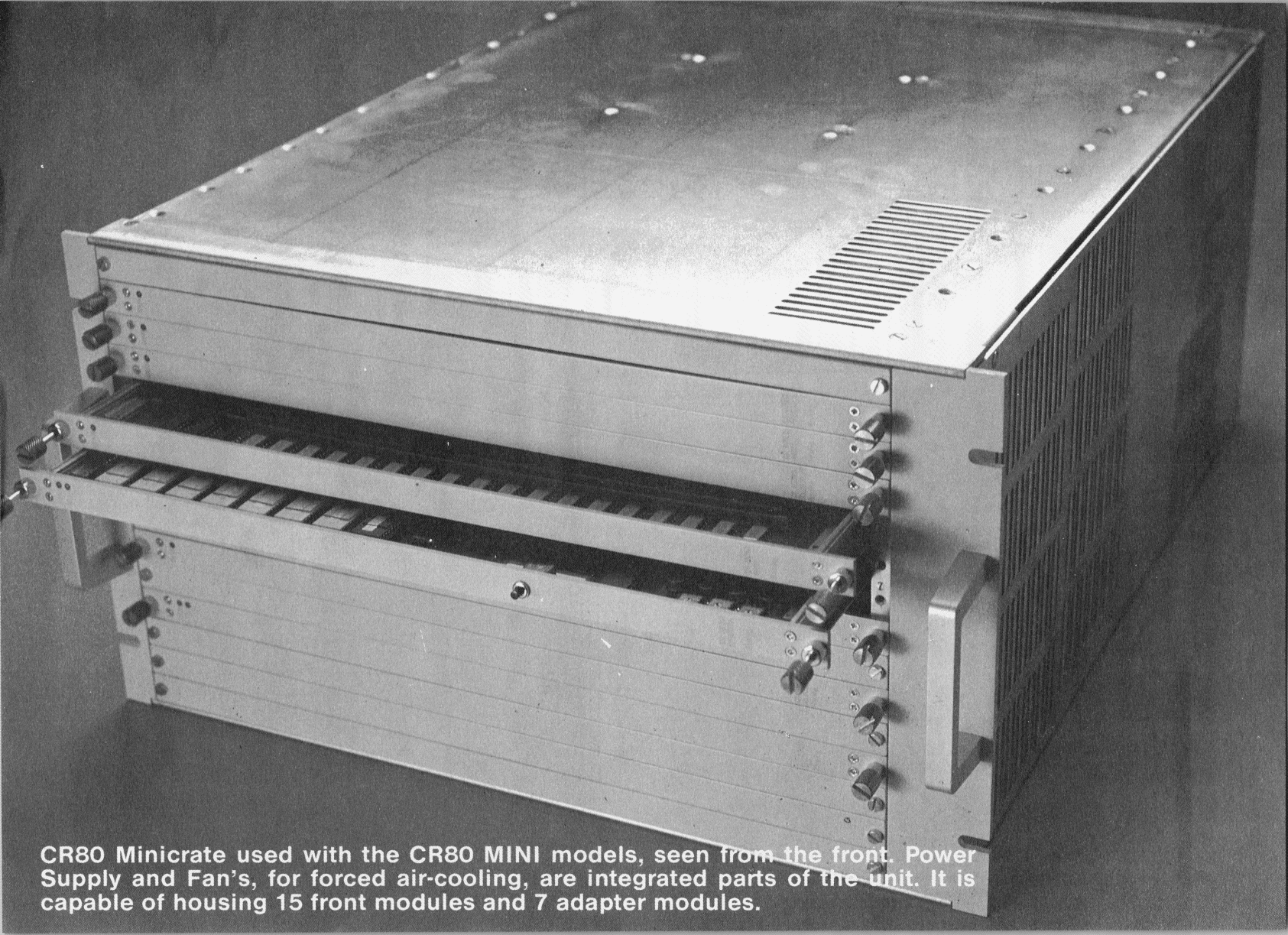
The Floppy Disk Test Program verifies proper operation of the Floppy Disk System: Floppy Disk Controller, Floppy Disk Adaptor, and Floppy Disk Drive.

The test is based on prerecorded data and special "work track" on a test diskette. The program includes facilities for generating and writing test diskettes.
The following elements are tested: Data bus interface, Diskette addressing, RAM buffer addressing, Cyclic Redundancy Code (CRC) generator and checker, interrupt circuit, unit selection, bad marking and all specified commands and status codes.

4.10.8.10 TDX-HOST I/F Test Program

The TDX-HOST I/F Test Program verifies proper operation of a TDX-HOST I/F (STI) module. The program performs the test by establishing a communication channel loop defined by the TDX-HOST I/F, the TDX-Cable and the TDX-controller.
The TDX-HOST module will in this way operate as both a transmitting and a receiving module. Data frames (TDX frames) are sent through the established communication channel and test responses are produced as results of comparisons of sent and received data frames.

NOTES:



CR80 Minicrate used with the CR80 MINI models, seen from the front. Power Supply and Fan's, for forced air-cooling, are integrated parts of the unit. It is capable of housing 15 front modules and 7 adapter modules.

Extension system for the CR80 MINI and TWIN range of computers. Shown is the
EPM front module and associated EPA adapter module which is inserted into the
Processor Unit. It connects, via twisted pair flat cable, to the also shown EI
module inserted into the extension CU-Crate.

5. CR80 MINI and TWIN, Non-Memory-Mapped Computers

5.1 Introduction

The CR80 MINI and TWIN non-memory-mapped computers are of modular construction similar to the CR80 MAXIM and FATOM memory-mapped computers. The major difference between the two types of systems, is that the MAP module and the data channel are not available with the unmapped systems. The system software supporting the unmapped CR80 computers is AMOS, Advanced Multiprocessor Operating System, which is described in the following chapter of the handbook. The hardware features supporting these systems are described in the following sections.

The modular construction allows for a wide range of computer configurations. These can vary from a single CPU, 64K-word RAM (16 bit) with a few peripherals to a configuration with dualized processor units each with multiple CPU's, 256K-word RAM and more than 400 I/O controllers.

5.2 CR80 MINI and TWIN Unmapped Computer Architecture

The CR80 MINI and TWIN computer systems, single and dualized systems, can be configured with a processing power ranging from 0.6 to 1.3 mega instructions per second (MIPS), memory sizes from 64K to 256K word (128 to 512 Kbyte), and can support up to 400 I/O controllers.

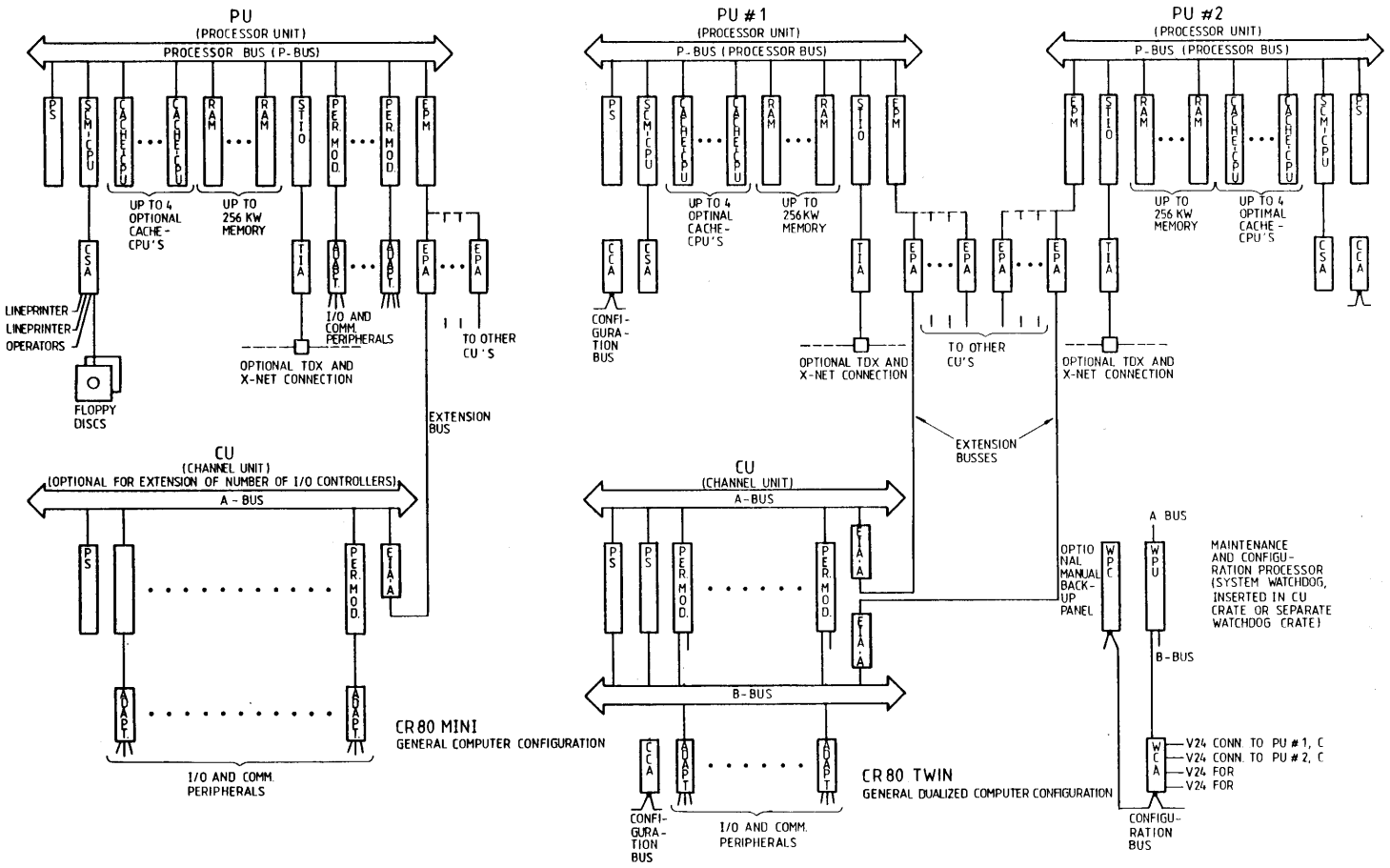
In order to direct the discussion, block diagrams of general CR80 MINI and TWIN computer configurations are shown overleaf (fig. 5.2-1). The basic elements in the systems are readily identified; namely, Processor Unit (PU), Channel Unit (CU), TDX and X-Net Bus, Extension Bus and Configuration Bus.

The TDX and X-Net Bus, Maintenance and Configuration Processor and its related Configuration Bus are identical for CR80 MINI and TWIN unmapped computers and mapped CR80 MAXIM and FATOM computers. These are described separately elsewhere in this handbook and are described in this chapter only where relevant to unmapped systems.

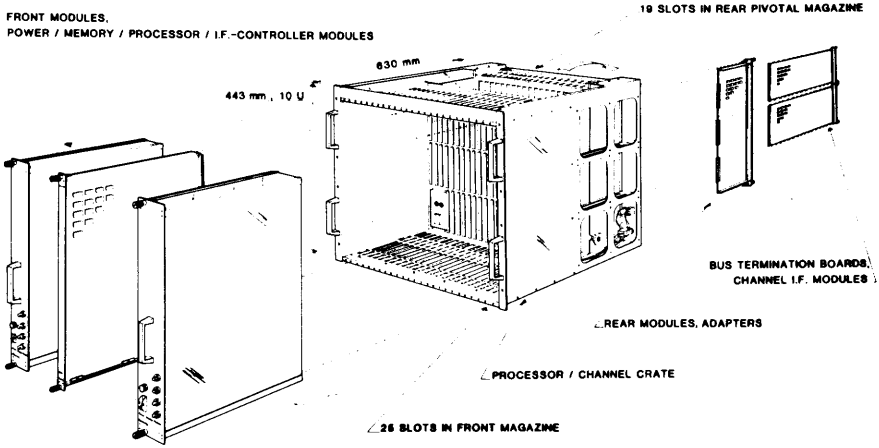
The Processor Unit (PU) is a selfcontained mechanical unit housed either in the standard CR80 M-crate with 25 module slots, or in the CR80 minicrate with 15 module slots, both shown in figure 5.2-2. The PU contains the system's processing power (CPU's), preferably also the working storage (RAM) and, if used, the TDX Bus Interface module (STI). If the system specifications require it, the PU can also be used for Peripheral modules since all internal transfer buses in the CR80 MINI and TWIN computers are electrically and functionally identical. In figure 5.2-3, the physical lay-out of a typical PU is shown. As seen in the figure only one transfer bus is available, the P-bus (processor bus), for interfacing the modules in the unit. This P-bus, described in detail later in this chapter, is a single multilayer printed circuit board; some of its characteristics are given in the next paragraph.

Figure 5.2-1

CR80 MINI and TWIN Unmapped Computers



CR80M PROCESSOR UNIT & CHANNEL UNIT



CR80M MINICRATE

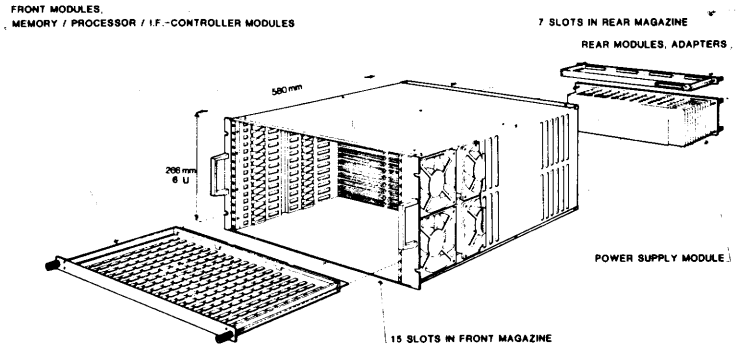
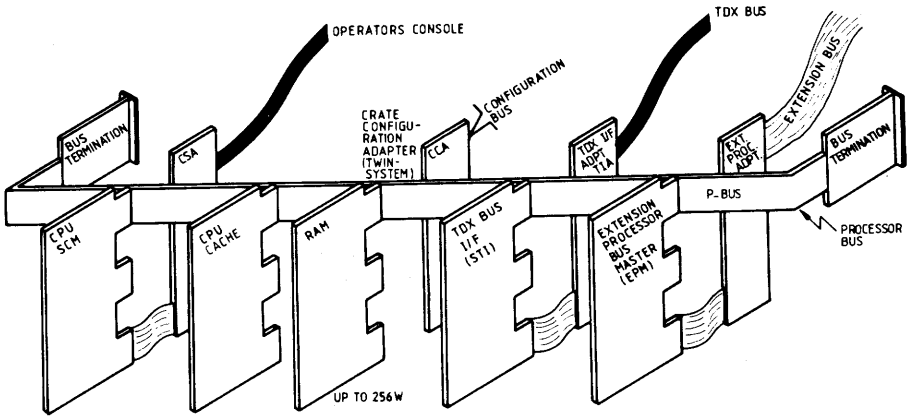


Figure 5.2-2
CR80, PU or CU Crate (25 position) and
CR80 PU-Minicrate (15 position)

Figure 5.2-3
Typical Physical Lay-out Processor Unit (PU)



NOTE: POWER SUPPLY NOT SHOWN

The P-bus contains a parallel address bus with a memory addressing range up to 256K word; and a parallel data bus of 16 bits plus 2 parity bits. The transfer rate supported by the bus timing is 4 megawords per second. Since the data transfer is asynchronous, the actual transfer rate is determined by the physical length of the P-bus and the types of modules connected to the bus. I/O Interrupts (8 bits) are transmitted from Peripheral modules in serial form on the bus and received by the System Control (SCM) part of the SCM-CPU. DC power distribution to the different modules in the PU is also accomplished via the P-bus printer circuit board.

The EXTENSION BUS used for interfacing between a PU and one or more CU's is implemented by differential transmission on a 30 twisted-pair flat cable. The EXTENSION BUS allows for greater physical distance between the modules (PU-CU) and its design expands the maximum number of addressable Peripheral modules.

The Channel Unit (CU), like the PU, is a selfcontained mechanical unit housed in a standard CR80 25 position Crate (figure 5.2-2). The Channel Unit used in dualized CR80 TWIN systems is identical to the one used in the CR80 FATOM mapped systems except that it is interfaced to the EXTENSION BUS (via EIA-A and B modules) instead of the DATA CHANNEL.

A typical physical module lay-out for a CU in the unmapped CR80 MINI or TWIN computers is shown in figure 5.2-4. From the figure, it is seen that the CU in the CR80 TWIN computers has a dual transfer bus structure: Data Bus A and Data Bus B. Each bus is connected by means of extension bus interface modules EIA-A & EIA-B to the TWIN PU's. The electrical specifications for the data buses are identical to those for the P-bus except that the supply voltages are 0.5V higher than nominal. This allows for the voltage drop in the power-ORing and limiting circuitry of the modules, which ensures that the failure of one power supply will not stop operation, and prevents a short in one module from shorting the power supplies in the CU. The functional and operational characteristics for the A and B data buses are as described above for the P-bus.

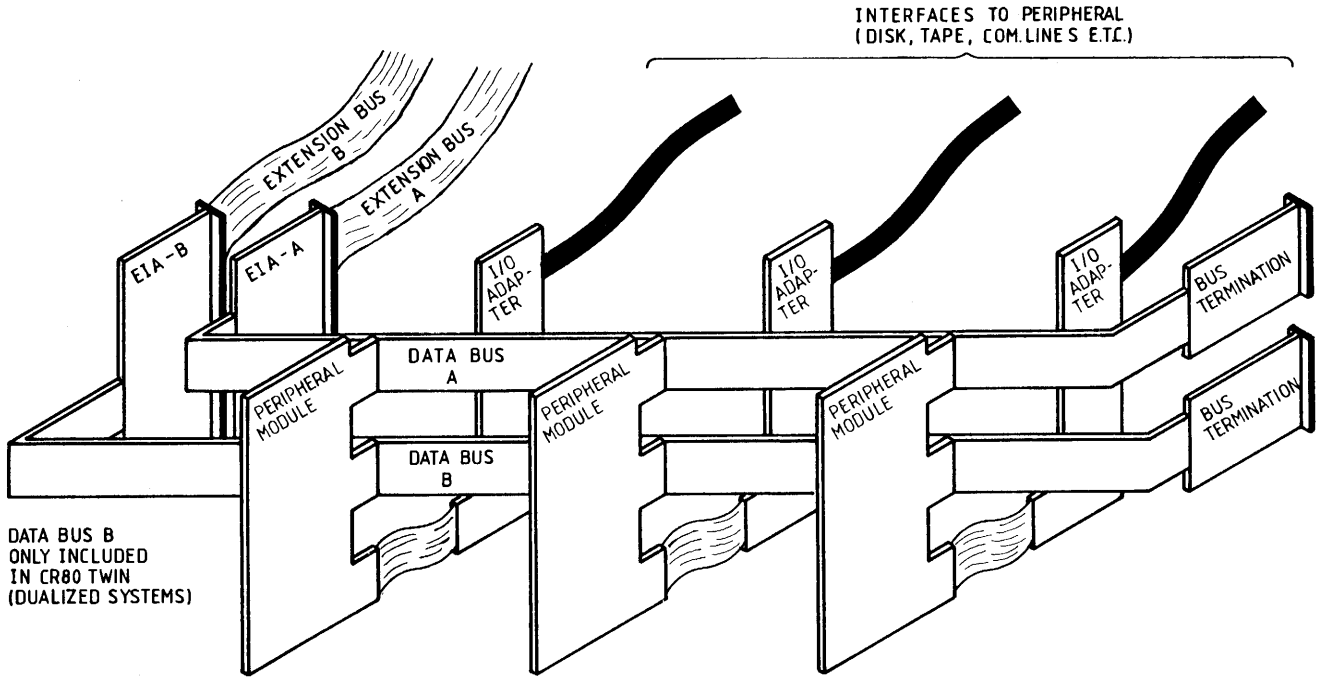


Figure 5.2-4
Typical Channel Unit
Physical Lay-out

TDX and X-Net devices, used for interfacing terminals, communication lines, process control, etc. are identical to those used in the CR80 MAXIM and FATOM mapped computers except for the TDX-CR80 interface module (STI0). This TDX-module incorporates different firmware for direct memory access (DMA) operation. Detailed descriptions of the TDX and X-Net subsystem, are found in chapter 8 (STI module in chapter 2).

5.3 Central Processor Units (CPUs)

Two different types of CPU's, the CPU-SCM and CACHE-CPU, are used with the unmapped CR80 MINI and TWIN multiprocessor PU's. One basic CPU-SCM module and additional CACHE-CPU modules as required, up to a total maximum of 5 CPU's can be accommodated in a multiprocessor PU (practically processing power enhancement is only achieved with up to 4 CPU's).

5.3.1 CPU-SCM Module:

The System Control Module (SCM) part of CPU-SCM performs the following functions in the PU:

- Generates master timing signals for the processor bus: 1 and 8MHz.
- Receives and queues I/O interrupts to be serviced by the CPU's. Queue status is signalled to the CPU's by special wires external to the transfer bus.
- Generates two special timer interrupts:
  160 us signalled to the CPU's on a special line (fast timer)
  10 ms handled as an I/O interrupt (real time).
- Performs bus arbitration to allow for up to five CPU and/or DMA modules to be connected to the same processor bus. The signals used for the arbitration are not part of the P-bus signals, but are carried on special wires in the CPU's I/O connector areas.
- Provides for the boot strap loader through a memory module area (PROM)
- Interfaces through a serial (V24) and two parallel ports a system console and two line printers.

The CPU part of the CPU-SCM module is identical to that of the CACHE-CPU module; both modules are described later in this chapter.

5.3.2 CACHE-CPU Module

The CACHE CPU is a general purpose CPU identical to the CPU part of the CPU-SCM but having the SCM part replaced by a CACHE memory and a control unit. The CACHE memory is a 1K x 38 bit RAM. The control unit performs read and write operations in CACHE memory as required. The CACHE memory uses a write-through algorithm, and relies on the basic principle that, in the step by step execution of a program, the probability of rereading a memory address more than once is high.

Two advantages are obtained by using CPU's with CACHE memory:

- Loading of the processor bus is reduced, allowing for the use of more CPU's before bus contention occurs. Up to five CPU's, one SCM CPU and four CACHE CPU's, can be installed in a PU.
  See fig. 5.3.2-1 below.

- Average access time when performing memory read operation is reduced, giving increased processing speed.

The CACHE memory is totally transparent to the programmer; that is, no software is required to support the CACHE operation. For further information on the CACHE memory is referred to Chapter 2.

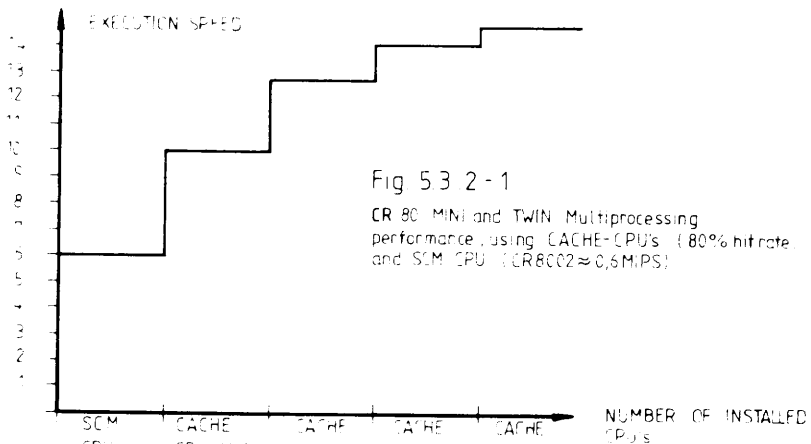


Fig. 5.3.2-1

CR 80 MINI and TWIN Multiprocessing performance, using CACHE-CPU's (80% hit rate and SCM CPU (CR8002 ≈ 0,6 MIPS)

5.3.3 CPU Part of the SCM-CPU and CACHE-CPU Modules:

The CPU is implemented with a standard instruction set of 16-bit instructions, including process switching, semaphore facilities for safe multi-processor systems programming, and an instruction modification facility providing dynamic change of instruction addressing schemes.

Separate base registers for programs and data areas (processes) are provided facilitate dynamic relocation of programs and data areas independently.

The CPU is micro programmed and has a microcycle time of 250 nsec. The interrupt handling scheme allows several CPUs in a multi-processor system to compete for interrupts. Each CPU only reacts to interrupts, that have a priority which is higher than or equal to the priority of the process being executed.

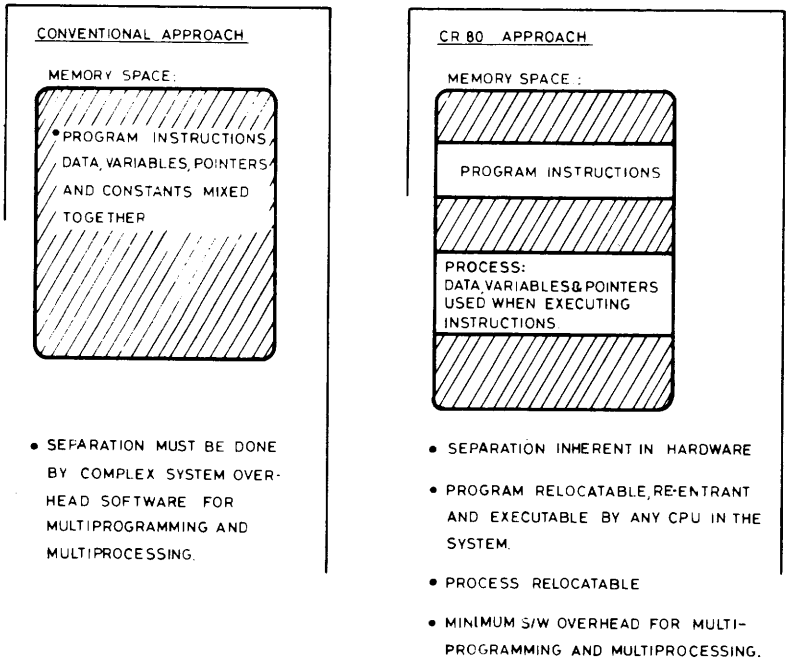
The CPU addresses up to 256K 16 bit words of memory using a paging technique. Four pages of 64K each are handled. The connection between page number and physical memory address is:

Page	Memory
0	0- 65,535
1	65,536-131,071
2	131,072-196,607
3	196,608-262,143

5.3.3.1 PROGRAM and PROCESS

The key to the safe reliable multiprogramming and multiprocessing of the CR80 MINI and TWIN unmapped computers is the separation in both hardware and software of the computer operations into a program part and a process part refer to the figure below.

The program part is the list of machine instructions to be performed, and the process part (separated totally from the program part in memory) includes the data, variables, and pointers needed by the program part when performing the total algorithm.



The process part, beyond containing all variables used by the program at execution time, also at its lowest (relative) memory addresses has a complete image of the CPU registers including the program counter. This area is called the Process Descriptor. When executing a program, the registers are loaded into the CPU from this process descriptor before execution takes place.

When a process is interrupted, the CPU registers are again saved into the process descriptor by hardware. A process can thus be started and stopped at any point in execution and time as all the CPU registers are loaded and saved each time. This allows for easy timesharing of CPU time between many processes. Each process contains all data relevant to the execution of a program. Thus, many processes may describe executions of the same, re-entrant, program.

As the program counter, pointing to the next instruction in a program to be executed, is one of the registers loaded and saved into the process area, only the start address of the process area (called the BASE address) has to be known to resume execution of a program.

This allows for many processes to execute the same program in a timeshared mode.

The processes may, independently of each other, execute different parts of the same program. If the system contains more than one CPU, processes can execute the same program in multiprocessing mode at different points at the same time.

For clarity, the definitions of program and process are introduced:

> A program is a collection of instructions and possibly parameters, assembled to perform a certain task. The program is addressed relatively to the program base register. The standard instruction set includes a number of instructions which work on program data. However, all these instructions can only read program data, not write into or modify the program data, thereby ensuring re-entrant code.
>
> A process is a data area containing description of a specific execution of a program. A process contains, in its first part, a process descriptor of

which the first 14 words are register contents that are loaded/saved by hardware, when processes are switched in and out of the CPU.

Memory allocation of programs and processes is strightforward.

A program can be placed anywhere in page zero* of memory (i.e. in locations 0-65535) without changing the program code itself. The only data to be changed if a program is moved to new locations are the PROG and PRPC register save locations in a process that executes the program.

PROG points to the origin of the program.
PRPC points to the instruction next to the one being executed.

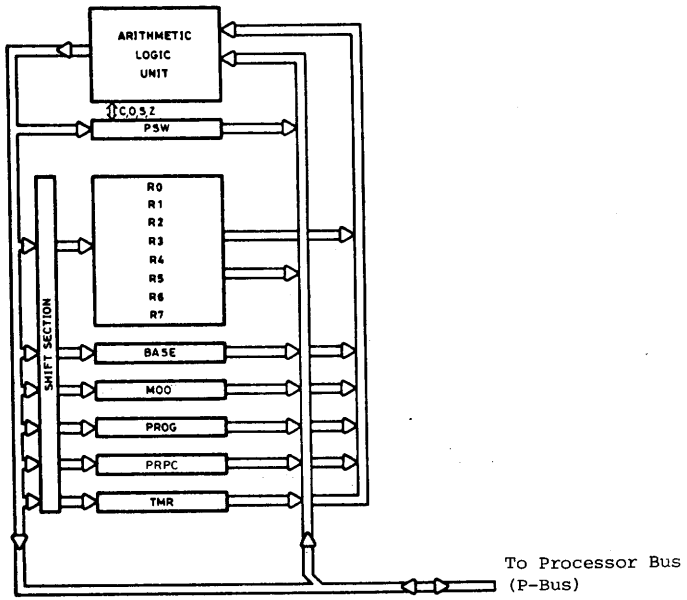
A process can be placed in any of the 4 memory pages. The BASE register points to the start of the process within that page.

* This is valid with the AMOS Operating System, with the extended AMOS (XAMOS) Operating System program can reside in all 4 pages.

5.3.3.2 THE CPU REGISTERS

The CPU contains 8 general purpose registers and six special registers:

The figure below shows the Accessible Registers, ALU and shift control of the CPU, when using the standard instruction set.



General Purpose Registers: R0 - R7

Special purpose Registers:

- PSW: Process Status Word.
- BASE: Process Start Location.
- MOD: Modify Register.
- PROG: Program Start Location.
- PRPC: Program Counter
- TMR: Timer Register.

- GENERAL PURPOSE REGISTERS

The general purpose registers are utilized as:

- Accumulators.
- Intermediate Storage.
- Index Registers.

- PROCESS STATUS WORD (PSW)

The PSW register contains flags concerning interrupts, status of arithmetic logic unit (ALU), current page and priotiry of process. The PSW register definition is summarized in the tabulation shown overleaf (table 5.3-1), and described below.

- Interrupt Flags & Masks:
  The bits 15, 14, 13, and 11 concern interrupts as described in the interrupt section.

- ALT
  This bit (12) is set when the alternative instruction set is executed

- CPU Number.:
  The bits 10-8 contain the number of the CPU in which the process is running. These bits are set by switches (manually) in the CPU. When a running process is saved into its process descriptor area in memory, these bits are stored into memory together with the other PSW bits and thus indicate the number of the CPU in which the process was executing the last time. When a process is again loaded into a CPU, the old contents of bit 10-8 are lost and replaced by the switch contents of the CPU that loaded the process.

Program Status Word (PSW)

BIT NUMBER	NAME	LOAD-ABLE	SET BY	DESCRIPTION
15	TIMER MASK	Y	SOFTWARE	HIGH MEANS: FAST TIMER DISABLED
14	CPU MASK	Y	SOFTWARE	HIGH MEANS: CPU INTERRUPTS DISABLED
13	IO MASK	Y	SOFTWARE	HIGH MEANS: IO INTERRUPTS DISABLED
12	ALT	Y	SOFTWARE	SET WHEN ALTERNATIVE INSTR. SET IS EXECUTED
11	LOCAL-INT	Y	FIRMWARE	SET WHEN A LOCAL INTERRUPT IS GENERATED
10	CPU NMB2	N	HARDWARE	
9	CPU NMB1	N	HARDWARE	CPU NUMBER SET BY SWITCHES
8	CPU NMB0	N	HARDWARE	
7	Z	Y	FIRMWARE	THE ALL ZEROES CONDITION OF ARITHMETIC OPERATIONS (HIGH-ALL ZEROES)
6	S	Y	FIRMWARE	THE SIGN BIT OF THE RESULT OF ARITHMETIC OPERATIONS
5	V	Y	FIRMWARE	THE OVERFLOW CONDITION OF ARITHMETIC OPERATIONS (1 MEANS NO OVERFLOW)
4	C	Y	FIRMWARE	THE CARRY OF ARITHMETIC OPERATIONS
3	PG0	Y	SOFTWARE	PAGE INDICATION. PROVIDES AN OFF SET
2	PG1	Y	SOFTWARE	IN MULTIPLES OF 64K TO DATA ADDRESSES
1	PRI1	Y	SOFTWARE	PROCESS PRIORITY
0	PRI0	Y	SOFTWARE	

NOTE: THE ONLY BITS WHICH CAN NOT BE CONTROLLED BY A LOAD PSW INSTRUCTION ARE 8,9,10.

Table 5.3-1, PSW Register Definition

- Z, S, V, C:

  The bits 7-4 contain status flags from the arithmetic/logic unit (ALU) in the CPU.

  These flags are updated only after arithmetic instructions. When they are updated, the following rules apply:

  The Z is set if output of the ALU was all zeroes, and is otherwise cleared.

  The S is set if output of the ALU had its most significant bit set, i.e. if the output was negative (in 2's complement notation). Otherwise S is cleared.

  The V is cleared if overflow occurred in the ALU, i.e. if positive numbers were added together giving a negative result. If negative numbers were added together giving a positive result. If a negative number was subtracted from a positive number, giving a positive result. V is otherwise set.

  The C is set/cleared according to the function performed in the ALU. After an addition C=1 means carry, after a subtraction C=0 means borrow.

- Memory Page:

  The bits 3-2 designate the memory page in which the process is stored. Bit 3 is the least significant.

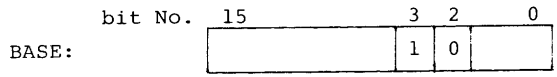
- Priority Bits:

  The bits 1-0 designate the priority of the process. This has significance in connection with process switching caused by interrupts: This is described in the interrupt section. Bit 1 is the most significant.

- BASE REGISTER

A process can be placed in any of the 4 memory pages. The BASE register points to the start of the process within that page.

The BASE value designates by bits 3, 2 the page number, in which the process is located. Thus a process in page 2 for example, must have a starting address obeying the following scheme.



The two page bits are transferred to the process status word (PSW) from the base register, when a process is loaded into the CPU.

- MODIFY REGISTER (MOD)

Most instructions can be modified when preceded by a MODify instruction.

The MODify instruction add or subtract some value to or from the MOD register. In order to anticipate base relative modifications, which are assumed to appear most often, the MOD register is always preset to the value BASE.

Several types of MODification appear:

- BASE relative memory address modification
- PROG relative memory address modification
- PRPC relative memory address modification
- I/O module address modification
- Parameter modification (such as immediate value modification, shift number modification).

- PROGRAM REGISTER (PROG)

the PROG register points to the start of the program, being executed by the process loaded into the CPU.

  - The program can be placed anywhere in page zero of memory (i.e. in locations 0-65535) without changing the program code itself. The only data to be changed if a program is moved to new locations are the PROG and PRPC registers in the processes that use the program.

- PRPC (PROG + PC) REGISTER

The PRPC register points to the instruction next to the one being executed (program counter PC relative to origin of program PROG).

- TIMER REGISTER

This register is used for tying a process to time. The register can be preset to any value (by the instruction LOAD TIMER).

When the PSW bit 15 is 1, the TIMER register is never changed.

When the PSW bit 15 is 0, the TIMER is decremented by one each time a fast timer interrupt occurs. The frequency of timer interrupts is determined in the CPU-SCM module and setable in the interval 10 - 160uSecs.

If, after TIMER register decrement, the TIMER contents is negative and a local *) interrupt is not active, the TIMER is updated:

TIMER<---TIMER + TIMERPRESET

where TIMERPRESET is the contents of the memory cell addressed by BASE+19. Then the CPU jumps to a location with address equal to PROG plus the contents of the memory word addressed by BASE+15. The return address, current PRPC+1 is stored in BASE+16.

*) See section on interrupts.

5.3.3.3 THE STANDARD INSTRUCTION SET

The standard instructions are divided into the following groups:

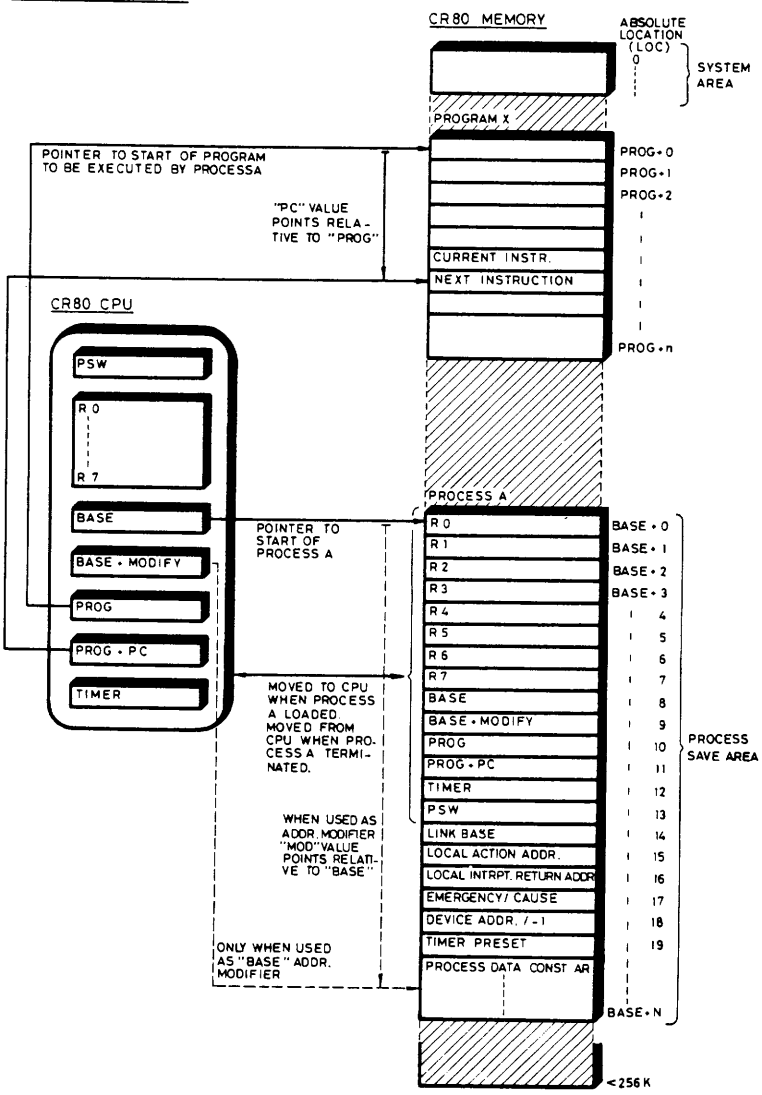
- MODIFY INSTRUCTIONS
- MOVE INSTRUCTIONS
- JUMP INSTRUCTIONS
- ARITHMETIC AND LOGIC INSTRUCTIONS
- SKIP INSTRUCTIONS
- SHIFT INSTRUCTIONS
- SINGLE BIT INSTRUCTIONS
- INPUT/OUTPUT INSTRUCTIONS
- SPECIAL INSTRUCTIONS

The instructions are described in detail in the chapter on the CR80 Instruction Set.

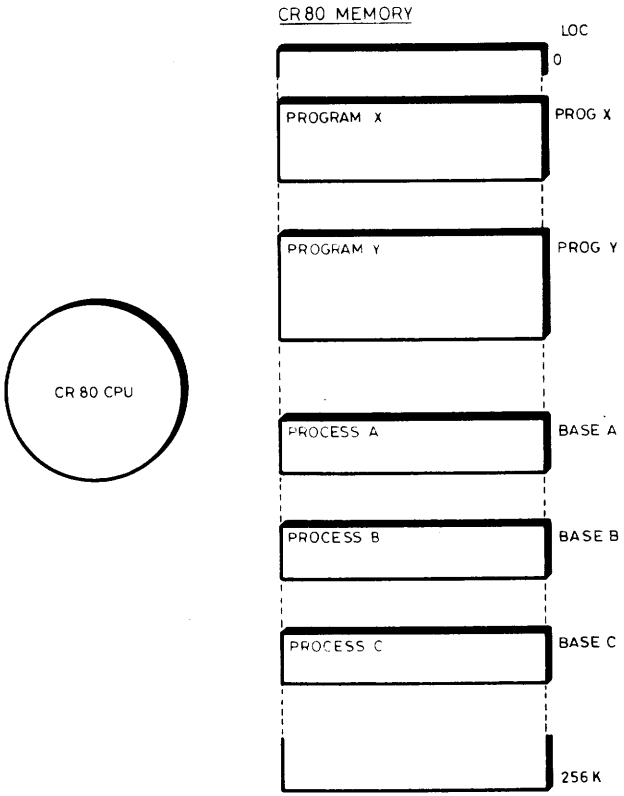
5.3.3.4 EXECUTION AND MULTIPROGRAMMING

The execution of the instructions in the program area, by a process on the CPU, is illustrated overleaf (figure 5.3-1).

Figure 5.3-1 CR80 EXECUTION



- MULTI PROGRAMMING



EXAMPLE 1:

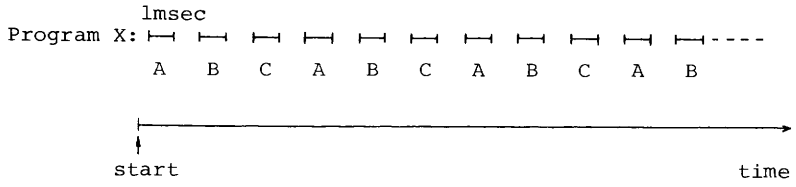
As all Programs are re-entrant, because no instructions can modify the program itself, many processes can execute the same program concurrently without interaction between them. As an example, we let process A, B and C execute program X. By having the "TIMER PRESET" (BASE + 19) of the processes A, B and C containing a value corresponding to 1 millisec. and "LOCAL ACTION ADDR" (BASE + 15) of process:

A point to these instructions in program X: SAVE A, LOADB

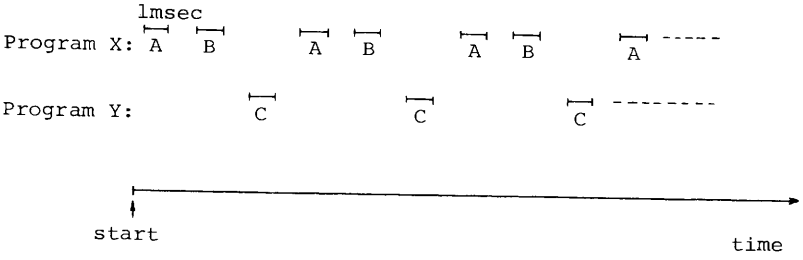
B point to these instructions in program X: SAVE B, LOADC

C point to these instructions in program X: SAVE C, LOADA

when a process executes in the CPU, the "TIMER" is initially loaded with the "TIMER PRESET" value and afterwards decremented until -1, which forces a jump to the instruction pointed at by LOCAL ACTION ADDR. We have now obtained equal time sharing between user A, B and C.



EXAMPLE 2:



The CR80 MINI and TWIN hardware and instruction set greatly facilitates multi-programming by:

- the separation of program code and its execution (process);
- not allowing programs to modify themselves;
- macro instructions LOAD PROCESS and SAVE PROCESS which transfer all pointers and registers in and out of the CPU (about 18 microsec. each); and,
- the automatic timer mechanism.

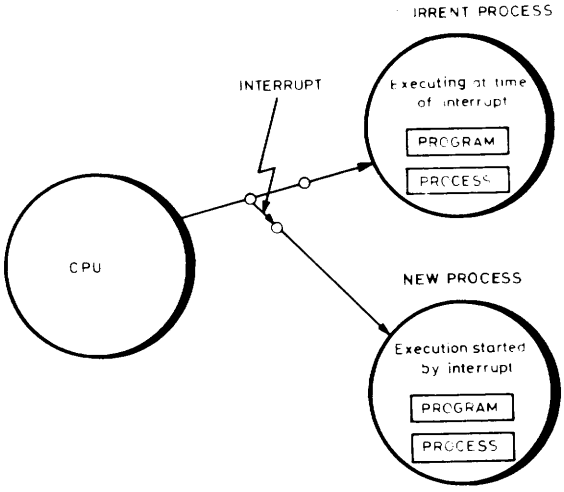
5.3.3.5 MASTER CLEAR

When a Master Clear is detected (power on), the CPU will load a master clear process from page zero. This process is assumed to have a Base: location $2^{16}-2^5$ or $FFE0_{16}$.

The memory module where this process is located is the PROM of the SCM-CPU. The program executed by the Master Clear process is normally used for system initialization and bootloading of programs from disks.

5.3.3.6 INTERRUPT HANDLING

The CR80 MINI and TWIN computers employ an extensive set of interrupt handling features which support multiprocessing in a real time environment. An interrupt directly forces a CPU to close down execution of the process running at the time of the interrupt, and to start the process which corresponds to the interrupt. This is done automatically by hardware and there is no requirement for going through a time consuming interrupt service routine.



CR80 PROCESS SWITCHING WHEN INTERRUPTED

The following type of vectorized priority interrupts are incorporated in the CR80 MINI and TWIN computers:

1. I/O Interrupts incl. Power Failure & Real time Clock
2. CPU Interrupt
3. Fast Timer Interrupt
4. Timeout Interrupts (Inter-
5. Trap Interrupts
6. Parity Error Interrupt

Interrupt sources external to the CPU (Interrupts 1 and 2 may be masked by setting appropriate bits in PSW).

Interrupts local to process being executed in the CPU (Interrupt 3 may be masked by setting appropriate bit in PSW).

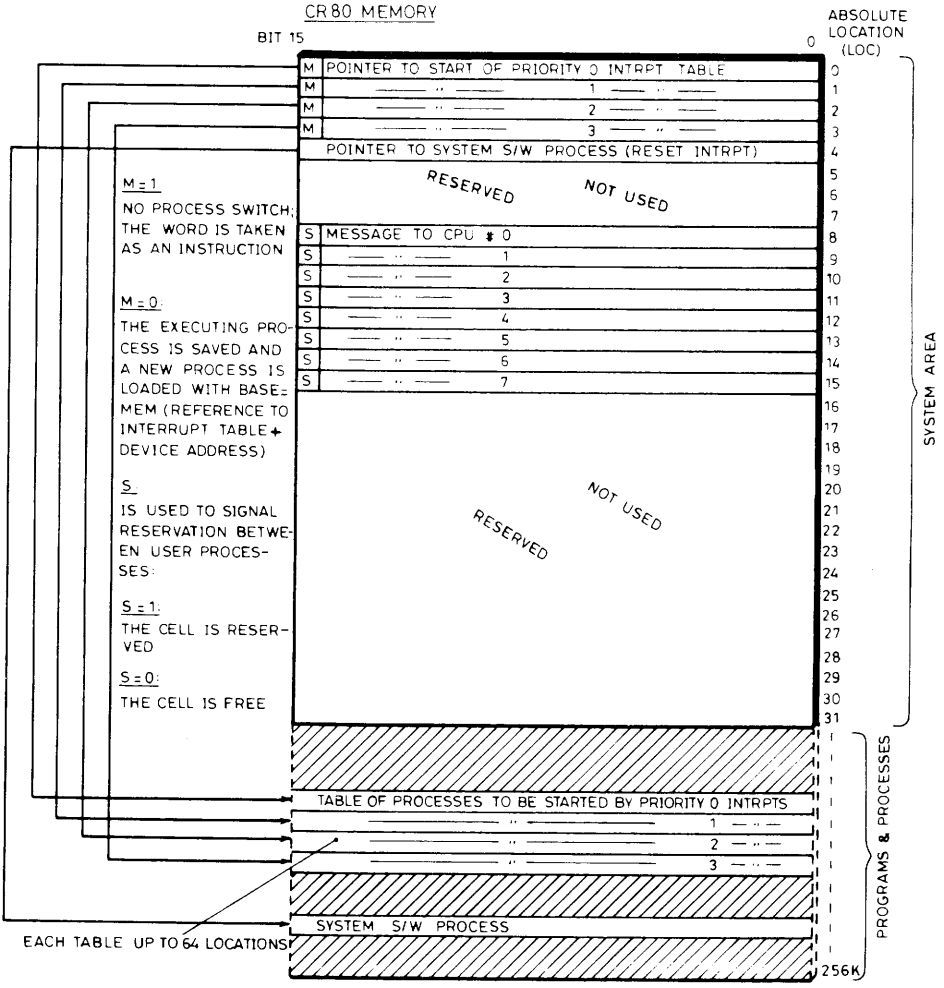
The interrupts make use of the "SYSTEM AREA" described in the next section.

5.3.3.6.1 SYSTEM AREA

The lowest 32 words of memory are reserved for use by the system software (Operating System, Monitor).

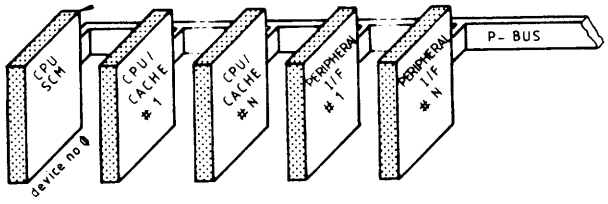
The values stored here by the system software are utilized by the CPU (or CPUs) hardware when interrupts occur.

CR80 MEMORY


CR80 MEMORY
BIT 15
0
ABSOLUTE LOCATION (LOC)
M POINTER TO START OF PRIORITY 0 INTRPT TABLE
M " " 1
M " " 2
M " " 3
POINTER TO SYSTEM S/W PROCESS (RESET INTRPT)
RESERVED
NOT USED
S MESSAGE TO CPU # 0
S " " 1
S " " 2
S " " 3
S " " 4
S " " 5
S " " 6
S " " 7
NOT USED
RESERVED
SYSTEM AREA
M = 1
NO PROCESS SWITCH; THE WORD IS TAKEN AS AN INSTRUCTION
M = 0
THE EXECUTING PROCESS IS SAVED AND A NEW PROCESS IS LOADED WITH BASE=MEM (REFERENCE TO INTERRUPT TABLE + DEVICE ADDRESS)
S
IS USED TO SIGNAL RESERVATION BETWEEN USER PROCESSES
S = 1:
THE CELL IS RESERVED
S = 0:
THE CELL IS FREE
TABLE OF PROCESSES TO BE STARTED BY PRIORITY 0 INTRPTS
" " 1
" " 2
" " 3
SYSTEM S/W PROCESS
EACH TABLE UP TO 64 LOCATIONS
PROGRAMS & PROCESSES
256K

5.3.3.6.2 I/O INTERRUPTS

I/O Interrupts are generated by peripheral devices on the Processor Bus (Data Bus).



Interrupts from the peripheral devices are received by the CPU SCM and stay pending there until they are served by one of the CPU's.

The CPU's with lower process priority (PSW bits 0 and 1) than the pending I/O interrupt from the SCM, compete to read the interrupt from the SCM, the successful CPU receives an 8 bit word from the SCM containing DEVICE NO. and PRIORITY LEVEL (PR) of the interrupting peripheral:

bit No. 7 6 5 4 3 2 1 0

DEVICE NO.	PR

DEVICE NO.: 0-63 (highest device No. first served, if more than one peripheral interrupt with same priority level at the same time)

PRIORITY LEVEL: 0-3 (3 highest priority)

Once the SCM has gotten the interrupt, the priority level (PR) of the interrupt is compared with the pritority of the currently executing process(es) on the CPU(s), contained in Program Status Word (PSW) bits 0 and 1.

The interrupt will be served immediately only if the priority PR is greater than, or equal to, the process(es) currently being executed by the CPU(s).

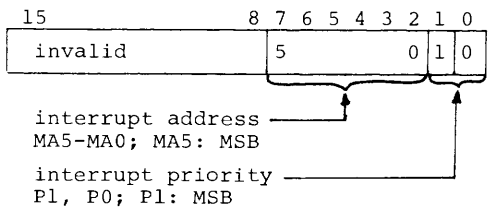
When the interrupt is served, a word is read from the absolute location corresponding to the priority level (locations 0 to 3). If the word is positive (M = 0), it is used as a pointer to an interrupt table, wherefrom the BASE address of the new process to be started by the interrupt is taken at an entry equal to the six bit DEVICE NO. The entry in the interrupt table is reset by loading SYSTEM BASE. The fetch and reset are performed under semaphore protection.

The device address is stored in the new process area in location BASE+18. The base of the interrupted process is stored in location BASE+14 and can be used as a link by the interrupt serving process as a means of retransfering control to the interrupted process.

It should be notes that the SCM-CPU (device No. 0)-generated interrupts (10 ms Real Time Clock (RTC) and power failure) will be transmitted before the queued I/O interrupt, when an interrupted CPU reads. The power failure interrupt has the highest priority and is transmitted before the RTC; furthermore, power failure will disable the receiving of an I/O interrupt, but the queued I/O interrupts are still valid after detection of power failure.

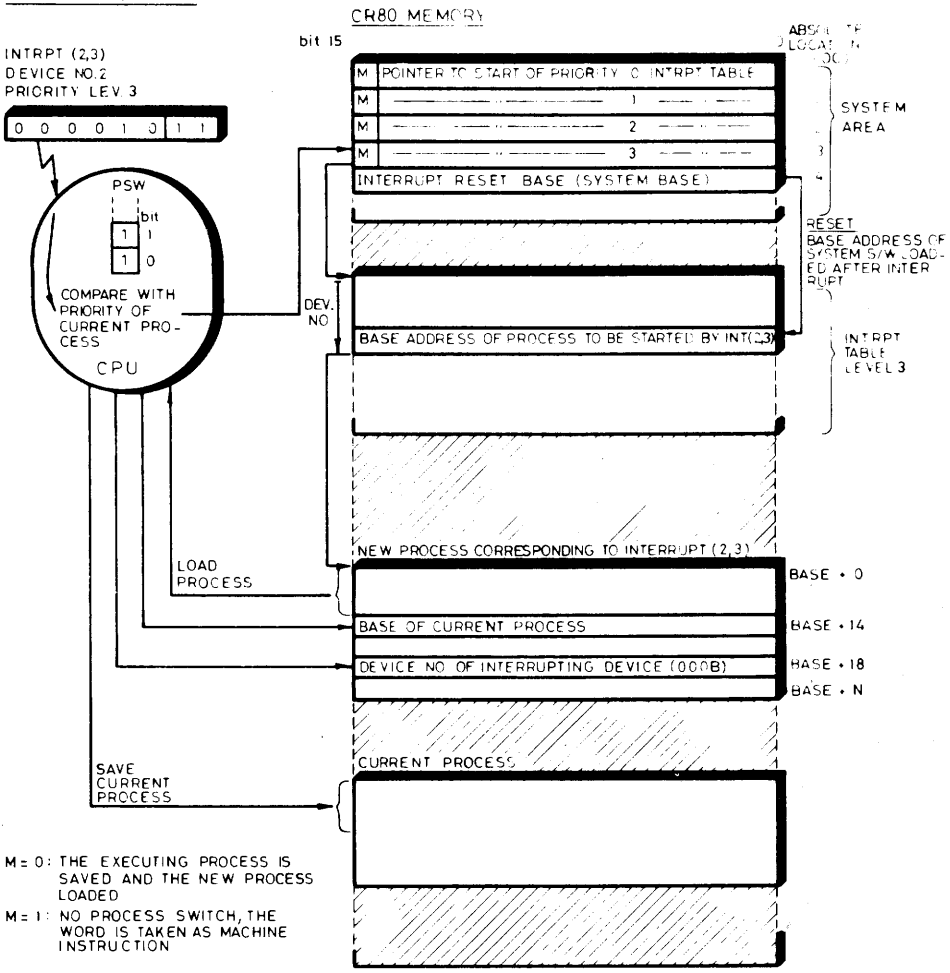
Description	Device No. (MA5-MA0)	Priority
10mS Timer (RTC)	0	0
Power Failure	0	1

Read Interrupt



Parity lines are invalid.

EXAMPLE (M = 0):

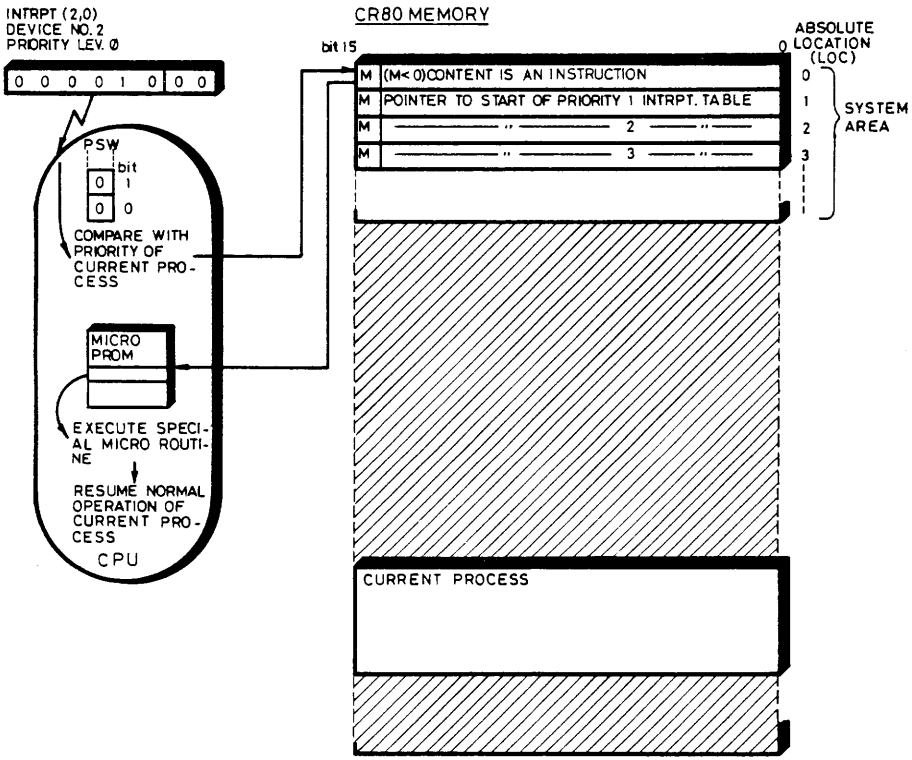


M = 0: THE EXECUTING PROCESS IS SAVED AND THE NEW PROCESS LOADED

M = 1: NO PROCESS SWITCH, THE WORD IS TAKEN AS MACHINE INSTRUCTION

If the word read from the absolute location representing the priority level (location 0 to 3) is negative (M=1), the word is interpreted as being a machine instruction, which is then executed. This is a means for executing especially fast customized micro-routines.

EXAMPLE (M=1):

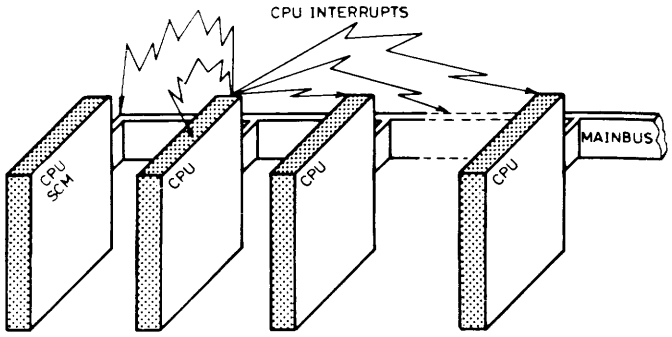


The table shows the I/O Interrupt execution times in cycles:

I/O interrupt execution times in cycles	switch process	start of special micro execution	unsuccessful competition
Compete for interrupt	5	5	5
Decide for specified routine	2	2	
	7	7	
Execute	0	1	
	7	8	
Switch process	41		
	48		

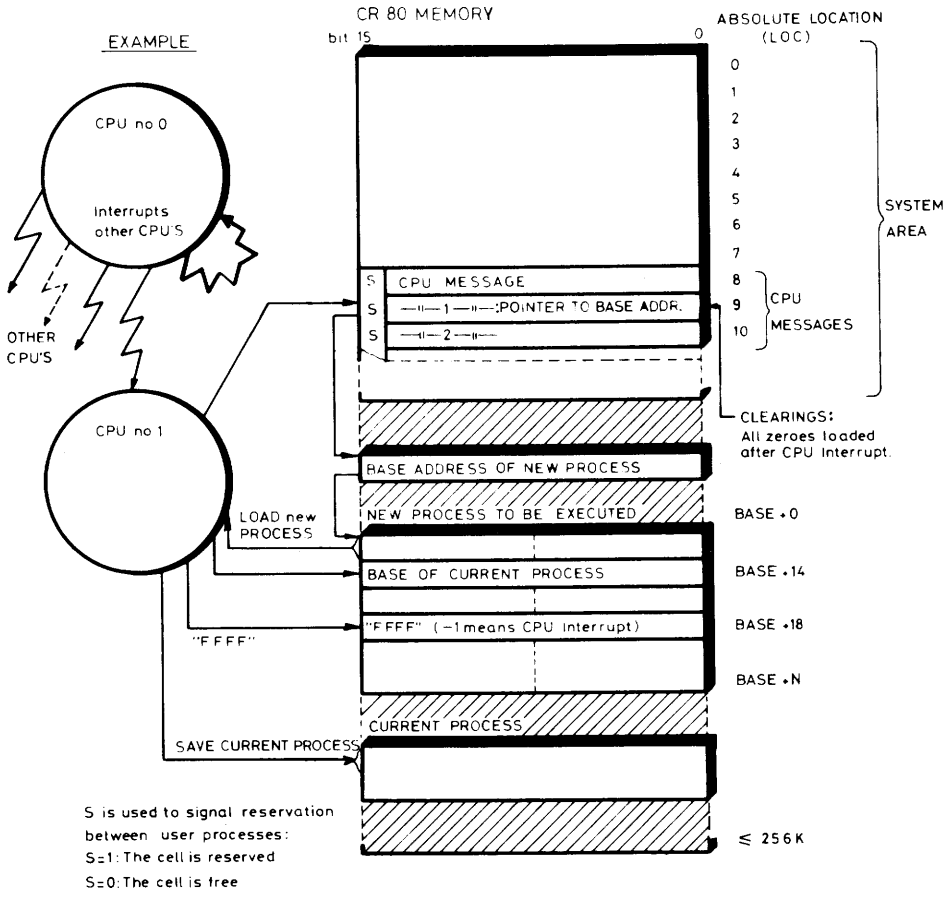
5.3.3.6.3 CPU INTERRUPTS

A CPU interrupt is generated by a CPU executing a CPU INTERRUPT instruction and is broadcasted to the other CPUs and itself on the processor.



The CPU interrupts are received only if the CPU mask bit in PSW (bit 14) is cleared (= 0), CPU interrupt will be pending for those CPUs that have their mask bit set (= 1). The interrupting CPU causes the other CPUs and itself to read their corresponding CPU message word from the "SYSTEM AREA" at absolute location 8+CPU number. If the CPU message word is positive (S = 0), it is used as an absolute pointer to a memory cell containing the BASE address of a new process.

The interrupted CPU saves the current process and loads the new process after clearing the CPU message word (all zeroes). The fetch and clearing of the CPU message is performed with semaphore protection. The base of the interrupted process is stored as a link in the new process area in location Base+14. $FFFF_{16}$ is stored in cell Base+18 in the new process area (this may be used by the process to distinguish between CPU and I/O interrupts). If contents of CPU message is zero, this means no message - no process switching takes place. If contents of CPU message is negative (S = 1), this means that the message is being updated (ensures no clashes between CPUs simultaneously wanting to load the same message word). CPU interrupts are pending during execution of MODIFIED instructions.

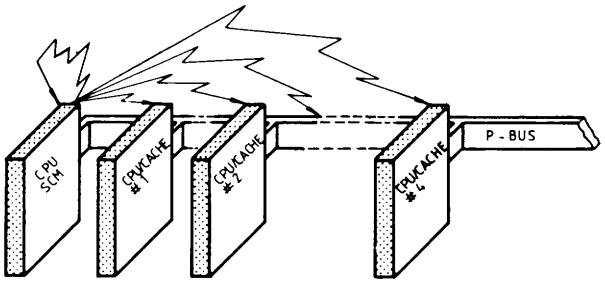


CPU INTERRUPT EXECUTION TIMES IN CYCLES	CPU MESSAGE=0 (no message)	SWITCH PROCESS
GET CPU MESSAGE	6	6
TEST MESSAGE ZERO	1	1
	7	7
SWITCH PROCESS		38
		45

5.3.3.6.4 FAST TIMER INTERRUPTS

The fast timer interrupts are generated by the SCM within a setable interval of 10 - 160us and are treated in all CPU's which have not masked the timer.

FAST TIMER INTERRUPT



A fast timer interrupt is served by decrementing the TIMER register.

Execution times in number of cycles is tabulated below.

Execution Times	Timer DEC	Timer DEC +Timer ACT
DEC Timer	3	3
	3	3
Preset Timer		3
Intern. Int.		10
		16

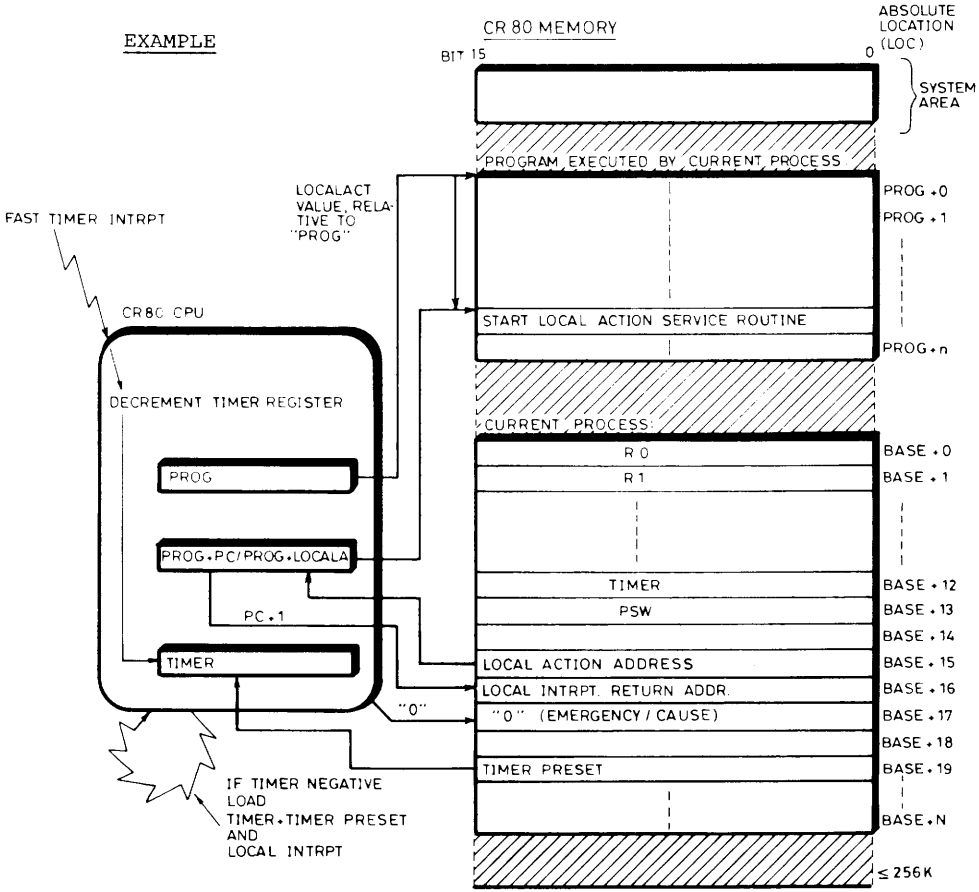
If, after doing this, the TIMER is negative and the local interrupt bit is not set (bit 11 of PSW=1), the TIMER is updated:

TIMER<---TIMER + TIMER PRESET

TIMER PRESET is fetched from location BASE+19. The LOCAL ACTION address is loaded from location BASE+15, and a branch to location PROG + LOCAL ACTION is performed. The return address, Current PC + 1, is stored in location BASE+16. Bit 11 of PSW is set (local interrupt active). The cause code 0 is stored in location BASE+17.

- Emergency action, see Timeout Interrupt description.
- Fast timer interrupts are pending during execution of MODIFIED instructions.

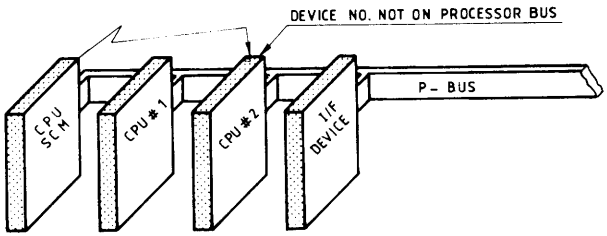
EXAMPLE


CR 80 MEMORY
BIT 15
0
ABSOLUTE LOCATION (LOC)
SYSTEM AREA
PROGRAM EXECUTED BY CURRENT PROCESS
PROG + 0
PROG + 1
PROG + n
START LOCAL ACTION SERVICE ROUTINE
LOCALACT VALUE, RELATIVE TO "PROG"
FAST TIMER INTRPT
CR80 CPU
DECREMENT TIMER REGISTER
PROG
PROG+PC/PROG+LOCALA
PC+1
TIMER
"0"
IF TIMER NEGATIVE LOAD TIMER+TIMER PRESET AND LOCAL INTRPT
CURRENT PROCESS
R 0
R 1
TIMER
PSW
LOCAL ACTION ADDRESS
LOCAL INTRPT. RETURN ADDR.
"0" (EMERGENCY / CAUSE)
TIMER PRESET
BASE + 0
BASE + 1
BASE + 12
BASE + 13
BASE + 14
BASE + 15
BASE + 16
BASE + 17
BASE + 18
BASE + 19
BASE + N
≤ 256K

5.3.3.6.5 TIMEOUT INTERRUPT

A Timeout Interrupt is generated when an addressed memory module or device does not respond. Only the addressing CPU is interrupted.

TIMEOUT INTERRUPT
DEVICE NO. NOT ON PROCESSOR BUS



If no other local interrupts are active (bit 11 of PSW=0), when a Timeout interrupt is generated, a LOCAL ACTION address is loaded from location BASE+15. A branch to location PROG + LOCAL ACTION is performed and the return address, Current PC + 1, is stored in location BASE+16.

Bit 11 of PSW is set. The cause code 3 is stored in location BASE+17.

- Emergency Action

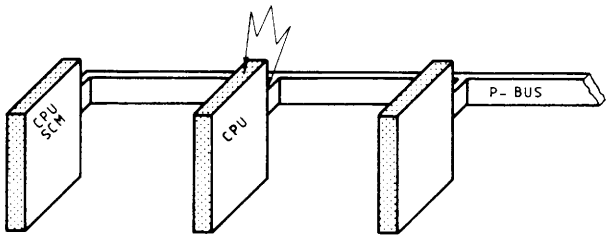
  If a local interrupt was already active (bit 11 of PSW = 1) the following emergency action is taken:

  From location No. FFFE an absolute address is fetched to which a branch takes place (EMERGENCY ACTION). The same process is still executing. PRPC is the only register changed. Current PC+1 is stored in location BASE+17.

- Timeout interrupt will clear a pending MODIFY.

5.3.3.6.6 TRAP INTERRUPTS

A Trap Interrupt is generated by execution of a Trap or illegal instruction, or an attempted execution of a not implemented instruction.



A Trap interrupt causes a branch to PROG + LOCAL ACTION if no local interrupts are active. Simultaneously, the return address: Current PC + 1 is stored in location BASE+16 and the interrupt cause code 1 is stored in BASE+17.

If local interrupts are active, the Trap interrupt causes a branch to EMERGENCY ACTION (see Timeout interrupt).

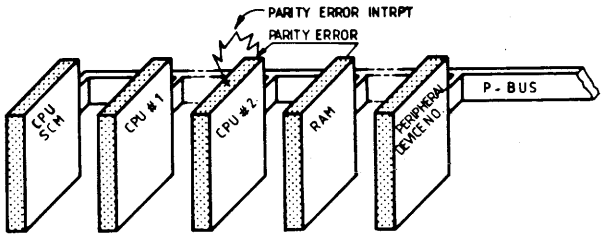
- TRAP interrupt will clear a pending MODIFY.

5.3.3.6.7 PARITY ERROR INTERRUPT

The CPU is equipped with memory parity. With each byte stored in memory by the CPU, a parity bit is also stored.

When the CPU reads data from memory, 1 parity bit is generated for each byte read and compared to a parity bit read from memory.

If the two parity bits differ, the CPU generates a Parity Error interrupt.



The Parity Error interrupt is similar to the Timeout interrupt except for the cause code, which for Parity Error is 2.

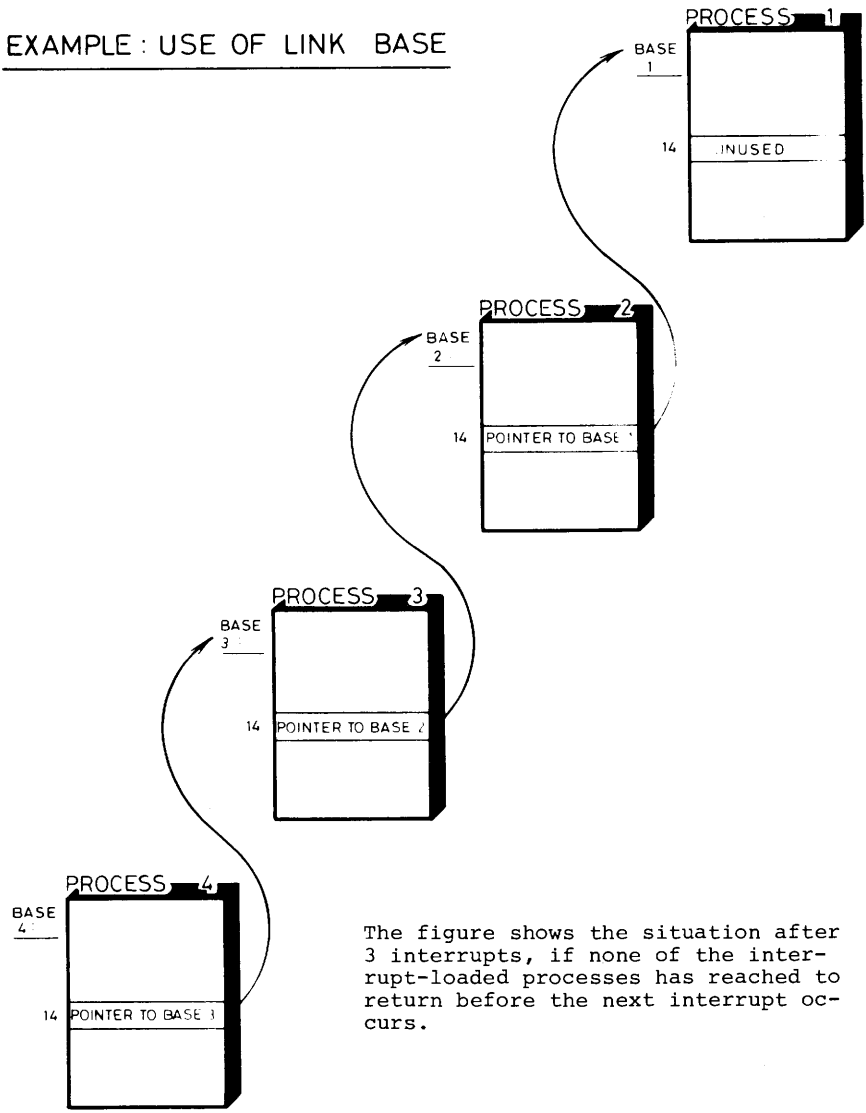
- PARITY ERROR interrupt will clear a pending MODIFY.

5.3.3.6.8 USE OF LINK BASE (BASE+14)

A successful I/O interrupt or CPU interrupt forces the CPU to switch process.

The BASE address of the saved process is stored in LINK BASE (BASE+14) of the loaded process. The loaded process thereby has a link back to the process that was forced out when the interrupt occurred. Return to the saved process by the loaded process is by using the instruction "LOAD PROCESS" and the BASE address stored in the LINK BASE (BASE+14) of the loaded process. Linking is done to any depth of processes and interrupts.

EXAMPLE : USE OF LINK BASE



The figure shows the situation after 3 interrupts, if none of the interrupt-loaded processes has reached to return before the next interrupt occurs.

5.4 Mechanical, Electrical and Interconnection Specifications

5.4.1 Introduction

The mechanical, electrical and interconnection specifications for the CR80 MINI and TWIN unmapped computers are, to a great extent, identical to the CR80 MAXIM and FATOM mapped computers. Therefore, only those items which differ are specified here; the remaining specifications are found in the CR80 MAXIM and FATOM chapter on PU and CU crate Electrical, Mechanical and BUS I/F specifications.

5.4.2 CR80 MINI and TWIN Processor Unit (PU)

The PU is packaged in a mechanically self-contained unit. The modules are housed in cages called PU crates. These crates contain the CR80 MINI or TWIN Processor Unit systems central processing, power and main memory; and, in non-dualized CR80 MINI systems, also contain the I/O modules. The number of physical modules installed in the crates is dependent on the desired PU capacity.

The standard types of PU crates available are described in the data sheets.

The block diagram of a typical PU configuration is shown overleaf. The similarities and differences between the unmapped CR80 MINI and TWIN PU's and the mapped CR80 MAXIM and FATOM PU's are given below:

- PU Crate

- the unmapped CR80 MINI and TWIN PU crates require only one transfer bus, P-Bus and different system control connections (bus arbitration and interrupt signalling).

- PU Modules

- Bus termination modules (MBT) are identical.

- RAM modules can be used in the single-port version for unmapped systems.

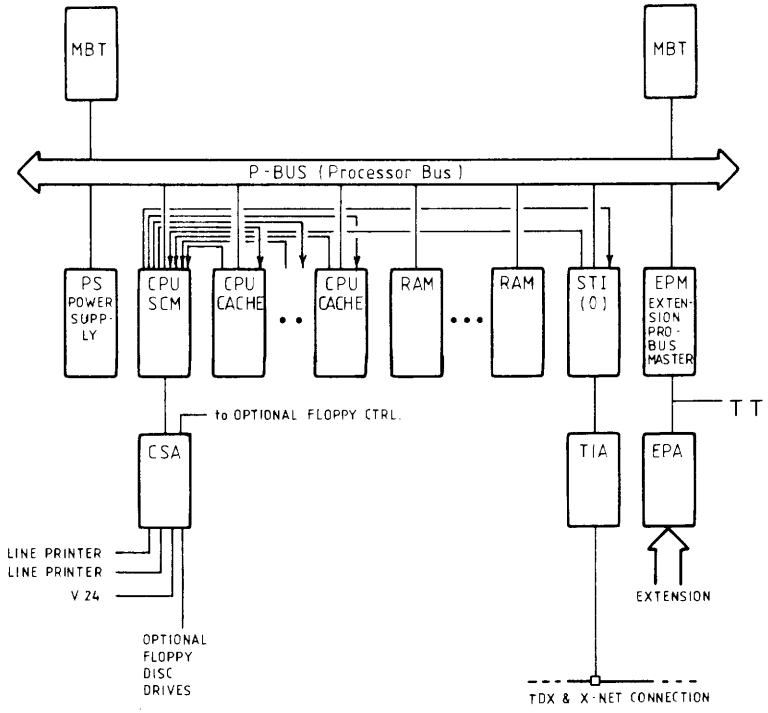
- CPU-SCM and its adapter, CSA, are special modules for unmapped computers; the CPU-SCM module has no connections on the C-bus J2 connector.

- CPU CACHE module hardware is identical to the mapped CPU CACHE's hardware but the firmware corresponds to the standard unmapped instruction set and the unmapped system control connections. The module requires connections to its channel bus connector for DC power input.

- STI, TDX-Bus - CR80 interface module has hardware identical to the mapped version but firmware adapted to the unmapped operation. The module require a special bus jumper when installed in unmapped systems, because it only interfaces via the C-Bus connector. The TIA, TDX-Bus Interface Adapters, are identical for mapped and unmapped systems.

- The EPM, EPA modules are only used in unmapped systems for extending the physical allowable distance between modules, and to expand the number of accessible Peripheral Modules.

Figure 5.4.2-1
TYPICAL PU IN CR80 MINI and TWIN UNMAPPED COMPUTERS



5.4.2.1 Crate, Processor Unit (PU)

The general specifications for the unmapped CR80 MINI and TWIN PU-Crates are identical to those for the mapped CR80 MAXIM and FATOM PU-crates, except that no C-bus is implemented in the PU-Crate for unmapped systems, and that some of the module slots are prepared for insertion of Peripheral Modules.

5.4.2.2 Processor Bus (P-bus)

The P-bus specifications are identical to those of the mapped computers CR80 MAXIM and FATOM, with following exceptions:

- Interrupt lines INR and INA are used for I/O interrupt transmission on the unmapped P-bus.
- AD18 and AD19 constant high.
- AE(L) constant low (jumper in bus termination board).

The address lines are as defined in the following tables:

TABLE 5.4.2.2.a
ADDRESSING SPECIFICATIONS FOR ADDRESS SOURCING MODULE (CPU/DMA) IN UNMAPPED PROCESSING UNIT

ADDRESS from address sourcing module

R/W (L/H-0/1)	LSI,LSO (L/H-0/1)	AD17 (L/H-0/1)	AD16	AD(15:12)	AD(11:6)	AD(5:0)	Address OPERATION	Destination Module
0	00	*)	X	X	CM	MA	Read from Per. Module "MA". Command "CM" is performed by Per. module.	Peripheral Module
1	00	**)	X	X	CM	MA	Write to Per. Module "MA". Command "CM" is performed by Per. module.	Peripheral Module
0	01 or 10 or 11	Physical memory address					Read word in location designated by AD(17:0)	Memory
1	01	Physical memory address					Write lower byte in location designated by AD(17:0)	Memory
1	10	Physical memory address					Write upper byte in location designated by AD(17:0)	Memory
1	11	Physical memory address					Write word in location designated by AD(17:0)	Memory

*) RIO 1, SIO 0
**) WIO 1, CIO 0

TABLE 5.4.2.2.b

ADDRESSING SPECIFICATIONS FOR PERIPHERAL AND MEMORY MODULES

Addressing Specifications for Peripheral Module

R/W (L/H-0/1)	LSI,LSO (L/H-0/1)	AD(19:18) (L/H-0/1)	AD(17)	AD16	AD(15:12)	AD(11:6)	AD(5:0)	OPERATION
0	00	1)	X	X	X	CM	MA	Read from Per. module "MA". Command "CM" is executed by Per. module
1	00	1)	X	X	X	CM	MA	Write to Per. module. "MA". Command "CM" is executed by Per. module.

Addressing Specifications for Memory Module

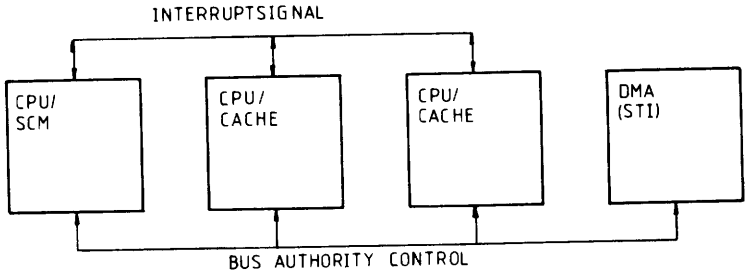
0	01 or 10 or 11	1)	PHYSICAL ADDRESS	Read word in location designated by AD(19:0). Output full word & parity to bus.
1	01	1)	PHYSICAL ADDRESS	Write byte on AD(7:0) into lower byte of location designated by AD(19:0)
1	10	1)	PHYSICAL ADDRESS	Write byte on AD(15:8) into upper byte of location designated by AD(19:0)
1	11	1)	PHYSICAL ADDRESS	Write word in AD(16:0) into word location designated by AD(19:0)

1) Note: With no map module in processing unit, AD(19:18) is high.

5.4.2.3 PU-Module Interface Connector (J3)

The interface connector (J3) is located in the same location in unmapped (CR80 MINI and TWIN) and mapped (CR80 MAXIM and FATOM) PU's, except for the signalling used for SCM - CPU and DMA intercommunication. This section defines the differences.

The signals implemented between modules are defined in the figure below:



Pin layout for the three types of modules are defined in the following tables: 4.4.2.3a - c

Signal	REAR VIEW B		A	Signal
		43		
	GND	42	GND	
	P1	41	P0	
	GND	40	INT(L)	
	GND	39	CC(H)	
	TI(L)	38	CPI(L)	
	GND	37	PFL(L)	
	LBG(L)	36	OAC(L)	
	GND	35	GND	
	BG0(L)	34	BRQ0(L)	
	BG1(L)	33	BRQ1(L)	
	BG2(L)	32	BRQ2(L)	
	BG3(L)	31	BRQ3(L)	
	BG4(L)	30	BRQ4(L)	
	BG5(L)	29	BRQ5(L)	
	BG6(L)	28	BRQ6(L)	
	GND	27	GND	
	+5V	26	+5V	
	GND	25	GND	
		24		
		23		
		22		
		21		
64 wires flat cable to CSA		20	These signals are the interface between	
		19	the CPU SCM and the CSA for communication	
		18	with the system console and printers.	
		17		
		16		
		15		
		14		
		13		
		12		
		11		
		10		
		9		
		8		
		7		
		6		
		5		
		4		
		3		
		2		
		1		

Table 5.4.2.3.a
CPU-SCM Module Interface Connector (J3)

Signal	REAR VIEW B		A	Signal
	GND	43	reserved, CR	
	GND	42	reserved, CR	
	reserved, CR	41	GND	
	reserved, CR	40	reserved, CR	
	GND	39	reserved, CR	
	reserved, CR	38	GND	
	reserved, CR	37	reserved, CR	
	GND	36	reserved, CR	
	reserved, CR	35	BRQ(L)	
	GND	34	BG(L)	
		33		
		32		
		31		
		30		
		29		
		28		
		27		
		26		
		25		
		24		
		23		
		22		
		21		Available for 64 wires flatcable connection to adapter
		20		
		19		
		18		
		17		
		16		
		15		
		14		
		13		
		12		
		11		
		10		
		9		
		8		
		7		
		6		
		5		
		4		
		3		
		2		
	Not useable	1	Not useable	

Table 5.4.2.3.b
DMA Module Interface Connector (J3)

Signal	REAR VIEW B	A	Signal
	GND	43	CPI(L)
	GND	42	INT(L)
	GND	41	P1
	GND	40	P0
	GND	39	TI(L)
	GND	38	LBG(L)
	GND	37	
	GND	36	FIN(L)
	GND	35	BRQ(L)
	GND	34	BG(L)
		33	
		32	
		31	
		30	
		29	
		28	
		27	
		26	
		25	
		24	
		23	
		22	
NC		21	NC
		20	
		19	
		18	
		17	
		16	
		15	
		14	
		13	
		12	
		11	
		10	
		9	
		8	
		7	
		6	
		5	
		4	
		3	
		2	
		1	

Table 5.4.2.3.c
CPU-CACHE Module Interface Connector (J3)

5.4.2.3.1 ELECTRICAL INTERFACE SPECIFICATION

The electrical interface line specifications in terms of upper or lower current limits are given in the tabulation below.

CPU and DMA Module arbitration lines:

BRQ (6:0)

BG (6:0)

Driver: $I_{OL} \geq 48$ mA

Receiver: $I_{IH} \leq 100$ uA

$I_{IL} \leq 2$ mA

LBG:

Open collector signal.

Driver: $I_{OL} \geq 48$ mA

$I_{OH} \leq 100$ uA

Receiver: $I_{IH} \leq 100$ uA

$I_{IL} \leq 2$ mA

Interrupt Lines:

P0, P1, INT(L).

Driver: $I_{OL} \geq 60$ mA

Receiver: $I_{IH} \leq 100$ uA

$I_{IL} \leq 2$ mA

PF:

Open collector signal:

Driver: $I_{OL} \geq 48$ mA

$I_{OH} < 100$ uA

Receiver: $I_{IH} \leq 100$ uA

$I_{IL} \leq 2$ mA

5.4.2.3.2 Functional Description of Signals

- Bus Authority Control

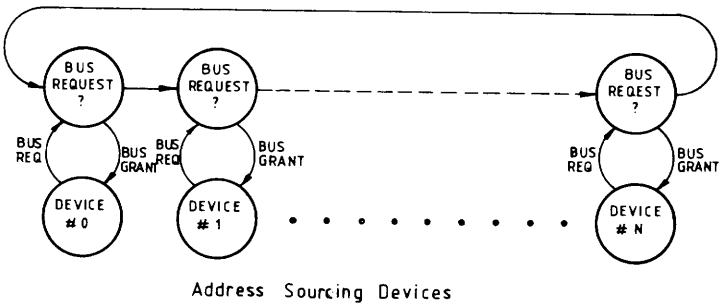
The bus arbitration function for the CPU & DMA modules connected to the processor bus is located in the SCM CPU.
The arbitration is controlled by means of two signals between each of the address-sourcing modules and the SCM.

BRQ(L) issued as request for bus access
BG(L) issued to the requesting module when bus access is granted.

A special signal used by the CPU's during semaphore operations is implemented, LBG(L), which allows for one CPU to make more transfers on the bus while the other CPU's are kept back.

The modules are serviced cyclically, meaning that all modules have the same access rights when requesting. The scheme used is illustrated in figure below.

The timing specifications for the signals are given in figure 4.4.2.3.2-1.



Address Sourcing Devices

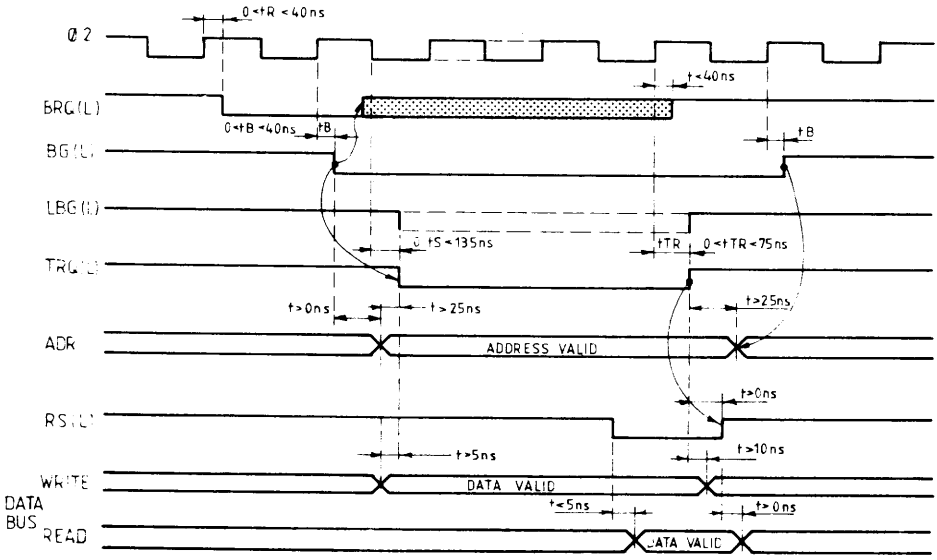
- Interrupt Signals

The interrupt signals P0, P1 and INT are issued by the SCM to the CPU and contain the status of SCM's I/O interrupt queue as defined below:

P1	P0	INT	DESCRIPTION
0	0	0	Priority 0 is highest priority in queue
0	1	0	Priority 1 is highest priority in queue
1	0	0	Priority 2 is highest priority in queue
1	1	0	Priority 3 is highest priority in queue
0	0	1	No valid interrupt pending

The CPU interrupt signal, CPI(L), is a separate connection between the CPU's and used for issuing interrupts to the connected CPU's.

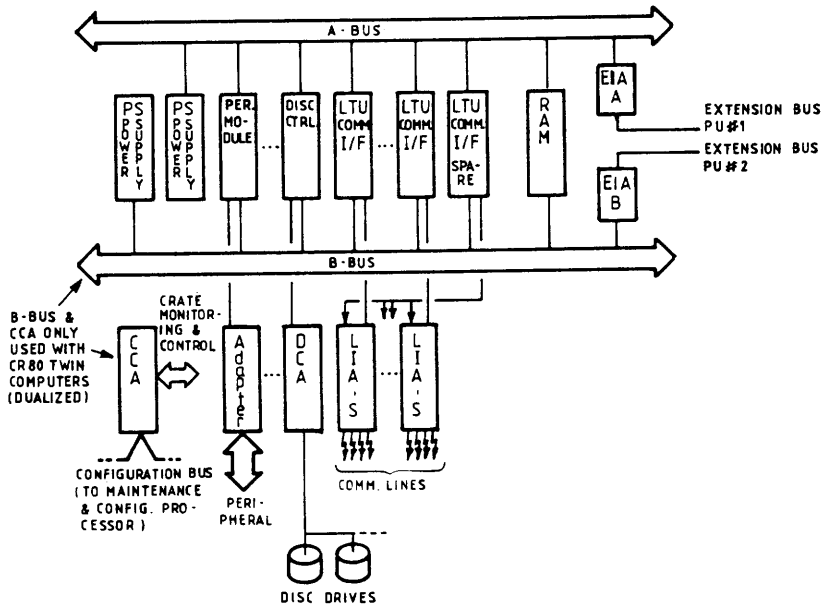
Figure 5.4.2.3.2-1
TIMING SPECIFICATION FOR CR80 UNMAPPED CPU AND DMA MODULES



5.4.3 CR80 MINI and TWIN Channel Unit (CU)

The channel unit (CU) used with the unmapped CR80 MINI and TWIN computers is identical to the mapped CR80 MAXIM and FATOM CU, except that the data channel interface modules CIA-A and B are replaced by the extension bus interface modules EIA-A and B.

The general configuration of the CU is shown below. For detailed specifications, refer to the chapter on CR80 MAXIM and FATOM mapped computers. The module locations and slot definitions are found in the CR80 data sheets and depends on the selected CU-crate type.

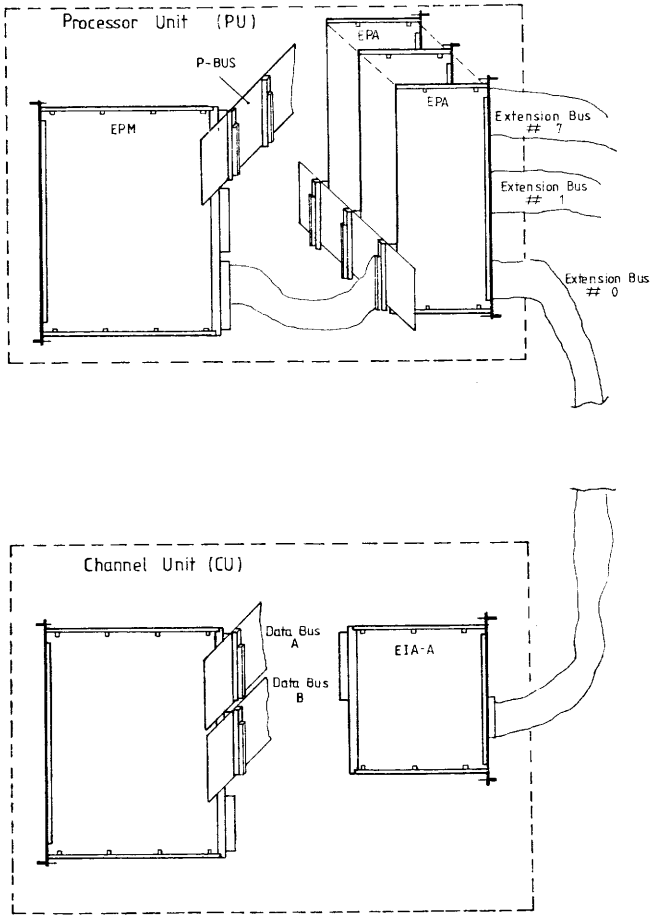


5.4.4 Extension Bus

The extension bus performs the interface between the CR80 MINI or TWIN Processor Units (PU) and a Channel Unit (CU) in unmapped systems. The bus is a point-to-point connection, interfaced in the master end via an EPA (Extension Processor Adapter) and an EPM (Extension Processor Master) to the PU's processor bus. In the slave end of the bus, an EIA-A or EIA-B performs the interface to data bus A or data bus B in the CU. The design of the EPM allows for connection of up to eight EPA's and thereby up to eight CU's to one EPM as shown in figure 5.4.4-1.

In the following subsections, the Extension Bus physical and electrical interface specifications are given.

Figure 5.4.4-1
Extension Bus



5.4.4.1 Physical Interface

The Extension Bus is a 60p flat cable arranged in 30 twisted pairs for differential transmission. The interface connectors are located at the rear of the units as follows:

Channel Unit:	On front panel of EIA-A and EIA-B.
Processor Unit:	On front panel of EPA.

The pin layout for the connectors is defined in table 5.4.4-1.a.

1.	RS(H)+	31.	ADA14+
2.	RS(H)-	32.	ADA14-
3.	IRE(H)+	33.	ADA13+
4.	IRE(H)-	34.	ADA13-
5.	SCK(L)+	35.	ADA12+
6.	SCK(L)-	36.	ADA12-
7.	IRC(L)+	37.	ADA11+
8.	IRC(L)-	38.	ADA11-
9.	MC(H)+	39.	ADA10+
10.	MC(H)-	40.	ADA10-
11.	SIN(H)+	41.	ADA9+
12.	SIN(H)-	42.	ADA9-
13.	R/W(L/H)+	43.	ADA8+
14.	R/W(L/H)-	44.	ADA8-
15.	TRF(L)+	45.	ADA7+
16.	TRF(L)-	46.	ADA7-
17.	ADA21+	47.	ADA6+
18.	ADA21-	48.	ADA6-
19.	ADA20+	49.	ADA5+
20.	ADA20-	50.	ADA5-
21.	ADA19+	51.	ADA4+
22.	ADA19-	52.	ADA4-
23.	ADA18+	53.	ADA3+
24.	ADA18-	54.	ADA3-
25.	ADA17+	55.	ADA2+
26.	ADA17-	56.	ADA2-
27.	ADA16+	57.	ADA1+
28.	ADA16-	58.	ADA1-
29.	ADA15+	59.	ADA0+
30.	ADA15-	60.	ADA0-

Table 5.4.4.1.a
EXTENSION BUS, Pin layout

5.4.4.2 Functional Description of Signals

The EXTENSION BUS consists basically of two parts: the data transfer part and the interrupt transfer part.

The data transfer is performed by multiplexing the address from the master (PU) and data from master/slave on the same lines. The multiplexing is performed in accordance with the handshaking signals on the interfaced buses (processor bus and data bus).

The interrupt transfer is performed by means of an intermediate storage in the EPM and EIA-A and B to allow long transmission paths.

The bus signals are defined as follows:

Data-Address-Control Signals

ADA0-ADA21	Multiplexed between address lines (AD0-AD17, BS0-BS1, LS0-LS1) and data lines (DA0-DA15, LP-UP).
MC(H)	Masterclear
TRF(L)	Transfer TRF(L) = TRQ(L) or AE(L). Transfer is used for handshaking together with Response RS(H).
RS(H)	Response is used for handshaking together with TRF(L).
R/W(L/H)	Read/Write controls the data transfer direction.

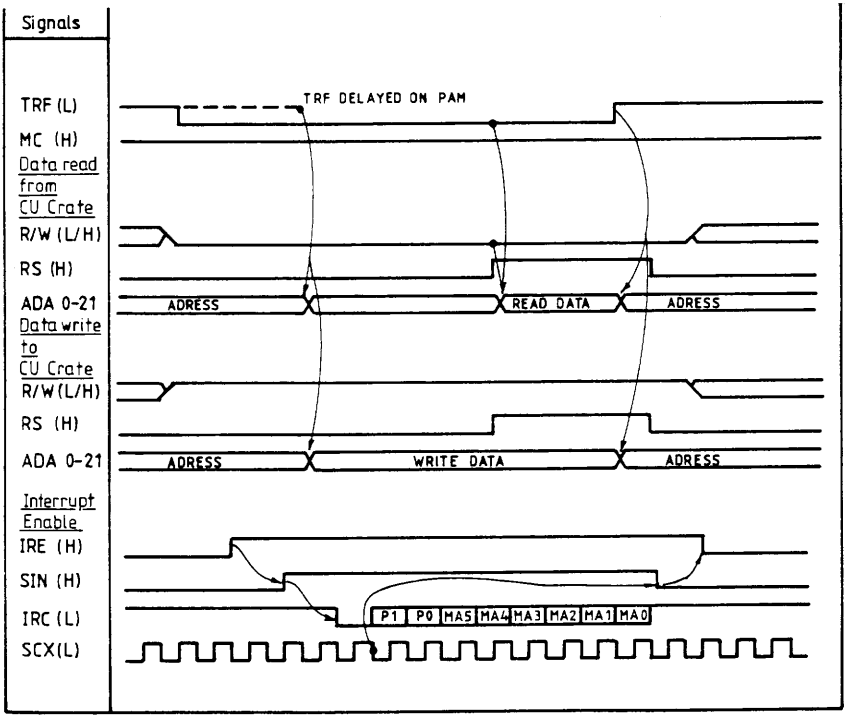
Interrupt Signals

IRE(H)	Interrupt Enable. An interrupt is found by the interrupt transmission circuit on the EIA.
SIN(H)	Send Interrupt. An interrupt is requested from the EPM to the EIA.
IRC(L)	Interrupt code which is sent from the interrupt transmission circuit in a 9-bit code (one interrupt start bit and eight interrupt bits) synchronous with the Shift Clock (SCK).
SCK	Shift Clock. A clock sent from the EIA module to the interrupt receiving shift register in the EPM.

5.4.4.3 Electrical I/F

The Extension Bus employs differential drivers/receivers with termination at the driver end. Precautions should be taken so the common mode voltage at the receiver does not exceed +/- 5V.

5.4.4.4 Timing



AMOS, the Advanced Multiprocessor Operating System for the CR80 MINI and TWIN Computers, is together with the standard commercial packages CROPS and CRMINI forming the integrated commercial EDP system available with the CR80 MINI (Model 801) computers.

AMOS is particularly suited for use in real time systems such as this CR80 MII (Model 831) Front End Processor, interfacing IBM type communications lines ar devices to an ICL Mainframe. In addition AMOS also supports Batch- ar development environments.

6. AMOS - CR80 STANDARD SOFTWARE FOR NON-MAPPED SYSTEMS

6.1 Introduction

The CR80 Advanced Multi Processor Operating System, AMOS, is the standard operating system for non-mapped CR80 MINI and TWIN range of computers.

AMOS supports CR80 single unmapped PE configurations with up to 256K words of main memory and 8 CPUs and a virtually unlimited amount of peripheral equipment.

AMOS is particularly suited for use in real-time systems, but, also supports other environments like software development and batch.

AMOS is compatible with DAMOS at the level of input/output, whereas lower level services like process management and interprocess communication, are different.

A comprehensive suite of software development tools is available under AMOS.

The following languages are supported:

- assembler
- the CR80 system programming language SWELL
- Pascal
- Cobol

The AMOS standard software is described in the following sections. The description is divided into:

- Overview of AMOS
- Compatibility with DAMOS
- Kernel, the AMOS operating system kernel

- AMOS Input/Output, the AMOS standard interfaces to peripheral I/O equipment including the AMOS file management system
- System initialization
- Operating System for SW development
- Programming Languages
- Utility Programs

6.2 Overview of AMOS

The Advanced Multiprocessor Operating System (AMOS) for the CR80 computers is a hierarchical structure of system components, where each component in the layer builds on the lower layers.

AMOS consists of the following components:

- an operating system kernel including
  a memory management module
  a modular set of device drivers
- a unified input/oputput system including
  a file management system

The modularity of AMOS makes it possible to build systems where not all components are used, where new modules are included, or where certain components are replaced by customerized modules to fulfil very special requirements. In a small application with no need for backing store files the file management system need not be included. Unused device drivers can be left out and new device drivers can be easily included in the system.

Kernel

The AMOS Kernel is the lowest of the AMOS layers.

The Kernel is responsible for:

- management of software processes
- interprocess communication
- management of the CPUs
- the lowest level of device handling (interrupt handling)

Process Management

Process management is concerned with implementation of software processes as a data type and with implementation of the functions available for operating on processes.

A process may be defined as an incarnation of the data transformations obtained by execution of a program in a given context. A context is taken to mean a set of CPU registers (CPU resident or saved).

Processes are uniquely identified by symbolic names throughout a CR80 computer. A process may thus be referred to without a requirement to know in which memory locations it resides. The only instance at which such a knowledge need be specified is when a process is created.

The following functions are available for operating on processes:

- create process
- remove process
- start process
- stop process

Process Communication

Interprocess communication is based on the exchange of messages and answers.

Two concurrent processes can cooperate by sending messages to each other. A message consists of a number of machine words. Messages are transmitted

from one process to another by means of message buffers selected from a pool.

Efficiency is obtained by the queuing of buffers, which enables a sending process to continue immediately after delivery of a message or an answer regardless of whether the receiver is ready to process it or not.

AMOS implements three independent kinds of messages:

- messages
- system messages
- path messages

System messages are reserved for use by the AMOS operating system itself.

The Kernel provides two synchronization mechanisms - signals and parent signals. These mechanisms do not transfer information between processes but can only be used for synchronizing processes.

Signals are available to all processes. A process may send a signal to another process. This causes a flag to be set in the process control block of the receiving process.

Parent signals are sent implicitly by the Kernel to a parent process when a child process of the parent process retires or encounters a run time fault.

All communication and synchronization mechanisms may be awaited separately or in any combinations by means of a general Wait-event function.

The Wait-event function is also used for awaiting delays - cyclic or acyclic - and notifications from I/O devices.

Interrupt Handling

The Kernel controls the input/output interrupt facilities of the processor.

A process may reserve an interrupt and thereby establish monopoly of using that interrupt until it is released again.

On the arrival of an interrupt, hardware saves the executing process and loads the process the identification of which is fetched from an interrupt table.

If the interrupt is not waited for, a dummy interrupt process is loaded; it increments the occurrence count of the interrupt and reloads the former executing process.

If the interrupt is awaited by a process (via Wait-event), the Kernel interrupt process is loaded. It makes the waiting process enter the execution state.

CPU Management

CPU management is responsible for the allocation of CPUs to the preempted processes.

CPUs are handled by the Kernel as separately identifiable objects addressed by a symbolic name.

CPUs are grouped in pools. Each pool has its own ready list(s) of processes and is scheduled separately. When a process is created, it is determined which pool of CPUs it shall execute on.

The CPU allocation algorithm works independently for each CPU pool. The CPU allocation algorithm is invoked:

- when a process calls a wait function to receive a not yet occurred event

- when a CPU has been allocated to a process a predefined amount of time (tunable)

Memory Management

The memory management module administers a shared pool of memory segments. Segments may dynamically be allocated to processes and deallocated after use.

Device Drivers

Each device connected to a CR80 system is controlled by a device driver, which is a software process.

The main purpose of a CR80 device driver is to bridge the gap between the hardware interface and the CR80 process concept.

Device drivers are implemented as CR80 processes establishing a generalized interface to the user processes fitting into the CR80 I/O system. The generalized interface defines standard formats for driver operations and the command codes are unified where possible.

User processes communicate with drivers by using the send system message and wait system answer functions of the monitor. These functions may be used indirectly by calling the I/O system or directly from a user process (for instance to use a device in some specialized way).

A device driver has exclusive access to the corresponding device. Any use of a device must therefore be performed via the actual device driver which then is able to manage and control the access rights of the requesting user processes, granting mutual exclusion, etc. when necessary.

The device drivers check the hardware equipment, remedy errors if possible and report serious failures.

A large number of different device drivers exist, interfacing a wide range of hardware equipment to the CR80:

- terminals attached via V24 interface
- large disks (12 - 300 MB)
- diskettes
- line printers
- card readers
- DMA channels
- magnetic tape transports
- communication lines and terminals interfaced via LTUs for a variety of protocols including X25 and BSC
- the TDX communication system

The device drivers for disk and diskette drives have an interface which deviates from the standard and should only be accessed indirectly via the File Management System.

The individual device drivers are described in detail in the section on AMOS Input/Output.

6.3 AMOS Input/Output

The AMOS I/O system provides the application programs with a unified interface to peripheral devices.

The AMOS I/O system is compatible with the DAMOS I/O system.

The AMOS I/O system calls directly to a peripheral device driver process except for disks and diskettes where the I/O system calls the AMOS file management system. In the former case the device is treated as a single file in itself, in the latter case the file management system will structure the device into logical files organized in a hierarchical directory structure.

The AMOS I/O system is a set of procedures which may be called from all application programs. The I/O system code is shared by all programs and invoked by MON-instructions. However, no context switch takes place. The I/O system checks the validity of parameters passed to it, and the legality of a requested operation. If the check is satisfied a message is prepared for the appropriate device driver and sent to it. User identification code is specified to the driver, enabling it to perform any relevant authentication.

The commands to the I/O system are divided into four groups:

- Environment Control
- Direct Input/Output
- Sequential Input/Output
- Input/Output Utilities

The first group, which to a certain extent is device dependent, covers such operations as:

create file, assign terminal.

The direct I/O operations directly reflect the data transfer commands of peripheral device drivers and of the file management system. They may be performed either with or without suspension of the calling process. It is the

responsibility of the user to allocate and specify to the I/O system the necessary buffers. The buffers used for one command must be specified to the I/O system by a list of buffer references. Each of these references may specify either a local buffer or an external buffer requested from the buffer manager. Local and external buffers may be mixed freely in a buffer list.

The sequential I/O procedures are implemented by the I/O system on top of the direct I/O procedures. For the purpose of sequential I/O, the I/O system will allocate the necessary buffers for implementation of a read-ahead/write-behind strategy. The I/O system copies data between the user area and the buffers. Single bytes or records of any length may be transferred. Blocking/Deblocking of records are handled by the I/O system.

The input/output utility procedures are built on top of the sequential I/O procedures. These procedures include:

connect
: connects a stream to a file, mode must be specified as input or output.

disconnect
: disconnects a stream from a file

get_position
: returns the current position of the stream

set_position
: the stream is adjusted so that the next byte will have the specified position on the file to which the stream is connected

inbyte
: the next byte from the stream is delivered

backspace
: next byte to be delivered will be the same as the last delivered

inrec

delivers a record with a specified number of bytes

inline

delivers a record terminated by a NewLine character (Linefeed)

intype

delivers a character and its type (error, space, digit, letter other)

inelement

delivers a number, an identifier or a special character

outbyte

outputs a byte on the stream

outnl

outputs a NewLine character on the stream

outrec

outputs a record with a specified number of bytes on the stream

outline

outputs a record with a specified number of bytes and a New Line character on the stream

outtextp

a text string embedded in the program part is output on the stream

outtextb

a text string embedded in the data part is output on the stream

outhexa

a word is output on the stream as four hexadecimal digits and preceeded by a specified character

outinteger

a word is output on the stream as decimal digits. The number may output as

signed

unsigned

the minimum number of digits output may be specified.

Leading zeroes may be suppressed.

outlonginteger

a double word is output on the stream. Format control is similar to outinteger

flush

currently buffered data are output to the file connected to the stream

6.3.1 File Management System

The File Management System (FMS) is responsible for storing, maintaining, and retrieving information on secondary storage devices (disks).

The number and kind of devices attached to the FMS is dynamically reconfigurable.

The FMS internally consists of a set of coroutines. The user interface is handled by a coroutine which distributes the commands to internal coroutines. These coroutines are of different kinds; one kind for each access method implemented. The internal coroutines all access the disks through a disk cache coroutine. This coroutine manages a pool of sector buffers according to a priority scheme, and accesses the physical disks through device handlers only when necessary.

Device and Volume Handling

Each device known to the FMS is represented by a Device Control Block (DCB). A DCB contains the information which makes it possible for the file system to use the device. This information includes the identification of the command synchronization element of the corresponding device handler, and the kind of device.

When a volume is mounted on the device, its description is included in the DCB. Both devices and volumes may be referenced by symbolic names. The device control block is linked to all open files on the volume mounted on that device.

The file system may be given commands concerning:

- Management of peripheral devices.
  Devices may be assigned to and deassigned from the file system dynamically. Instances of device handlers are at the same time created or deleted.

- Management of volumes.
  Volumes may be mounted on and dismounted from specific devices.

User Handling

Each process using the backing store file management system is within the file system represented by a User Control Block (UCB). The User Group Identification (UGI) is contained in the UCB. Several processes may be active under the same UGI, thus having the same access rights. The UCB is linked to the file control blocks for files which are open to the user.

The file management system may be given commands concerning:

- Creation and Removal of user control blocks.

File Types

The file system supports two different organizations of files on disk. A contiguous file consists of a sequence of consecutive sectors on the disk. The size of a contiguous file is fixed at the time the file is created and cannot be extended later on. A random file consists of a chain of indices giving the addresses of areas scattered on the volume. Each area consists of a number of consecutive sectors. The number of sectors per area is determined at creation time, whereas the number of areas may increase during the lifetime of the file.

Directories

The file system uses directories to implement symbolic naming of files. If a file has been entered into a directory under a name specified by the user, it is possible to locate and use it later on. Temporary files do not need to be named. A file may be entered into several directories, perhaps under different names. Since a directory is also considered a file, it can itself be given a name and entered into another directory. This process may continue to any depth, thus enabling a hierarchical structure of file names.

File Handling

Each open file is within the file system represented by a File Control Block (FCB). The FCB contains information which makes it possible to access the corresponding file. This information includes the address of the file on the volume, its size and its type. The FCB is linked to user control blocks for all users who operate on the file indicating at the same time their access rights to the file.

File Commands

The commands given to the file system concerning files may be grouped as:

- Creation and removal of files.
  A user may request that a file is created with a given set of attributes and put on a named volume.

- Naming of files in directories.
  A file may be entered into a directory under a symbolic name. Using that name it is possible to locate the file later on. The file may also be renamed or removed from the directory again.

- Change of access rights for a specific user group (or the public) vis a vis a file. The right to change the access rights is itself delegatable.

Redundant Disks

The FMS allows use of redundant disk packs, which are updated concurrently to assure that data will not be lost in case of a hard error on one disk.

The FMS allows exclusion of one of the two identical volumes, while normal service goes on on the other one.

Bad Sectors

The FMS is able to use a disk pack with bad sectors, unless it is sector 0.

The bad sectors are handled by keeping a translation table on each volume from each bad sector to an alternative sector.

Security

The protection of data entrusted to the file management system is handled at the file level.

The mechanism for access control is based on the use of Access Control Lists (ACL). There is an ACL connected to each file. The ACL is a table which describes the access rights of each individual user (one being the public) to the

corresponding file. Whenever a user tries to access a file, the ACL is used to verify that the user is indeed allowed to perform this access.

Access Methods

The file management system implements two access methods to files:

Unstructured Access

For access purposes a file is considered a consecutive string of bytes. It is, therefore, a byte string which is transferred between a file and a user buffer. The user can directly access any substring in a file.

The commands which are implemented by this access methods are:

READBYTES - Read a specified byte string

MODIFYBYTES - Change a specified byte string

APPENDBYTES - Append a byte string to the end of the file.

Index Sequential Access

CRAM is a multi-level-index indexed sequential files system. It features random or sequential (forward or reverse) access to records of 0 to n bytes, n depending on the selected block size, on base of keys of 0-126 bytes. The collating sequence uses the binary value of the bytes so e.g. character strings are sorted alphabetically.

CRAM supports variable key sizes as well as variable record size within the same file (or subfile, see below).

CRAM works on normal contiguous FMS files which are initialized for CRAM use by means of a special CRAM operation.

The CRAM updating philosophy is based on the execution of a batch of related updatings, which all together form a consistent status change of the CRAM file, being physically updated as a single update by means of a LOCK operation. That is, after such a batch of updates, all these updates may either be forgotten (by means of the FORGET operation) or locked (by means of the LOCK operation). Both operations are performed without critical regions, i.e. without periods of CRAM data base inconsistency.

For convenience, CRAM supports subdivision of the CRAM file in up to 255 subfiles, each identified by a subfile identifier of 0-126 byte (as a key).

CRAM keeps track of the different versions of the CRAM data base by means of a 32 bit version number, which is incremented every time CRAMNEWLOCK (the locking operation) is called. This version number can only be changed by CRAMNEWLOCK (and CRAMINIT), but if the user intends to use it for some sort of unique update version stamping, it is delivered by the operations CRAMNEWOPEN,CRAMNEWLOCK, CRAMFORGETand CRAMNEWVERSION.

The file access commands supported by CRAM are:

CRAMINIT

Initializes a contiguous FMS file (which must be FMS open) to use as CRAM data base.

CRAMNEWOPEN

Opens an open FMS file for CRAM use. The files must previously have been initialized by CRAMINIT

CRAMREAD

Reads a CRAM record on base of a key and a direction, which must be one of: LT, LE, EQ, GE, GT. The direction gives the relation between the searched record compared to the given key.

CRAMUPDATE

Updates an already created CRAM record. Neither key size, key contents nor record size can be changed, only the record contents.

DRAMDELETE

Deletes the record with the given key from the data base.

CRAMPURGE

Deletes a span of records between two given keys from the data base.

CRAMCREATE

Creates a new CRAM record with a given key (and Key size) and a given record content (and record size).

CRAMCREATESUBFILE

Introduces a new subfile with a given name to the CRAM data base. The new subfile is left closed and empty. The subfile entry includes a user defined file option record with format like a normal CRAM record.

CRAMDELETESUBFILE

Deletes a subfile from the data base. The subfile must be empty, i.e. without records.

CRAMOPENSUBFILE

Opens the subfile for CRAM operations and delivers the option record to the user (see CRAMCREATESUBFILE). A direction must be specified as for CRAMREAD.

CRAMLOOKUPSUBFILE

Gets the option record of a subfile (as CRAMOPENSUBFILE), but does not open the subfile.

CRAMFORGET

Forgets a batch of CRAM updates.

CRAMBEGINLOCK

Reserved for future extension with queueing facility of update requesting processes. Presently working as CRAMFORGET.

CRAMNEWVERSION

Delivers the present version of the CRAM data base as a LONG INTEGER (32 bits).

6.3.2 Magnetic Tape File Driver

The Magnetic Tape driver is responsible for storing and retrieving information on magnetic tapes. It is able to handle one magnetic tape controller with a maximum of 8 tape transports in daisy-chain.
The driver is logically split into 3 ports:

- I/O-SYSTEM interface
- Main Processing
- Magnetic tape controller interface

Commands for the driver are received by the I/O-SYSTEM interface while the controller interface implements a number of (low level) commands for handling a tape transport.

Symbolic volume names and file names are implemented through use of label records.

The functions of the driver can be separated into four groups:

- Device functions
- Volume functions
- File functions
- Record functions

Device functions

The following functions are defined:

- Assign a given name to a given unit of the controller.
- Deassign a given device.

Volume functions

- Initiate the tape on a given device assigning a name to it by writing a volume header label.

- Mount a given volume on a given device.
- Dismount a given volume.
- Rewind a given volume.

File functions

- Create a file on a given volume. The following information must be supplied by the caller and will be written onto the tape in file header label records:
  1. File name
  2. Fixed/variable length record specification
  3. Record size.

  The file is opened for output and the given volume is reserved for the caller.

- Find a file with a given name on a given volume. The file is opened for input and the given volume is reserved.

- Skip a given number of files (backwards or forwards) on a given volume. The file at the resulting tape position is opened for input and the volume is reserved.

- Get information about the currently open file on a given volume. Information like file sequence number, record size and type (fixed/variable length) can be retrieved.

- Close currently open file or a given volume. Volume reservation is released.

Record functions

- Skip a given number of records (forwards or backwards) in a given file.
- Read a record in a given file.

- Write a record in a given file. The driver performs recovery from writing errors by
  1. backspacing over the record in error
  2. erasing a fixed length of about 3.7 inches (thus increasing the record gap).
  3. attempting the writing once more.

  This procedure will be repeated a maximum of 10 times.

6.3.3 TDX Host Interface Driver

This section gives an overview and describes briefly the main objects of the TDX system and the main functions supported by the TDX driver.

Objects in TDX Net

The main object in a TDX system is the TDX bus as described in Chapter 8. A large number of devices and equipment may be interconnected via the TDX bus. Such an interconnection (establishing a logical line between two points within the TDX system) is called a CHANNEL or a LOG-LINE.

An object directly connected to the TDX bus is called a TDX DEVICE.

The TDX system handles these four types of TDX devices:

- CONTROLLER
- HOST I/F (STI)
- LTUX I
- LTUX II

The TDX CONTROLLER is the TDX bus manager allocating bandwidth/time slices to connected TDX devices.

The TDX HOST I/F (STI) is a device interfacing a host computer (e.g. CR80) to the TDX bus. From the TDX system point of view, a host is a device (demanding large bandwidth/many time slices) communicating with a large number of TDX devices (usuallly demanding small bandwidths/few time slices).

The LTUX is a line terminating unit for the TDX system, interfacing a large number of device types to the TDX system.

The LTUX is the standard interface between input/output applications (communication lines, terminals, process monitoring/control etc.) and the TDX bus. The LTUX device provides up to 16 full-duplex individual channels of up to 9.6Kbits. The sum of bandwidth assigned to the channels through the LTUX is dynamically changeable by request to the TDX controller.

Functions Supported by TDX Driver

The interconnections supported by the TDX system may be created, changed and removed dynamically.

Though the low level protocol of all interconnections is the same (using the TDX frame formats), different higher level protocols may be implemented on the LOG-LINE level.

As a consequence the TDX driver handles the following basic functions:

- LINE CONTROL (creation and removal of LOG-LINES). On request from an application process the TDX driver will create a LOG-LINE with the specified attributes (address, protocol etc.) or remove a specified LOG-LINE.

- PROTOCOL ADAPTION
  Adapting a LOG-LINE protocol to the I/O system used by the application process. This may include assembly disssembly of data units, data conversion, etc.

- DATA TRANSFER

  Transferring data on LOG-LINE basis. LOG-LINES may be created for "usual" data transfer or for transferring control information (forming a dedicated control channel). The TDX driver is transparent with regard to data transfers, and the two types of LOG-LINES are handled exactly the same way.

6.3.4 LTU Driver

The LTU driver interfaces the LTU system to the AMOS I/O system. The driver makes a number of different LTU devices (possibly running a number of different protocols) conform to the "file" concepts of AMOS.

The LTU driver is a single CR80 process including a number of coroutines synchronized with:

- application processes (via AMOS I/O system)
- LTU-CR80 interface (via shared RAM)
- LTU-connections (via internal queues)

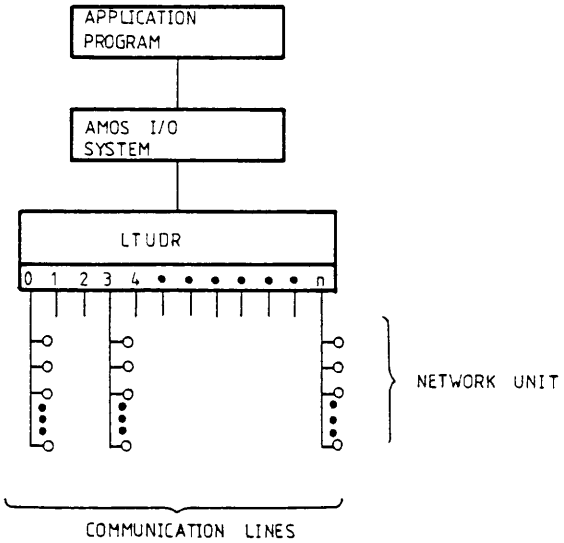
The coroutines synchronized with LTU connections are of different types according to different protocols, e.g. I400, X25 packet level.

The LTU driver interfaces an application program with a terminal, a virtual circuit or another addressable unit which is logically accessible via a communication line terminated by an LTU. Such a unit is called a network unit in the following discussion. The connection between the application program and a network unit is called a connection.

The application program must call the LTU driver through the AMOS I/O system.

The LTU driver handles the following basic function:

- Connections Control
  (creation and removal of CONs). On request from an application process the LTU driver will create a connection with the specified attributes or remove a specified CON.


APPLICATION PROGRAM
AMOS I/O SYSTEM
LTUDR
0 1 2 3 4 • • • • • • n
NETWORK UNIT
COMMUNICATION LINES

- Protocol Adaption

  Adapting a connection protocol to the I/O system used by the application process. This may include assembly, disassembly of data units, data conversion etc.

- Data Transfer

  Transferring data on a connection basis. Connections may be created for "usual" data transfer or for transferring control information. The LTUDR is transparent with regard to data transfer and the two types of connections are handled exactly the same way.

- LTU Boot Loading

  The LTU is included in the domain of the LTU driver by using the assign_LTU command. The create_boot_file command creates the boot file, and the I/O procedure "append_bytes" is used to boot load the specified LTU. Reception of the dismantle_boot_file command indicates that the boot loading has finished and that the LTU is ready.

The commands for control and data transfer are the standard I/O procedures:

assign
deassign
create
dismantle
enter
get root
look up
descent
read bytes
append bytes
modify bytes

Commands

Assign LTU

This procedure includes an LTU in the domain of the LTUDR. The procedure is called with a set of parameters, describing a physical LTU.

Deassign LTU

This procedure excludes a LTU specified by its logical number.

Deassign Line

This procedure excludes a line specified by its logical number.

Initiate Boot Loading

The boot loading of an LTU is initiated by calling the create_boot_file procedure.

This procedure is called with a set of parameters, describing a boot file for boot loading the LTU.

The procedure returns a boot file identification to be used when referring to this particular boot file.

Terminate Boot Loading

The boot loading is terminated by calling the dismantle_boot_file procedure.

This procedure makes the LTU driver remove a boot file specified by its "BOOT_FILE_id".

Create Connection

This procedure is called with a set of parameters, describing a connection (CON) for data transfer.

The procedure returns a connection identification to be used when referring to this particular connection.

Dismantle Connection

This procedure makes the LTU driver delete a connection specified by its "CON_id".

Naming Procedures

The naming procedures allow users to manipulate connections identified by symbolic names.

This facility is especially important for AMOS standard utility programs which locate files by means of symbolic file names.

The standard AMOS I/O procedures for looking up files

GETROOT
LOOKUP
DESCENT

are therefore supported by the LTU driver.

Connections may be given symbolic names by the standard AMOS I/O procedure

ENTER

The getroot procedure returns a dummy file description which may be used as a symbolic file directory needed in LOOKUP, DESCENT and ENTER.

This file descriptor implicitly identifies the LTU driver when LOOKUP, DESCENT and ENTER are called.

Data Transfer

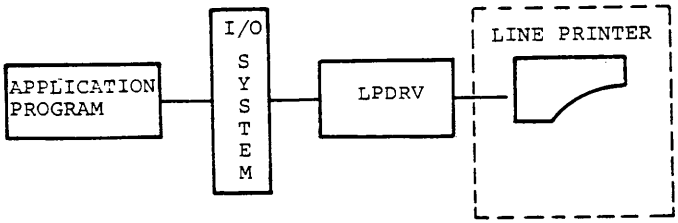
Data transfers are performed by using the standard I/O procedures for direct I/O.

6.3.5 Line Printer Driver

The Line Printer Driver (LPDRV) is the device driver for interfacing an application program to a line printer.

The application program must call the LPDRV through the AMOS I/O SYSTEM.

The AMOS I/O SYSTEM shields the application program from physical device characteristics, and makes the LPDRV look like a disk file to the application program.



I/O commands sent from application processes are inspected in the order they are received.

The output commands:

appendbytes
modifybytes

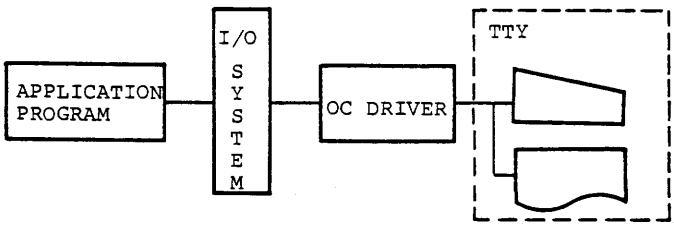
are handled by printing the requested number of bytes on the line printer.

6.3.6 OC Driver

The OC driver (TTYDRV) is the device driver for interfacing an application program to a physical teletype terminal, (hardcopy or screen type) connected via the serial V24 channel of the SCM.

The application program must call the OC driver through the AMOS I/O system (IOS).

The IOS shields the application program from physical device characteristics, and makes the OC driver look like a disk file to the application program.



Interface to Application Programs

The OC driver looks like a file management system, where all devices, volumes and files are mapped onto a single terminal.

When an input command is processed, data entered from the terminal keyboard are delivered to the application process.

When an output command is processed, the associated data are typed on the terminal (displayed on the screen).

Processing of I/O Commands

I/O requests sent from application processes are inspected in the order they are received. All commands but input commands are also processed in the order they are received.

The output commands:

appendbytes
modifybytes

are processed by typing the associated data on the terminal irrespective of any file positioning information associated with the commands.

The input command:

readbytes

is processed by returning the requested amount of characters or fewer, when input is entered from the terminal keyboard for the process in question.

Fewer characters than requested will be returned if the keyboard input is terminated by a carriage_return (13) or if the first character entered is an end_of_medium (25) character.

Interface to Terminal

The terminal handled by a TTYDRV process may be shared by many application processes.

In order to be able to direct input to a specific process and to identify the process which is outputting information, a simple protocol between the TTYDRV and the operator at the terminal is used.

The process connected to the terminal (outputting to it or receiving input from it) will always be identified by either the TTYDRV typing:

FM: <process name>

when it is going to output data from the process identified by <process name>, or by the operator using the break (interrupt) key and entering the <process name> for the process to receive his input. When the break key is used, the TTYDRV will suspend its current activity and type:

TO:

The TTYDRV now waits until the operator has entered a <process name> terminated by a carriage return.

The TTYDRV looks up the process corresponding to <process name>. If no such process exists a message

UNKNOWN

is typed, and the TTYDRV resumes its possibly suspended activity.

If the process exists, the TTYDRV sends a signal to the process, waits 1 second and then inspects its (possibly queued) input commands. If an input command is received from the identified process, the operator may now enter input which will be sent to the process. If no input command from the process is received, the TTYDRV types:

BUSY

and resumes its possibly suspended activity.

All input and output takes place between the terminal and one specific process until a new process is defined by either:

FM: <process name> or
TO: <process name>

When the currently connected process is not requesting input (no input command received), any entered input will be buffered by the TTYDRV (max. 50 characters) but not echoed. When an input command is received from the connected process, the TTYDRV delivers to it the contents of its buffers and while doing so, types the contents of the buffer.

Presentation

The OC driver provides the following presentation functions:

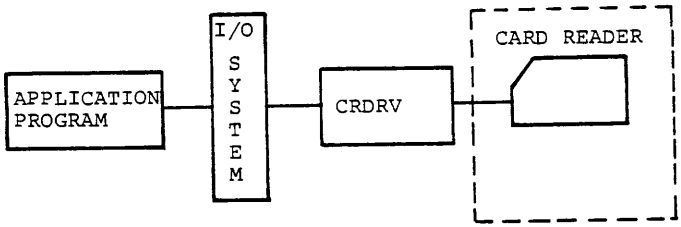
- folding of long lines
- tabulation control
- tabulation
- line editing

6.3.7 Card Reader Driver

The Card Reader driver (CRDRV) is the device driver for interfacing an application program to a card reader.

The application program must call the CRDRV through the AMOS I/O SYSTEM.

The AMOS I/O SYSTEM shields the application program from physical device characteristics, and makes the CRDRV look like a disk file to the application program.



I/O commands sent from application processes are inspected in the order they are received.

Input commands are handled by reading input from the card reader converting to ASCII and returning the requested amount of bytes or fewer if an error occurs or if an end-of-file-card is read.

An end-of-file-card (/ in columns 1 and 2 and otherwise blank) causes the CRDRV to return an end of file completion code.

The CRDRV skips blanks after the last non-blank character on a card. A newline character (value 1Ø) is inserted between cards.

6.4 SYSTEM INITIALIZATION

When the unmapped CR80 is mastercleared a boot strap loader is given control. The boot strap loader is contained in a Programmed Read-Only Memory which is part of the SCM.

The boot strap loader accepts a load command from the SCM serial V24 channel. The load commands must define a disk and a boot load module on the disk. This load module is fetched by the boot strap loader, placed in memory and given control.

The load module must contain as a minimum:

- the AMOS Kernel
- the AMOS Root process
- one or more AMOS idle processes
- the AMOS RTC process.

After initialization of the Kernel, the Root process starts initializing the other processes included in the load module.

6.5 OPERATING SYSTEM FOR SOFTWARE DEVELOPMENT

TOS is an operating system which supports interactive terminal users in a program development environment.

For a detailed description of TOS refer to the previous Chapter 4, DAMOS, section 4.8.

6.6 PROGRAMMING LANGUAGES

The following programming languages are supported under AMOS besides assembler:

- SWELL, which is an intermediate level systems programming language.
- PASCAL, which is a general purpose high level programming language.
- COBOL, which is a high level language for administrative application programming.

For a general description of these languages refer to the previous Chapter 4, DAMOS, section 4.9.

6.7 UTILITIES

The AMOS support software library contains a variety of on line and off line tools. The library includes:

- language processors
- text preparation programs
- program development and test tools
- file manipulation programs
- directory maintenance utilities
- system maintenance utilities
- hardware test and diagnostic programs.

For each of the programs a short description of its function is given, the way it is called (parameters) and a reference to the user's manual describing the program.

As many of the programs are functionally identical to corresponding DAMOS utilities, some of the descriptions are given by reference to the section on DAMOS utilities. For a general introduction to the descriptions refer to the previous Chapter 4, DAMOS, section 4.10.

6.7.1 LANGUAGE PROCESSORS

The AMOS language processors include:

- Micro assembler (refer to 4.10.2.1)
- Assembler (refer to 4.10.2.2)
- SWELL Compiler (refer to 4.10.2.3)
- PASCAL Compiler (refer to 4.10.2.4)
- COBOL Compiler (refer to 4.10.2.5)
- PASCAL Cross references (refer to 4.10.2.6)
- Assembler Cross references (refer to 4.10.2.7)
- Linkage editor (refer to 4.10.2.8)
- PASCAL pretty printer (refer to 4.10.2.9)
- SWELL pretty printer (refer to 4.10.2.10)
- Parsing System (refer to 4.10.2.11)

6.7.2 TEXT PROCESSORS

The AMOS text processors include:

- an Interactive Editor (refer to 4.10.3.1)
- a Batch Editor (refer to 4.10.3.2)
- a general Text Formatter (refer to 4.10.3.3)
- a File Merge Program (refer to 4.10.3.4)
- a File Concatenator (refer to 4.10.3.5)
- a Translate program for converting between capital letters and minuscules. (refer to 4.10.3.7).

6.7.3 TEST TOOLS

The AMOS test tools include:

- a Memory Save program
- a Patch program (refer to 4.10.4.1)
- a Disassembler (refer to 4.10.4.2)
- a System Generator
- an On Line Test Output facility (refer to 4.10.4.3)
- a Boot File Generator

6.7.3.1 Memory Save program

Invocation syntax:

SAVE <output file> <memory>

	Defaults	Use:
<output file> ::= O : <file id>	-	(output)
<memory> ::= F : <integer>	-	(start address)
, <integer>	0	(memory.section)
L: <integer>	-	(top address)

The program saves memory contents into the output file, which must exist before program invocation.

For further information: CSS/125/USM/0031

6.7.3.2 System Generator

Invocation syntax:

SYSGEN {<user params>}$^{2}_{2}$

	Defaults	Use:
<user params> ::= O : <file id>	-	(output)
L : <file id>	-	(log file)

The system generator is an interactive program which reads its commands from current input and displays messages on current output. The program creates a system load module from a number of individual program modules. The system module is output to the output file, which must be specified and exist before program invocation

For further information: CSS/121/USM/0023

6.7.3.3 Boot File Generator

ACTIVATION:

BOOT I: <load module file id> O: <boot module file id>
P: <page number> F: <first load address>
B: <base of process loaded>

FUNCTION:

A boot file, (as understood by the AMOS Boot Strap Loader), is generated in the file specified by the <boot module file id>. Its contents are a header and a load module.

Input and output file names must be given.
The default values of P/F/B are zero.

I,O can be entered in any order.
P,F,B can be entered in any order.

6.7.4 FILE MANIPULATION PROGRAMS

The AMOS file manipulation programs include:

- a File Copy program (refer to 4.10.5.1)
- a Directory Copy program (refer to 4.10.5.2)
- a File Display program (refer to 4.10.5.3)
- a File Compare program (refer to 4.10.5.7)
- a Line Printer Support program (refer to 4.10.5.5)
- a Card Reader Support program
- File Conversion programs (refer to 4.10.5.6)
- a File Format Conversion program
- a File Archive Maintenance program.

6.7.4.1 Card Reader Support Program

Invocation syntax:

CARD

The program copies a card file from the card reader to a DAMOS file.

The first card in a card file is a command card specifying the corresponding DAMOS file.

After the command card a number of data cards may follow ending with a special EOF card.

An EOF card has a "/" in columns 1 and 2 and blanks in all other columns.

Several Card files may be processed just be placing the card file one after another in the card deck.

The program terminates when an extra EOF card is encountered after a card file.

During the program execution, the program outputs the identifier and size (number of data cards) of the currently processed card file.

When all the card files have been processed, the program outputs the number of cards in the card deck.

The command card must contain:

COPYTO: <file id>

For further information: CSS/125/USM/0043.

6.7.4.2 File Format Conversion Program

Invocation syntax:

CONVSYS

The program CONVSYS can create, remove, rename, and list files, which are stored in the format used by the CR80 System One OS, while running under CR80 AMOS. It also allows these files to be converted to AMOS format.

From current input the program accepts the following commands:
BOOT <filename> <first> <base> <page>
CREATE <filename> <areasize> <blocksize>
DEMOUNT
INIT <no. of entries>
LIST
MOUNT <file system name> <device name>
QUIT
READ <filename> <file id>
RENAME <old filename> <new filename>
WRITE <filename> <file id>

The READ command copies a System One file into an AMOS file. The WRITE command performs the opposite function.

Prompts and messages are written on current output.

For further information: CSS/160/USM/0029.

6.7.4.3 File Archive Maintenance Utility

Invocation syntax:

ARCHIVE

The program ARCHIVE provides all functions necessary for manipulating files on magnetic tape and for mounting and dismounting of volumes. The main purpose is to copy all members of a directory to/from a magnetic tape, e.g. for back up purposes.

From current input the program accepts the following commands:

CONVERT	<conversion mode>
DISMOUNT	
INIT	<volume id>
LIST	
MOUNT	<volume id> <unit no.>
QUIT	
READ	<filename> <file id> <file selector>
READALL	<file id> <selector>
WRITE	<filename> , <file id>
WRITEALL	<file id>

Prompts and message are written on current output.

For further information: CSS/161/USM/0046

6.7.5 DIRECTORY MAINTENANCE UTILITIES

The AMOS directory maintenance utility programs are a suite of programs for inspecting and modifying the contents of backing store file structures. Programs for the following directory operations are available:

- create a file (refer to 4.10.6.1)
- delete a file (refer to 4.10.6.2)
- rename a file (refer to 4.10.6.3)
- enter an existing file into a directory (refer to 4.10.6.4)
- list the contents of a directory (refer to 4.10.6.5 and 4.10.6.6)
- list the attributes of a file (refer to 4.10.6.7 and 4.10.6.8)
- change the protection of a file (refer to 4.10.6.9)

6.7.6 SYSTEM MAINTENANCE UTILITIES

The AMOS system maintenance utilities include:

- a Disk Initialization program (refer to 4.10.7.1)
- a Disk Salvation program (refer to 4.10.7.2)
- a Print Queue Rectification program (refer to 4.10.7.3)
- a program for reading the time
- a program for setting the time
- a performance monitor program

6.7.6.1 Read Time Program

Invocation syntax:

TIME

The program displays the system date and time on current output.

The format is:

<year> : <month> : <day> <hour> : <min> : <sec>

6.7.6.2 Set Time Program

Invocation syntax:

SET_TIME <date> <time>

			Defaults	Use:
date	::=	<integer>	-	(year-1980)
	:	<integer>	-	(month)
	:	<integer>	-	(day)
time	::=	<integer>	-	(hour)
	:	<integer>	-	(minute)
	:	<integer>	-	(second)

6.7.6.3 Performance Monitor

Invocation syntax:

PERFORMANCE

The program PERFORMANCE provides facilities for monitoring the CPU usage imposed by a single process (task). The number of CPU milliseconds used can be logged on current output on a cyclic basis.
PERFORMANCE can also monitor the performance of a specific CPU module in terms of the load distribution on software priorities. Finally, PERFORMANCE can modify (tune) the following CPU parameters: size of timeslices for each software priority, hardware priority, and schedule count.

The program accepts the following commands from current input:

P	for	monitoring of a process
C	for	monitoring of a CPU
T	for	tuning of a CPU
Q	for	terminating

Additional parameters needed by the commands are prompted.
The P or C monitoring mode may be terminated by pressing the break key and typing the name of the process which is executing the PERFORMANCE program.

6.7.7 DIAGNOSTIC PROGRAMS

For a complete description of all diagnostic programs refer to an earlier section under chapter 4, DAMOS, section 4.10.8.

6.8 CR801 CROPS

6.8.1 INTRODUCTION

CROPS provides control functions for terminal based data base management systems. The CROPS system architecture is illustrated in figure 6.8.1 overleaf. CROPS operates as a modular subsystem in conjunction with the AMOS operating system kernel, providing scheduling mechanisms for SWELL and COBOL applications.

User terminals may be multidropped on the communication lines connected at the LTU interface. On each line, 4 terminals may be attached, thus a single LTU module may handle 16 terminals. The GT400 intelligent VDU terminal is used for CROPS applications.

CROPS is designed for secure operation. Users are allowed access to the system only by entering appropriate password at log on. Restrictions on the use of individual transaction types are implemented on a per user basis. Data integrity is provided to the level of completed transactions, and the applications data bases are fully recoverable on the basis of back-up and transaction logs.

Transactions, each consisting of 1 to 20 VDU pictures, may be submitted to the system in either direct- or indirect mode, the difference being that direct transactions are processed immediately, while indirect transactions are entered on a batch queue for processing when resources are available.

The operation of CROPS is best illustrated by summarizing the events occuring during the processing cycle of a transaction:

CHRISTIAN ROVSING A/S

CR 801/CROPS

CROPS System Architecture

Scale		Approved	
Date	81-01-15	E	
Drawn	JSL/MZL		
Parts no		30-1165	
Print no		Sheet 1 of 1 sheets	

Issue	1
Date	81-01-15
Approved	JSL

On request an empty transaction form is displayed on the user terminal.

The form is filled out interactively. The transaction may consist of several VDU pictures. A wide range of key-stroke commands for handling the pictures are available, e.g. get-next, get-prev, get-first, get-last. During this interactive phase the user communicates with the CROPS VTE.

The completed transaction is transmitted to the CROPS system kernel by the SEND-DIRECT or SEND-INDIRECT function key. If the SEND-DIRECT is applied the keyboard is automatically locked until the response transaction is returned; otherwise, in then SEND-INDIRECT case, the transaction is queued in the indirect batch queue associated with the terminal.

The appropriate application is scheduled and the transaction, if it is updating, entered in the transaction log file.

The application performs the transaction processing, thus making data base updates and spooled print entries as required. Normally, the application eventually generates a response transaction to the user. In case of user input errors the original input transaction with erronous fields inverted is returned to the VDU and the first picture displayed.

At this point the cycle is complete and CROPS is ready to accept the next user input transaction.

6.8.2 DESCRIPTION OF MODULES

The CROPS Command Interpreter (COS) performs system initialization/adaptation functions. During operation, COS processes system operator commands and further handles fault conditions reported by other CROPS modules.

The Virtual Terminal (VTE) is structured for interfacing a generalized VDU type data terminal to the CROPS system kernel. The VTE performs data conversion, i.e. mapping and formatting of picture data and assembly/disassembly of complete transactions into pictures. Only the mapping subassembly of the VTE is terminal type specific.

The LTU driver, in conjunction with the special purpose GT400 protocol firmware in the LTU module, provides the line side interface to the VTE.

The Scheduler controls the execution of applications to process incoming transactions. The scheduling algorithm is based on a "Smallest-Job-First" strategy.

The Application Interface (AIF) provides I/O-system like interface functions for the applications. Through the AIF, the applications may perform CRAM data base access (I/O), Transaction access (TIO) and produce spooled print on the PRAM print queues (PIO)

The FMS System incorporated in the CR801 CROPS system includes the CRAM index sequential access method and the CROPS Transaction Queue Monitor (TQM) as integral parts. TQM administers the CROPS session transaction queues, thus providing a transaction/picture I/O service for the VTE and the scheduler. The design of TQM allows for transaction processing without any critical regions, which means that system failure will not cause inconsistency of data base and transaction files.

The Library/loader system provides an object retrieval capability applied by the VTE and scheduler for fast loading of applications, transactions and

picture objects from a central disk library. The library is generated off line by a special Library Generator utility.

The Applications may be either SWELL-coded or COBOL coded. COBOL applications are executed based on interpretation by the COBOL Routine System. The only applications having a special status in the CROPS System are the LOGON application and the operators Communication application (OPCOM), which is a general application handling system related to operator transactions.

In the following section a short presentation of some typical CROPS applications is given.

6.8.3 CR801 Standard Application Software

The following application software packages are available under the name CRMINI

- Sales order entry and control
- Invoicing
- Accounts receivable
- Accounts payable
- Stock control
- Sales statistics
- General ledger.
- Purchase order control
- Production order control
- Material requirement planning
- Standard product costing
- Wage and salary sytem
- Budget planning
- Word processing
- Report generator and inquiry system

CRMINI PACKAGE

The CRMINI package is specially designed for commercial applications in small and medium size trading companies, wholesalers or distributors. By use of the extensive CR801 communication software, the CRMINI package could also be used in a distributed system for larger companies and organizations.

All program routines are developed in COBOL to on-line operation on the CR801 minicomputer system. Thus the operational advantages obtained are:

- Directy decentralized input of data on visual displays with on-line verification of all input information.

- Inquiries on visual displays about products, customers and cost accounts with up-to-date information.

- Easy installation and start-up as the user is able to operate the system after a few hours' instruction.

- Local control of operation so that the ultimate user can request and get his information whenever the need arises.

- In designing CRMINI much effort has been applied to develop a system which is easy to operate and which has the necessary security and control facilities.

CRMINI is a standard system built in modules that make it possible to adjust to the individual user. Because of its flexible structure, it is normally not necessary to change number systems or to make non-appropriate changes of business procedures.

ORDER ENTRY AND INVOICING

Fast, effective and reliable order entry and invoicing is an important requirement in order to satisfy customers and necessary for an efficient control of accounts receivable and inventory.

The CRMINI routines for order entry and invoicing provide opportunity for:

- Order entry based on telephone calls or order notes.
- Check on customer number and credit conditions
- Check on product number, products in stock, and unit prices
- Automatic or manual price setting with quantity-, line-and/or invoice discount.
- Real time updating of customer and stock.

The invoicing system can be used for pre-invoicing with automatic back order registration or ordinary invoicing based on actual shipment. It is also possible to print a packinglist, make corrections and print the invoice, corresponding to the actual shipment, in time for the invoice to follow the shipment.

In addition, the system offers the possibility of order entry with later delivery, invoicing of back orders, invoicing of special items not in stock, and issue of credit notes. On the visual displays, it is possible to make inquiries on products in stock, back orders by product or by customer, delivery times, units, etc.

ACCOUNTS RECEIVABLE/ACCOUNTS PAYABLE

An increasing number of customers with individual payment conditions gives rise to problems in most accounting departments as to updating, interest calculation, reminders and the general control of liquidity.

The CRMINI modules to accounts recievable/accounts payable offer the possibility of the balance forward principle or the open-item principle, where every payment should refer to an invoice. It is also possible to work with different currencies, which might be of interest to export/import companies.

STOCKCONTROL

Real time stock accounting is necessary for an effective inventory control.

CRMINI automatically offers a daily warning when stock is below the re-ordering level. A re-order list can be printed for the individual supplier with his identifications, to facilitate purchasing decisions.

Automatic control of changes in cost price, fast determination of inventory worth, and inventory list by location order are some of the other important advantages.

SALES STATISTICS

Information on turnover, profit per product, product group or customer is necessary to ensure the profitability of a company.

CRMINI offers information directly on visual displays about sales and cost-of-sales per product, product group or customer, by month and year, for correlation with the same period of the previous year or the last 12 months. This information is also available by geografical district, customer-type or salesman.

GENERAL LEDGER

The CRMINI general ledger offers features not often found in standard general ledger systems.

The following advantages are highlighted:

- On-line inquiry from visual display to separate accounts and postings.
- Account number system by own choice. Every report is based on individual criteria independent of the account number system.
- Data entry in different periods is possible with real time updating.
- Reference to budget and previous year is possible.
- Automatic balancing of all entries.

6.9 CR80 Software Documentation

6.9.1 CR80 Family Software Documentation Hierarchy

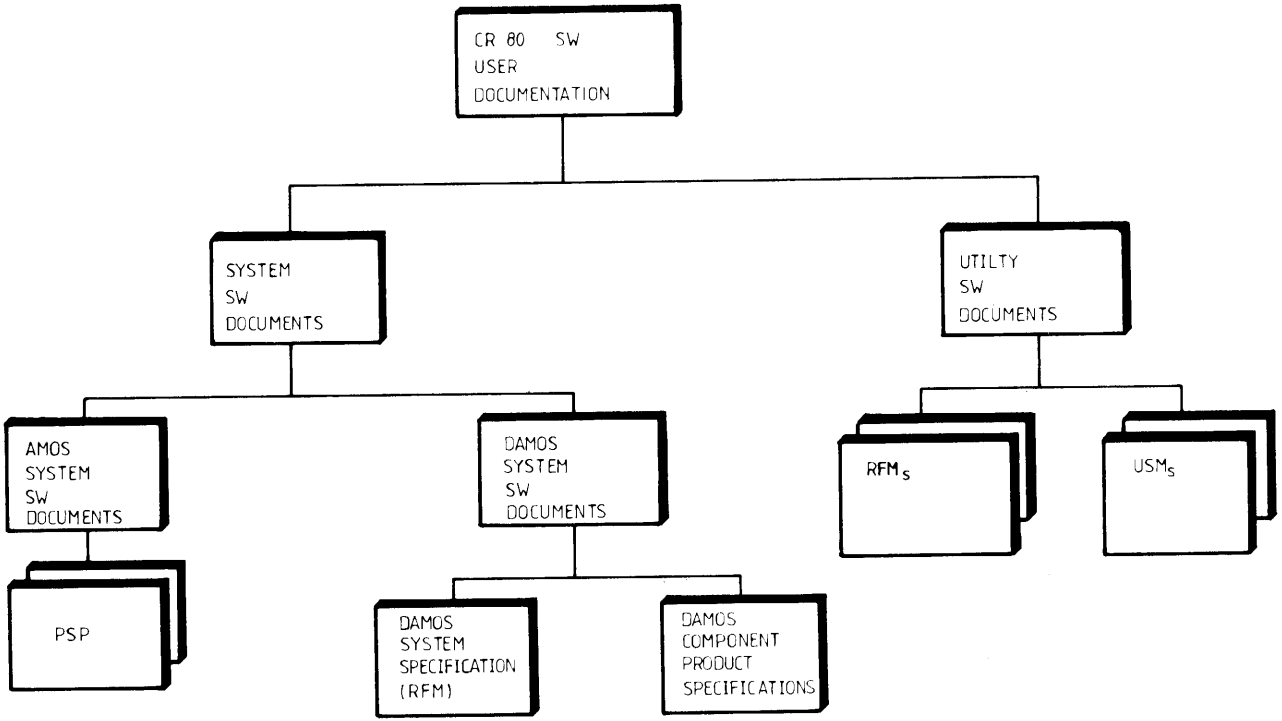
The CR80 Software documentation is divided into:

- product documentation
- user documentation

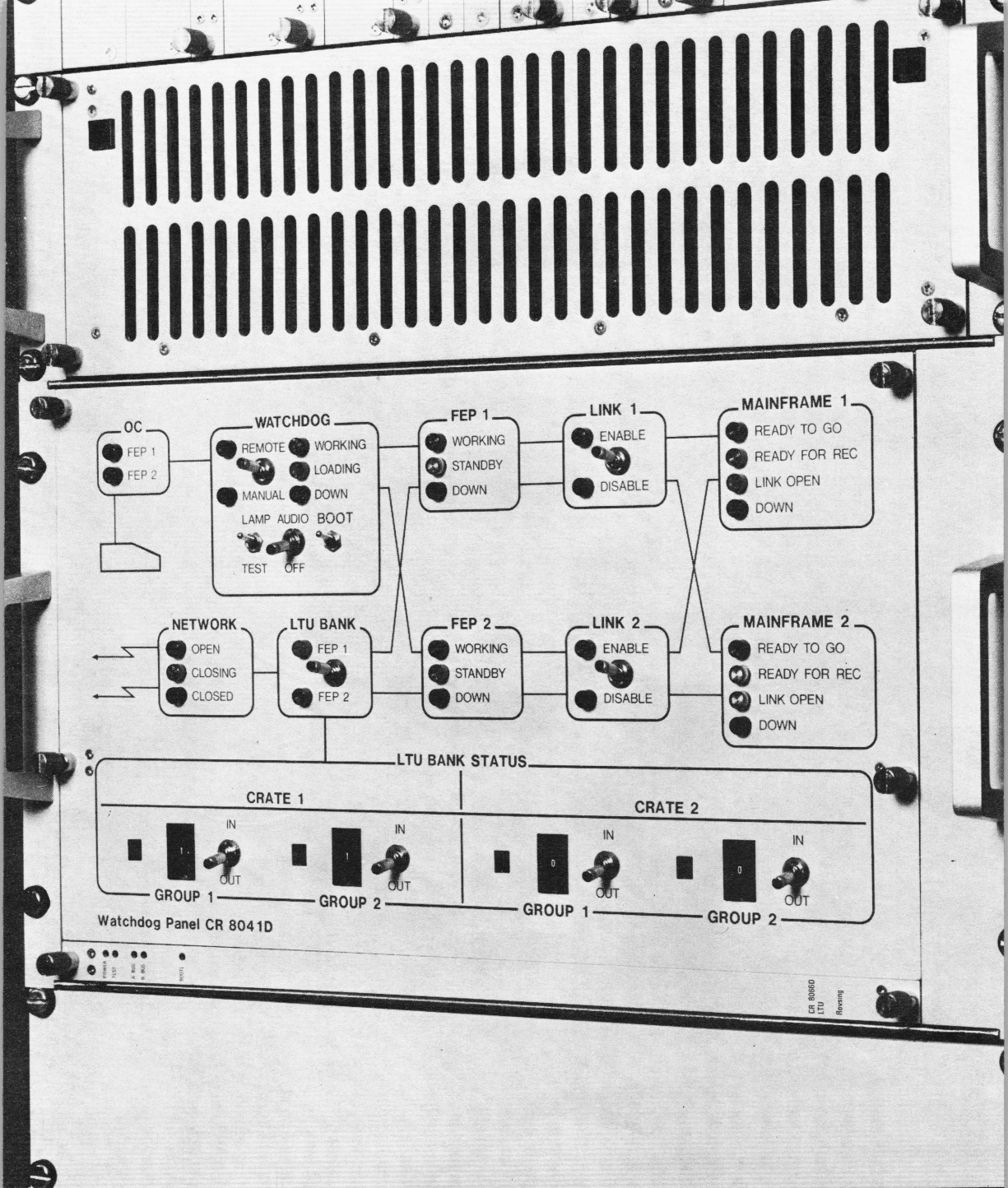
Product documentation contains information about the function, structure and design of a Software component. It has the form of a Product Specification (PSP) and is intended for use by maintenance personnel.

User documentation contains information about the function, operation and/or use of a Software component. It has the form of a Reference Manual (RFM) or a User's Manual (USM). It is intented for use by system analysts and programmers who are using a software component as a building block or as a tool.

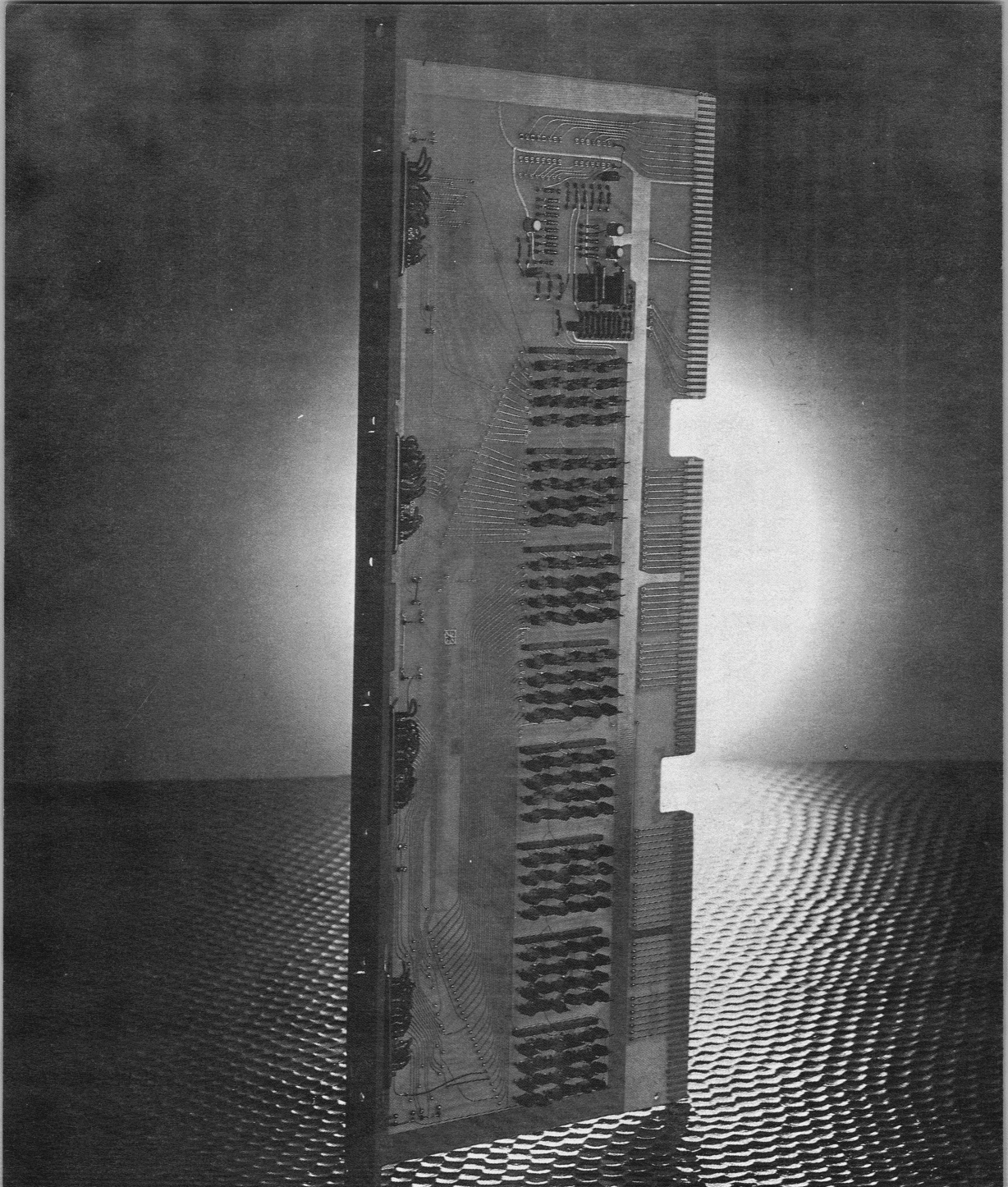
This hierarchy of CR80 software user documentation is described on the figure overleaf.


CR 80 SW
USER
DOCUMENTATION
SYSTEM
SW
DOCUMENTS
UTILTY
SW
DOCUMENTS
AMOS
SYSTEM
SW
DOCUMENTS
DAMOS
SYSTEM
SW
DOCUMENTS
RFMs
USMs
PSP
DAMOS
SYSTEM
SPECIFICATION
(RFM)
DAMOS
COMPONENT
PRODUCT
SPECIFICATIONS

NOTES:



Maintenance and Configuration Processor (MCP) system watchdog. The standard system is here shown with a customized Front Panel, which with the CR80 can be implemented quite simply, as the panel controller brings out all possible combinations of control and status signals to wrap posts. These are available for the user to select the relevant subset of, for graphic representation and nomenclature definition on the Front Panel.



The Switching Line Interface Adapter (LIA-S), implements the unique N + 1 Redundancy Capability of the CR80 Computers. It performs switching of communication lines from a failed Communication Interface to a spare Communication Interface being common to a group of Communication Lines. The switching is controlled by a Maintenance and Configuration Processor (MCP) system watch dog.

7. CR80 Maintenance and Configuration Processor (MCP) System

7.1 Introduction

The Maintenance and Configuration Processor (MCP) system, (System Watchdog (WD)) shown overleaf (fig. 7.1-1), consists of standard CR80 modules distributed in the computer system, interconnected by the redundant Configuration Bus, specially suited for monitoring and control of CR80 Computer Systems which includes redundancy. The elements in the MCP system are described in the following.

The WD CPU (one of the standard CR80 LTU modules) is interfaced either to two PU's via the dual ported CU transfer buses, Data Bus A and Data Bus B, or via V24 ports. These two ports are used for status and control messages between the CR80 system and the MCP system.

The WCA (Watchdog CPU Adapter) constitutes the interface between the WD CPU and the configuration Bus and the four available V24 communication ports. The V24 ports are used for connection of one or two system consoles and for connection to a communication port for remote maintenance and diagnostics of the CR80 system.

The Daisy Chained Configuration Bus is a dualized serial communication path between the WCA and the connected CCA's (Crate Configuration Adapters). The CCA is a standard CR80 adapter module designed for monitoring and control of the PU and CU Crates. The functions available are: monitoring of all Power Supply DC voltages, switching of LIA-S modules (switching a spare LTU to the line instead of a defect module), and monitoring of digital and analog inputs, and control of digital outputs.

The WD CPU and WD Panel Controller utilize alternative paths of the serial configuration bus for control and monitoring of the attached crates and associated modules. The serial configuration bus therefore is redundant with different parts of it being used in AUTO and MANUAL mode.

A fail safe circuit is implemented between the WD CPU and the WD Panel Controller, which performs automatic switching to the manual settings of

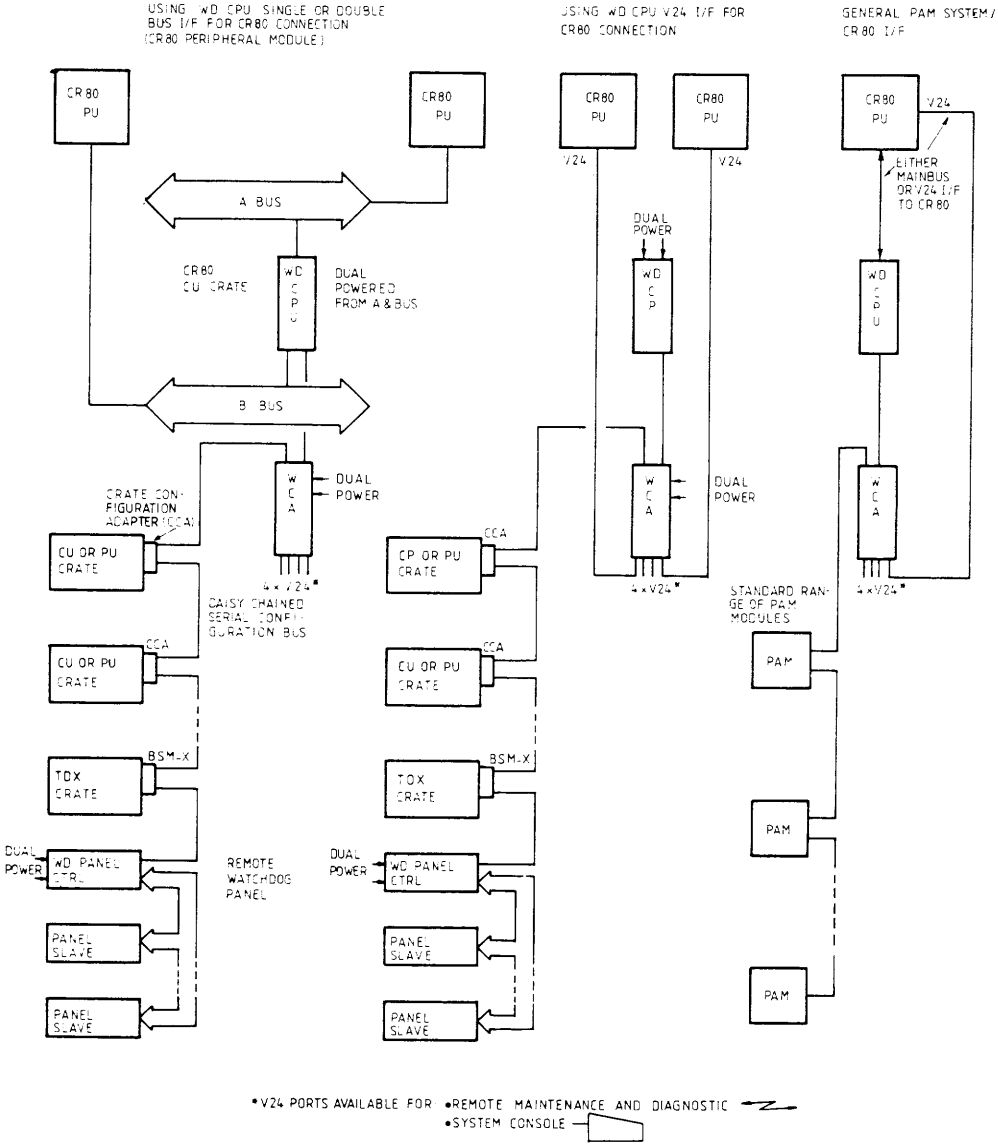


Fig. 7.1-1 CR80 Maintenance and Configuration Processor (MCP) System

the WD Panel in case of WD CPU failure of service. Similarly, replacement of the WD Panel Controller can be done without off-lining the system when under control of the WD CPU (AUTO MODE).

Crates under control of the MCP system is galvanically isolated by optocouplers from the serial configuration bus and can be removed from the operational configuration bus without electrical interference with the remaining part of the system.

Also shown in fig. 7.1-1 is the alternative use of the MCP system as interface between the CR80 Computers and the large range of PAM modules (manufactured by CRAS) for Process Control and Monitoring. The range of PAM modules are shortly described in section 7.8 of this chapter, but use with the MCP system will otherwise not be discussed in the following.

MCP main Characteristics are:

- One of the four V24 ports has extended number of control lines for connecting to modem, where remote diagnostics or logging is implemented.

- The MCP is capable of interfacing via the serial configuration bus with 32 CR80 crates and 32 other crates, and in each CR80 crate performs control of two banks of LTUs, four other switchable devices (Main frame I/Fs, etc.) and for monitoring of up to 10 crate power supply voltages, and 4 other external analog signals and 4 digital inputs.

- On the WD Panel Controller, which attaches to the serial configuration bus, configuration status of the CR80 crates can be displayed and manually changed by switches on the front plate. The WD Panel Controller can modularly be extended with up to 15 slave panels for large display and control systems (corresponding to 32 M-Crates and 32 other crates (TDX crates, etc.)). The WD Panel Controller can either be placed locally with the WD CPU or remoted up to 40

meters via the configuration bus (200 m if greater gauge wire is used with the configuration bus).

- WD-CPU program normally resides in PROM positioned on the WCA module, but alternatively can be bootloaded from the CR80 into WD-CPU RAM.

7.2 MCP System, General Description

This section described the modular general MCP system for CR80 systems. Please refer to block diagram overleaf.

The MCP system consists of 4 major components:

- Watchdog CPU (WPU)
- Watchdog CPU Adapter (WCA)
- Watchdog Panel Controller (WPC)
- Crate Configuration Adapter (CCA)

Extendable by:

- Special Configuration Adapter (SCA)
- Watchdog Slave Panels (WPS)
- Standard Modules from the PAM Process Control Product Range (A/D, D/A, dig. input/output, etc.)

The Watchdog CPU is simply a standard dual bus LTU which boots its program from EPROM located on the Watchdog CPU Adapter (WCA). It communicates to CR80 computers and terminals via 4 x V24 ports of which 3 have limited interface (send, receive, 3 incoming and 3 outgoing control lines) and 1 extended interface (send, receive, up to 9 incoming and 11 outgoing control lines), alternatively the Watchdog LTU can communicate with the CR80 via its common RAM and CR80 bus interface(s) for high speed applications.

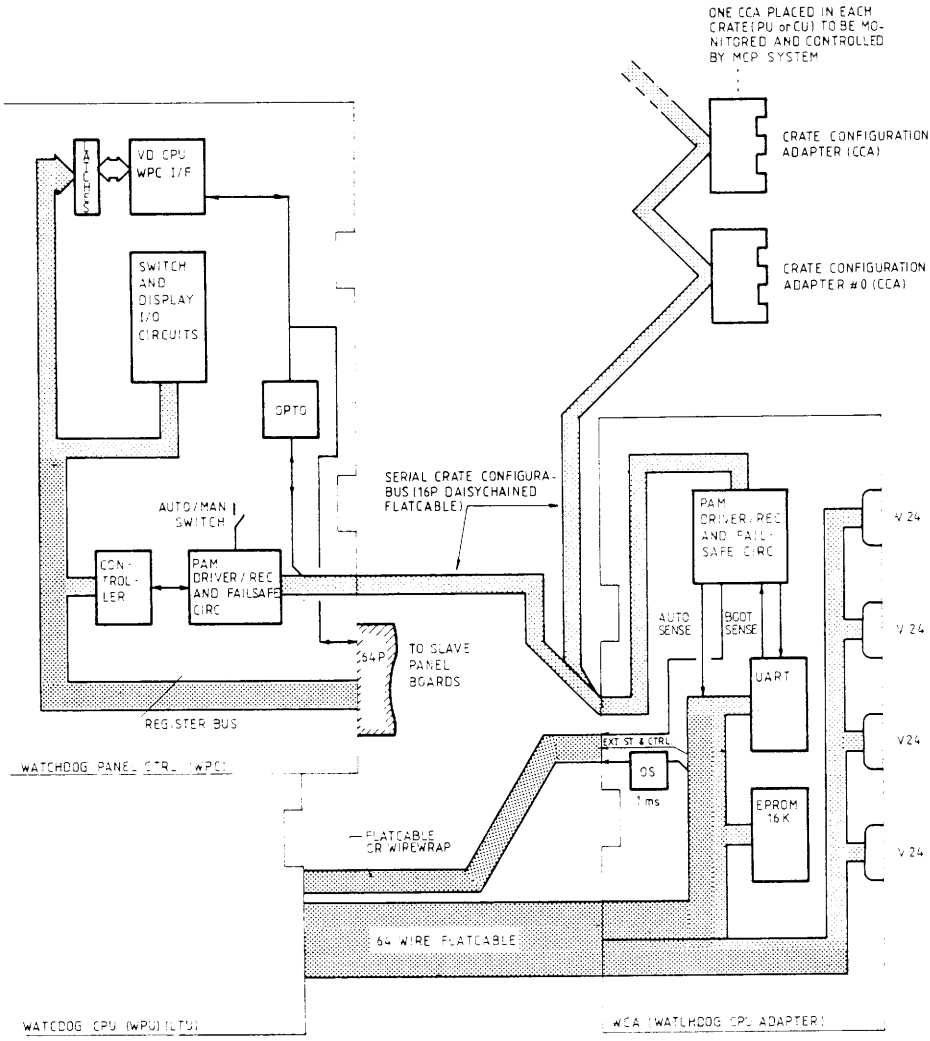


Fig. 7.2-1 MCP System

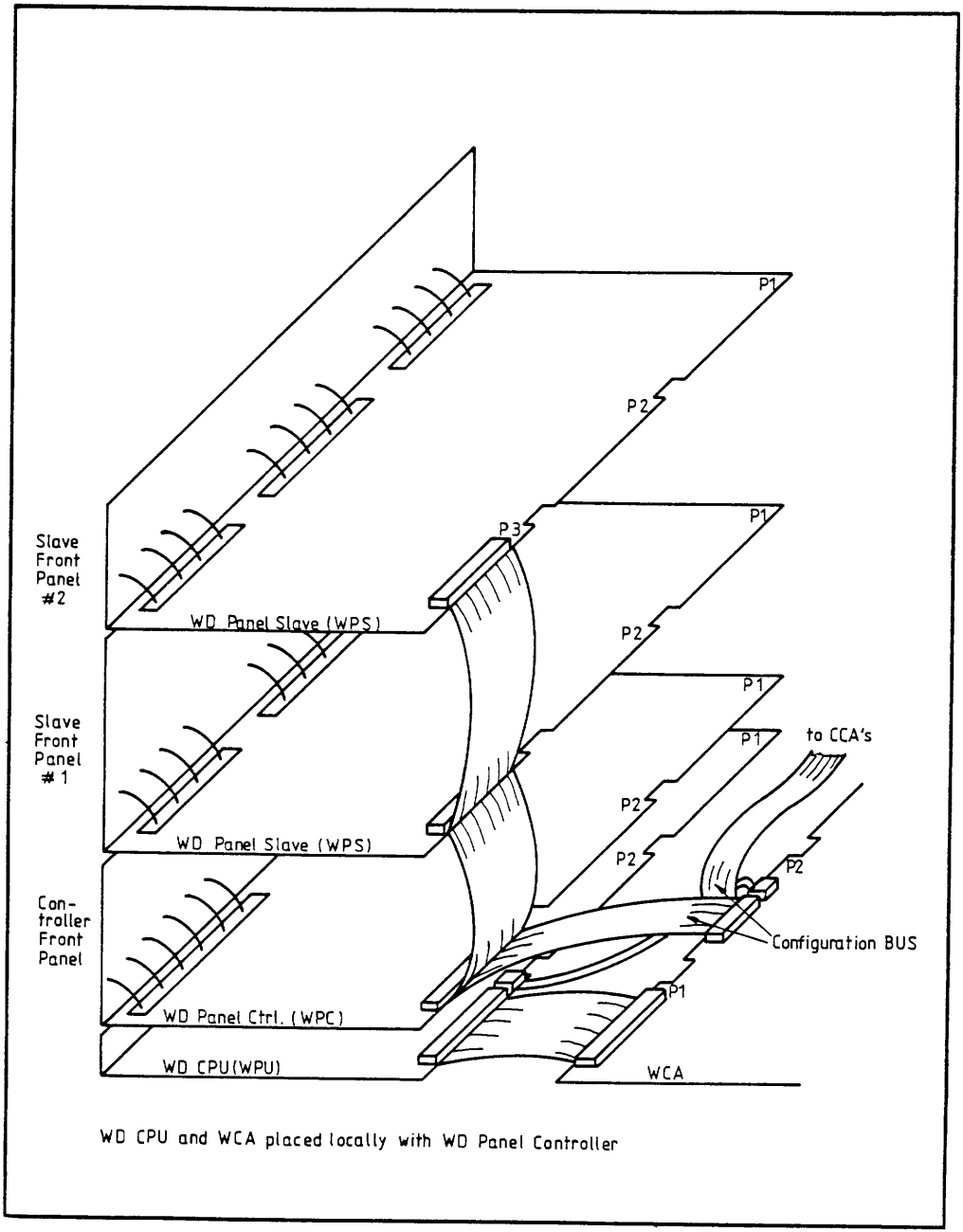


Figure 7.2-2
MCP INTERNAL CABLING

The Watchdog CPU communicates with the CCAs and WPC through the 16p flat cable configuration bus, serial communication at 4800 baud is used, based on the single chip PAM system (PAM, Large Scale Integrated Circuit for parallel to serial conversion and serial communication).

The WCA also contains a fail safe circuit, which in connection with the corresponding circuit on the WPC ensures that only one, either the WPU or the WPC, is in control of the configuration bus, and thereby the set-up and switching of the CR80 system, in which the MCP system is included.

The Watchdog panel controller (WPC) contains a single chip processor, the major task of which, is to scan switches set on its front plate and convey the settings to the CCAs via the configuration bus, using PAM serial communication. Also the actual result of setting the CCAs is read back and displayed on the front panel. An interface between the Watchdog Panel Controller and the WPU PAM lines is provided, thus giving the possibility of displaying the CCA's actual settings when the WPU is in control.

The WPC is modularly extendable with up to 15 slave boards, which interface to the WPC through 64p flat cable in the standard I/O area on J3.

The Crate Configuration Adapter (CCA) contains 1 PAM integrated circuit plus a number of multiplexed registers, interfaced to the configuration bus via OPTO couplers. This allows for control of 2 banks each of 8 LTUs + 1 spare (with LIA-S I/F) or 1 bank of 16 LTUs + 1 spare, switching control of 4 other devices, monitoring of the 10 crate power supply voltages plus 4 other analog signals and 4 digital inputs.

Up to 32 CCAs of this type can be connected to the configuration bus, further 32 PAM addresses are available for extending with special types of configuration adapters (SCA).

An example of an SCA is the BSM-X for monitoring and control of TDX Crates.

7.3 Use of PAM Addresses:

Dec.		
0	0000000	CCA No. 0
.	.	
.	.	
.	.	
.	.	
.	.	
31	0011111	CCA No. 31
32	0100000	SCA No. 32
.	.	
.	.	
.	.	
.	.	
.	.	
63	0111111	SCA No. 63
64	1000000	
	.	
	.	
	.	Reserved for future use
	.	
	.	
79	1001111	
80	1010000	NOT USED
81	1010001	WD CPU I/F on slave panel No. 1
.	.	
.	.	
.	.	
94	1011110	WD CPU I/F on slave panel No. 14
95	1011111	WD CPU I/F on slave panel No. 15
96	1100000	
.		
.		Reserved for special slave panels
.		(time of day, etc.)
.		
119	1110111	
120	1111000	WPU-WPC I/F on WPC board
121	1111001	WPU-WPC I/F on WPC board
122	1111010	WPU-WPC I/F on WPC board
123	1111011	NOT USED
124	1111100	NOT USED
125	1111101	NOT USED
126	1111110	NOT USED
127	1111111	WPU supervision on WCA board

7.4 Fail Safe Circuitry, Indicators and Configuration Bus

The main functions of the fail safe circuit is to ensure that only one, WPC or WD-CPU, has the configuration control, to switch from AUTO to MANUAL mode if the WD-CPU dies and in addition make it possible to bootload WD-CPU software, from EPROM placed on the WCA-board, into RAM on a standard LTU (which hereafter acts as WD-CPU). The purpose of the implemented display logic is to inform the operator about the actual state of the WD-system via the indicators on the WPC front panel.

Please refer to block diagram overleaf, which shows interconnections between the Watchdog Panel Ctrl (WPC) and the Watchdog CPU Adapter (WCA) via the configuration bus, as well as all signals on the configuration bus. Also shown is the fail safe circuit distributed on both boards and working together via the configuration bus in such a way that one of the boards may be removed for repair when the other is in control of the configuration bus.

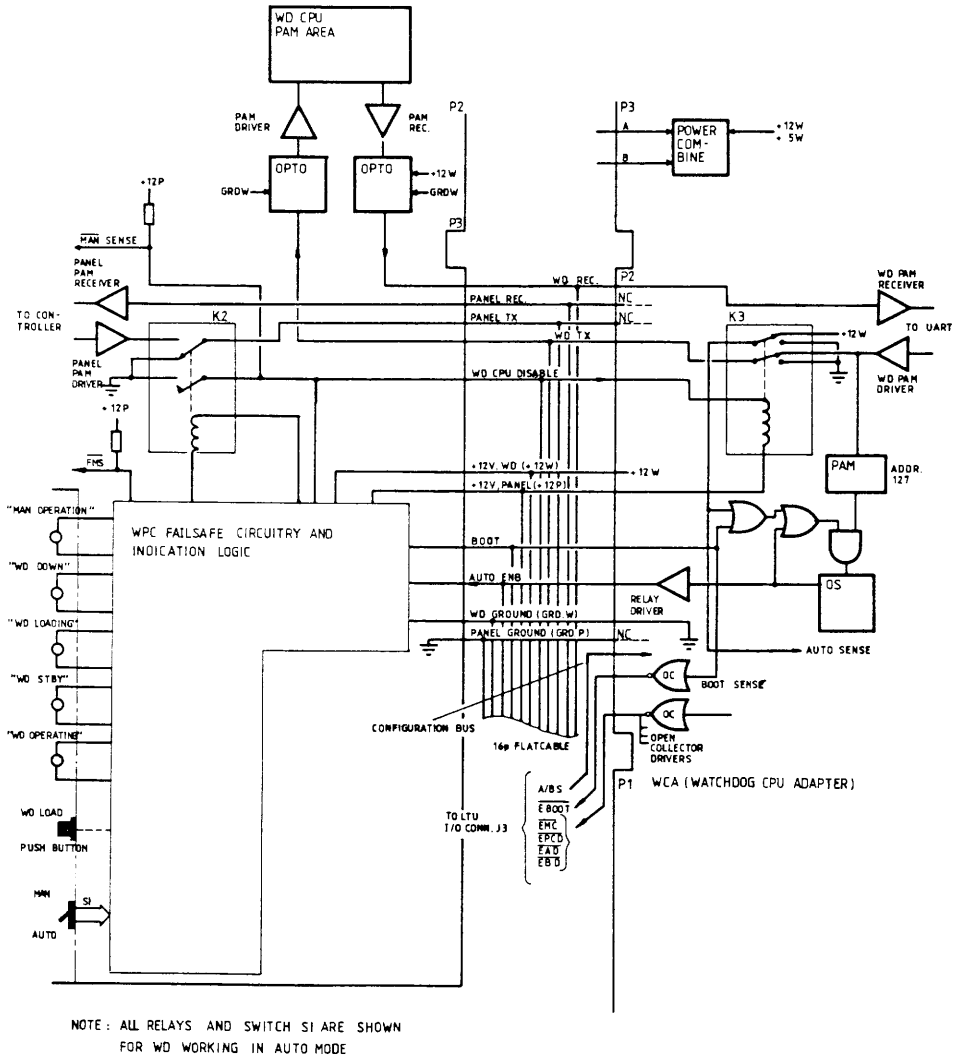
Functional Description

- Manual Mode:

  K2 is off and K3 is on thus ensuring WPC PAM-lines are connected to the Configuration Bus, the WD-CPU PAM-lines are disconnected and the AUTOSENSE line is low. If MANUAL mode is reached due to WD-CPU failure, the AUTO ENB-line is low too, otherwise it is high if the WD-CPU has been booted or running before switching.

- Auto Mode:

  K2 is on and K3 is off ensuring WPC PAM-lines are disconnected from the Configuration Bus, the WD-CPU-PAM-lines are connected and AUTOSENSE-line high. Now in case of the retriggerable



NOTE : ALL RELAYS AND SWITCH SI ARE SHOWN FOR WD WORKING IN AUTO MODE

Figure 7.4-1
CONFIGURATION BUS
PAM rec./driver &
FAILSAFE CIRCUITRY

monitoring "One Shot (OS)" not receiving signals often enough from the PAM-circuit due to WD-CPU failure, the AUTOENB signal will go low thus indicating the failure and the system then automatically switches back to MANUAL mode and give the indications described.

Also the relays K2 and K3 are switching and thereby enabling the Manual mode situation, AUTOSENSE-line low disconnecting the PAM-circuit from the monitoring OS. The system can only be brought back to WD-CPU-operation by one of the following procedures.

- WD-LOAD

The WD-CPU can be bootloaded in both MANUAL mode and AUTO mode switching with indications as listed overleaf. Only in power up situations or after a WD-CPU failure, does it make sense to do a bootloading operation. The bootloading is initiated by toggling the WD-LOAD push button, hereby activating a hold-circuit which makes BOOT-line high.

BOOT-line high informs the WD-CPU to boot its program from EPROM, placed on the WCA, into its memory. It is also opening between the PAM-circuit and the monitoring OS. Still, the WPC PAM-lines are connected to the Configuration Bus and the WD-CPU-lines are disconnected, and AUTOSENSE is held low. When the WD-CPU has finished its bootloading, it starts executing the program.

The first actions hereafter are to start sending positive self test pulses to the monitoring OS and thereby making AUTOENB-line high. Now depending on MAN/AUTO-switch position, the indicators will show the state. If in Manual mode nothing will happen except that WD-STBY lamp is on, switch to Auto mode causes the relays K2 and K3 to connect WD-CPU PAM-lines to Configuration Bus and disconnect WPC PAM-lines from Configuration Bus and that AUTOSENSE-line goes high. WD-CPU now assumes Configuration control. If in Auto mode, the WD-CPU will have control almost immediately.

Table of WD. Front Panel Indicators

SITUATION / INDICATION	MAN MODE WD DEAD	MAN MODE WD BOOTING	MAN MODE WD READY	AUTO MODE WD OPERATING	AUTO MODE WD DEAD	AUTO MODE WD BOOTING
• MAN OPERATION	X	X	X		X	X
• WD DOWN	X				X	
• WD LOADING		X				X
• WD STBY			X			
• WD OPERATING				X		
(ALARM)					(X)	

X: Lamp On

(ALARM) indication is issued by software, others by hardware logic.

7.5 Watchdog CPU, Adapter (WCA)

The WCA, shown overleaf, connects to the WD CPU (LTU) via the standard LTU 4 channels V24 interface (standard 64p flatcable connection between Front and Adapter module), but uses the not essential control lines (12 input and 12 output lines) for implementing a bus connection to the WCA.

The LTU-WCA bus consists of 8 LTU output data lines (OPAØ-OPA7), 8 LTU input data lines (IPBØ-IPB7), 4 LTU output bus control ines (OPCØ-OPC3) and 4 LTU input control lines (IPC4-IPC7). All lines are internally on the LTU connected to A, B and C ports of a PIO.

The 8 output data lines are governed by the 4 output control lines used for addressing the WCA EPROM, loaded into latch for extending output control lines of one of the V24 interfaces, for loading the UART for serial transmission to the WD CPU PAM line on the configuration bus, and for controlling the external control lines to the LTU.

The 8 input data lines are governed by the 4 output control lines used for reading data out of EPROM, UART or extended input lines of one of the V24 interfaces.

The 4 input control lines are used by the LTU for sensing UART receive full, UART transmit empty and AUTO sense.

Also the WCA connects to the LTU, via separate cable and lines from the standard 64p flatcable, for external control of the standard LTU, when it is used as a Watchdog CPU:

- External boot, $\overline{\text{Eboot}}$: The WCA by pulling this line down initiates the LTU bootload procedure and switches the LTU bootloading PROM on.

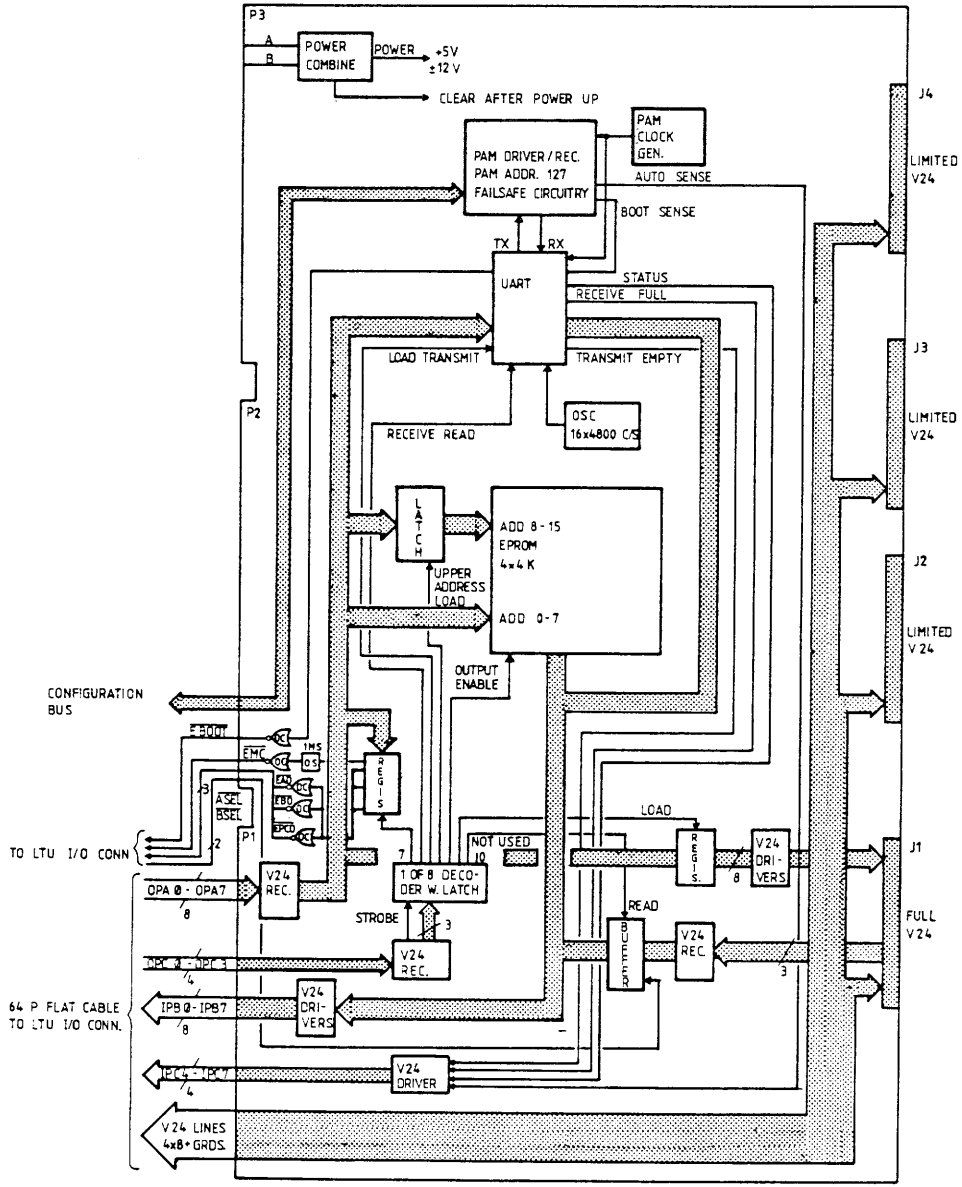


Figure 7.5-1
WATCHDOG CPU ADAPTER (WCA)
BLOCK SCHEMATIC

- External Master Clear, $\overline{EMC}$: The WCA by pulling this line down after a finished bootload, initiates executing of the program now located in RAM.

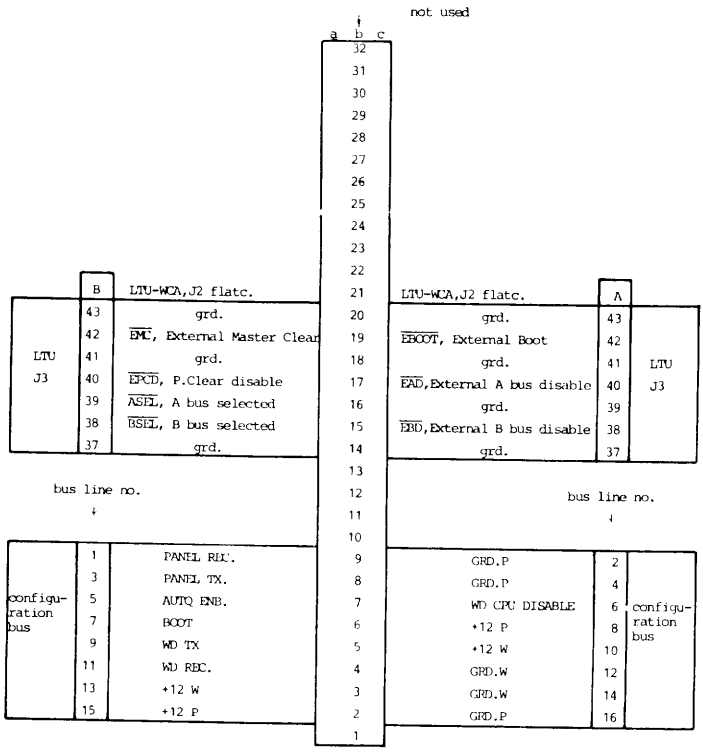
- External bus disable, $\overline{EAD}$ & $\overline{EBD}$: By pulling these lines down, the A, B or both busses are disabled for access from the CR80.

- External Programmed Clear Disable, $\overline{EPCD}$: Pulling this line down disables clearing of the LTU when switching between the A and B bus is performed by the CR80, "Toggle to me" commands.

- A/B select status, $\overline{ASEL}$, $\overline{BSEL}$: This line informs the WCA, which bus (A, B or none) the LTU is currently connected to; high indicates bus not selected and low indicated bus selected.

All the lines above are open collector, active low, with the terminating resistor on opposite board of the driver. Ensuring that lines will not be active when not connected to driver.

WCA Adapter J2



CHRISTIAN ROVSING A/S

WCA Connector J2

Fig. 7.5-2

ASM/ MM

30-1139

LTU Connector J3

	c	LTU-WCA,J2 flatc.	A	B	LTU-WCA,J2 flatc.	a	
	20	grd.	43		grd.	20	
	19	$\overline{\text{EXCOT}}$, External Boot	42		External Master Clear, $\overline{\text{EMC}}$	19	
WCA	18	grd.	41		grd.	18	WCA
P2	17	$\overline{\text{EAD}}$, External A bus disable	40		P.Clear disable, $\overline{\text{EPCD}}$	17	P2
	16	grd.	39		$\overline{\text{ASEL}}$, A bus selected	16	
	15	$\overline{\text{EBD}}$, External B bus disable	38		$\overline{\text{BSEL}}$, B bus selected	15	
	14	grd.	37		grd.	14	
			36				
			35				
	c	LTU-WCA, J1 flatcable	34		LTU-WCA, J1 flatcable	a	
	32	TXDA (A-103)	33		(A-104) RXDA	32	
	31	RTSA (A-105)	32		(A-108) DTRA	31	
	30	DCDA (A-109)	31		(A-106) CTSA	30	
	29	RXCA (A-115)	30		IPB1	29	
	28	IPBØ	29		OPA1	28	
	27	OPAØ	28		(A-114) TXCA	27	
	26	grd.	27		grd.	26	
	25	OPCØ	26		IPC4	25	
	24	TXDB (B-103)	25		(B-104) RXDB	24	
	23	RTSB (B-105)	24		(B-108) DTRB	23	
	22	DCDB (B-119)	23		(B-106) CTSB	22	
	21	RXCB (B-115)	22		IPB3	21	
	20	IPB2	21		OPA3	20	
	19	OPA2	20		(B-114) TXCB	19	
	18	LTU SEL.	19		grd.	18	
WCA	17	OPC1	18		IPC5	17	WCA
P1	16	OPC2	17		IPC6	16	P1
	15	TXDC (C-103)	16		(C-104) RXDC	15	
	14	RTSC (C-105)	15		(C-108) DTRC	14	
	13	DCDC (C-109)	14		(C-106) CTSC	13	
	12	RXCC (C-115)	13		IPB5	12	
	11	IPB4	12		OPA5	11	
	10	OPA4	11		(C-114) TXCC	10	
	9	grd.	10		grd.	9	
	8	OPC3	9		IPC7	8	
	7	TXDD (D-103)	8		(D-104) RXDD	7	
	6	RTSD (D-105)	7		(D-108) DTRD	6	
	5	DCDD (D-109)	6		(D-106) CTSD	5	
	4	RXCD (D-115)	5		IPB7	4	
	3	IPB6	4		OPA7	3	
	2	OPA6	3		(D-114) TXCD	2	
	1	grd.	2		grd.	1	
			1				

CHRISTIAN ROVSING A/S

LTU Connector

J2 to WCA connectors: J1 and J2

Fig. 7.5-3

Date	80-10-10	Approved	Date 1 80 10 10
Drawn	ASM/MM		
Drawing no.	30-1140		
Project no.		Scale 1 : 1	Sheet

7.6 Watchdog Panel Controller (WPC) and Slave Panels (WPS)

The central point of the WPC, shown overleaf, is the single chip microprocessor, which interfaces to the configuration bus directly via serial interface and to all other circuits via the WPC register bus.

The microprocessor is able to communicate, via the register bus with buffers for input switch setting and latches for outputting to indicator LEDs on its front plate, with up to 15 slave panels and also has an interface to the WD-CPU via the register bus/WD-CPU PAM serial bus.

- Functional Description

  In MANUAL mode (the WPC has configurration control) the WPC microprocessor performs the simple task of scanning cyclically the PAMs of the configuration adapters (CCA or SCA) from PAM address 0-63 and doing a direct mapping of register bus addresses and PAM addresses.

  It will for each PAM read the corresponding control switch settings 5 or 4 bit on its own or slave front panel, transmit it to the PAM, receive the status back from the PAM and display it on its own or slave front panel, afterwards going on to the next PAM.

  The maximum number of CCAs (SCAs) connected to the configuration bus is read by the controller on a 8 bit DIL switch.

  The advantage of this scheme is that customization of front panels can be done by wiring alone as no active circuits included and firmware does not need to be changed for different WD panel designs.

  In AUTO mode the WD-CPU can hand over the status of CCA-PAMS, via the register bus/WD-CPU PAM I/F, so that actual status of the CR80

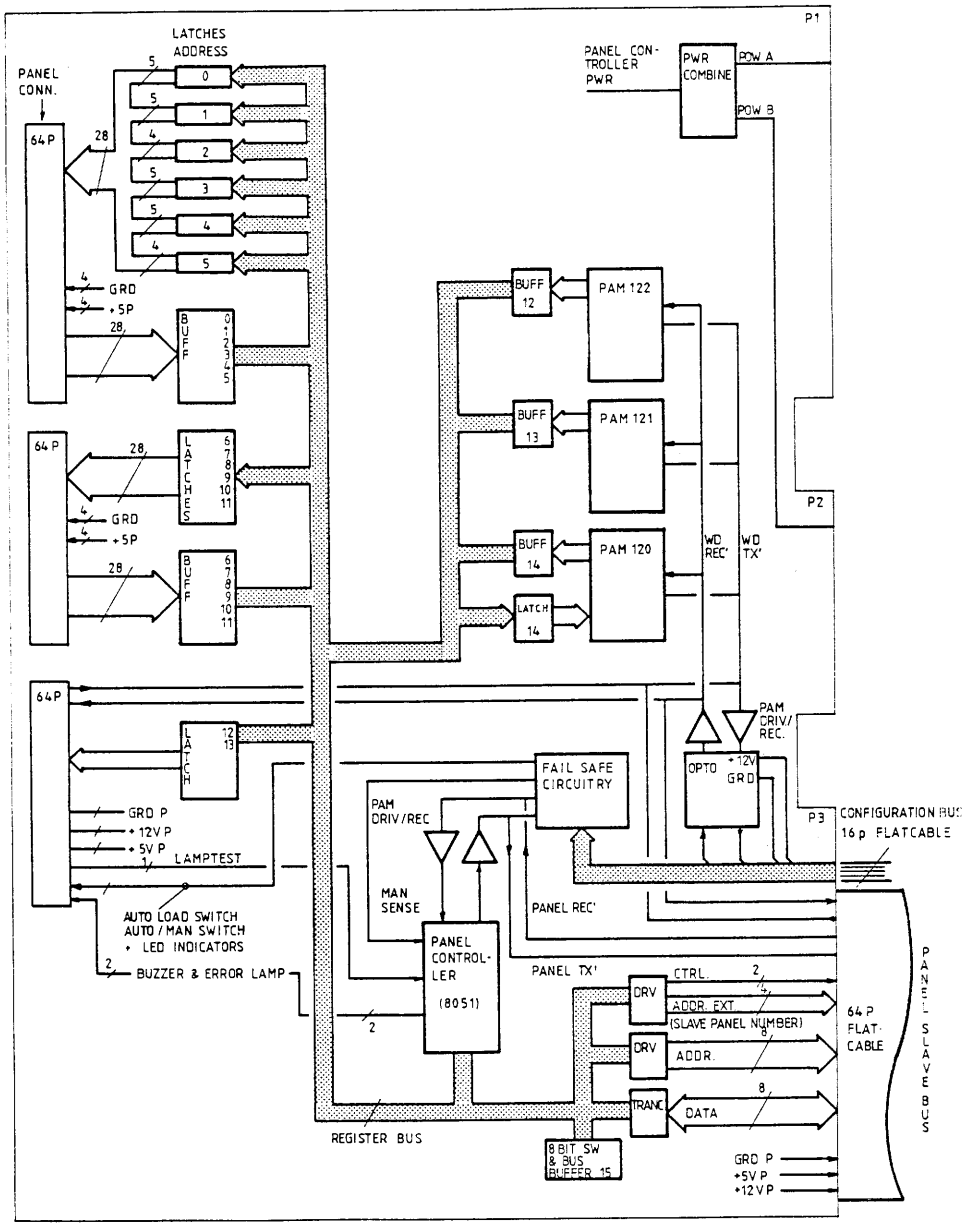


Figure 7.6-1

WATCHDOG PANEL CONTROLLER (WPC)

system can be displayed. This can simply be done by loading from the WD-CPU PAM addresses 120 and 121 with the register bus latch address and handshake signal, and PAM address 122 with the status to be displayed.

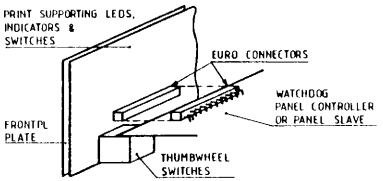
The WPC microprocessor polls on handshake signals when in AUTO mode (MAN sense high), receives status word, and loads this to the indicator latch pointed out, and returns a "status taken signal" via PAM address 120.

- Lamp Test

When the controller senses the lamp test switch "ON", it enters a test cycle in which it will set all bits in the indicator latches, first high then low and thereafter resume either polling of CCA associated switch settings (Manual mode) or display of WD-CPU handed over status. It must be pointed out that there is made no attempt to remember any status from before lamp test and that in Manual mode polling starts from the lowest read buffer number.

- Slave Panels and Generalized Front Panel

Overleaf is shown the standard generalized front panel layout used with the WPC and WPS; as well as the functional block diagram of the Watchdog slave panel (WPS).



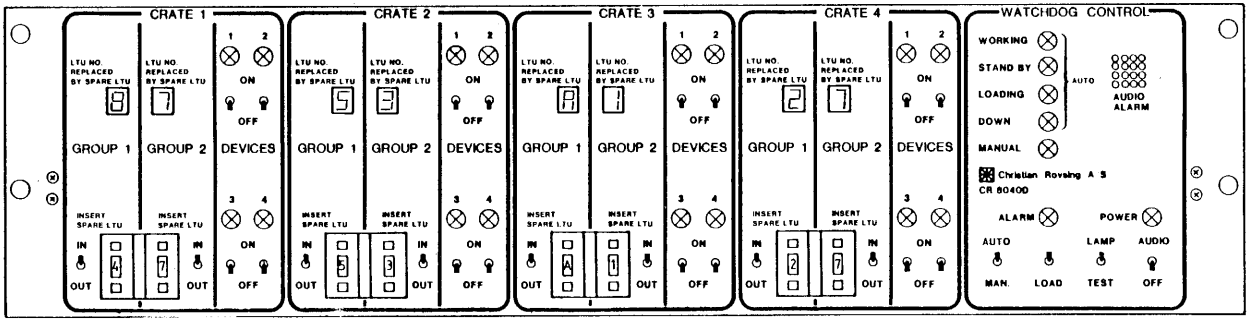


Figure 7.6-2
Standard Front Panel
For Watchdog Panel Ctrl.

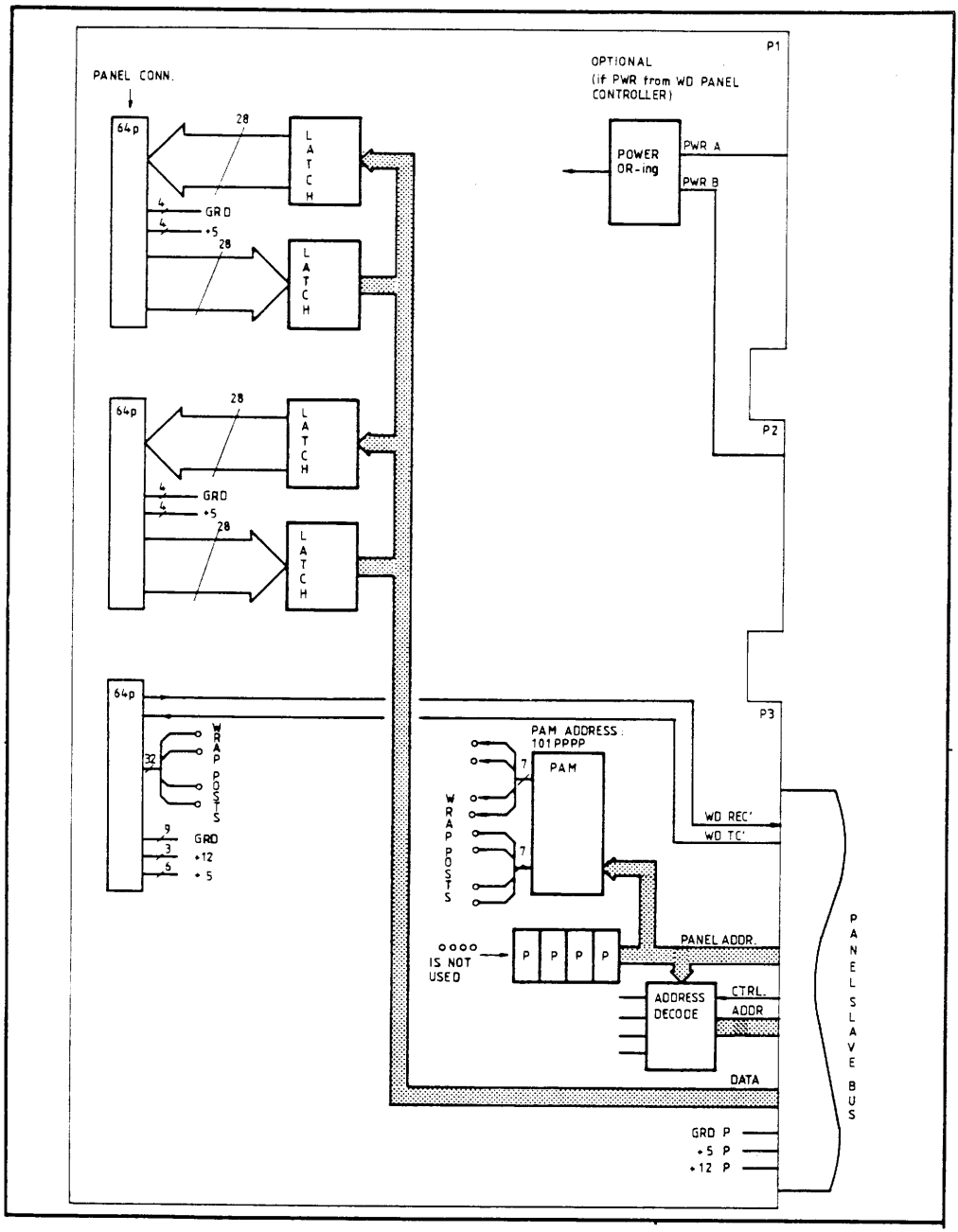


Figure 7.6-3
WATCHDOG SLAVE PANEL (WPS)

WD Panel Ctrl. (WPC) Connection to Slave Panels (WSP and Configuration Bus)

	Bus Line No.		WPC P3 A		B	Bus Line No.	
Configuration bus	2	GRD. PANEL	43	PANEL REC	1	3	Configuration bus
	4	GRD. PANEL	42	PANEL TX	3	5	
	6	WD-CPU DISABLE	41	AUTO ENB	5	7	
	8	+12 PANEL	40	BOOT	7	9	
	10	+12W	39	WD TX	9	11	
	12	GRD. WD	38	WD REC.	11	13	
	14	GRD. WD	37	+12W	13	15	
	16	GRD. PANEL	36	+12 PANEL	15		
			35				
	A		34		B		
(WPS) Slave Panel J3	33	WD TX	33	GND PANEL	33		(WPS) Slave Panel J3
	32	WD REC	32	GND PANEL	32		
	31	GND PANEL	31	PANEL TX	31		
	30	GND PANEL	30	PANEL REC	30		
	29	GND PANEL	29	GND PANEL	29		
	28	NC	28	NC	28		
	27	NC	27	NC	27		
	26	GND PANEL	26	GND PANEL	26		
	25	ERB0	25	ERB1	25		
	24	ERB2	24	ERB3	24		
	23	GND PANEL	23	GND PANEL	23		
	22	ERB4	22	ERB5	22		
	21	ERB6	21	ERB7	21		
	20	GND PANEL	20	GND PANEL	20		
	19	EAB0	19	EAB1	19		
	18	EAB2	18	EAB3	18		
	17	GND PANEL	17	EAB4	17		
	16	EAB5	16	EAB6	16		
	15	EAB7	15	GND PANEL	15		
	14	PA0	14	PA1	14		
	13	PA2	13	PA3	13		
	12	GND PANEL	12	GND PANEL	12		
	11	ERD	11	GND PANEL	11		
	10	GND PANEL	10	EWR	10		
	9	$\overline{\text{EENB}}$	9	GND PANEL	9		
	8	NC	8	NC	8		
	7	NC	7	NC	7		
	6	NC	6	NC	6		
	5	GND PANEL	5	GND PANEL	5		
	4	GND PANEL	4	GND PANEL	4		
	3	+5V	3	+5V	3		
	2	+5V	2	+5V			
			1				

7.7 Crate Configuration Adapter (CCA)

The crate configuration adapter, shown overleaf, is capable of switching two banks of each up to 8 LTUs + 1 spare or 1 bank of up to 16 LTUs + 1 spare under control of either the WD-CPU or WPC. Also it will switch up to 4 other devices, as well as monitoring all 10 crate voltages, V_{ref} and grd. for A/D adjustment + 4 other analog signals and up to 4 digital input signals. The CCA is based on 1 PAM integrated circuit and a number of multiplexed registers, of which 2 registers are used for LIA-S bank control (LTU), 1 register used for control of other devices than LIA-S and for monitoring digital signals, the last register is used with a 10 bit A/D converter with 16 ch. multiplexer for measuring voltages and analog signals, and 4 digital input monitoring bits. The PAM circuit is interfaced to the configuration bus via optocouplers, ensuring galvanic isolation between monitored and controlled CR80 crates.

The CCA has test measurement plugs for all 10 power supply voltages, grd. and V_{ref} for the A/D converter brought out on the front panel as well as the A/D adjustment potentiometer is accessible with screwdriver through hole in the front plate.

Special configuration adapters (SCA) are in their internal organisation to be an exact subset of the CCA.

7.7.1 CCA Procedures

- Three transmissions are necessary to load a control register:

  1) C_6, C_5 = 00: send pointer P_0-P_4 = C_0-C_4.

  2) Check that correct pointer was received by CCA in returned status.

  3) C_6, C_5 = 01: send data D_0-D_4 = C_0-C_4.

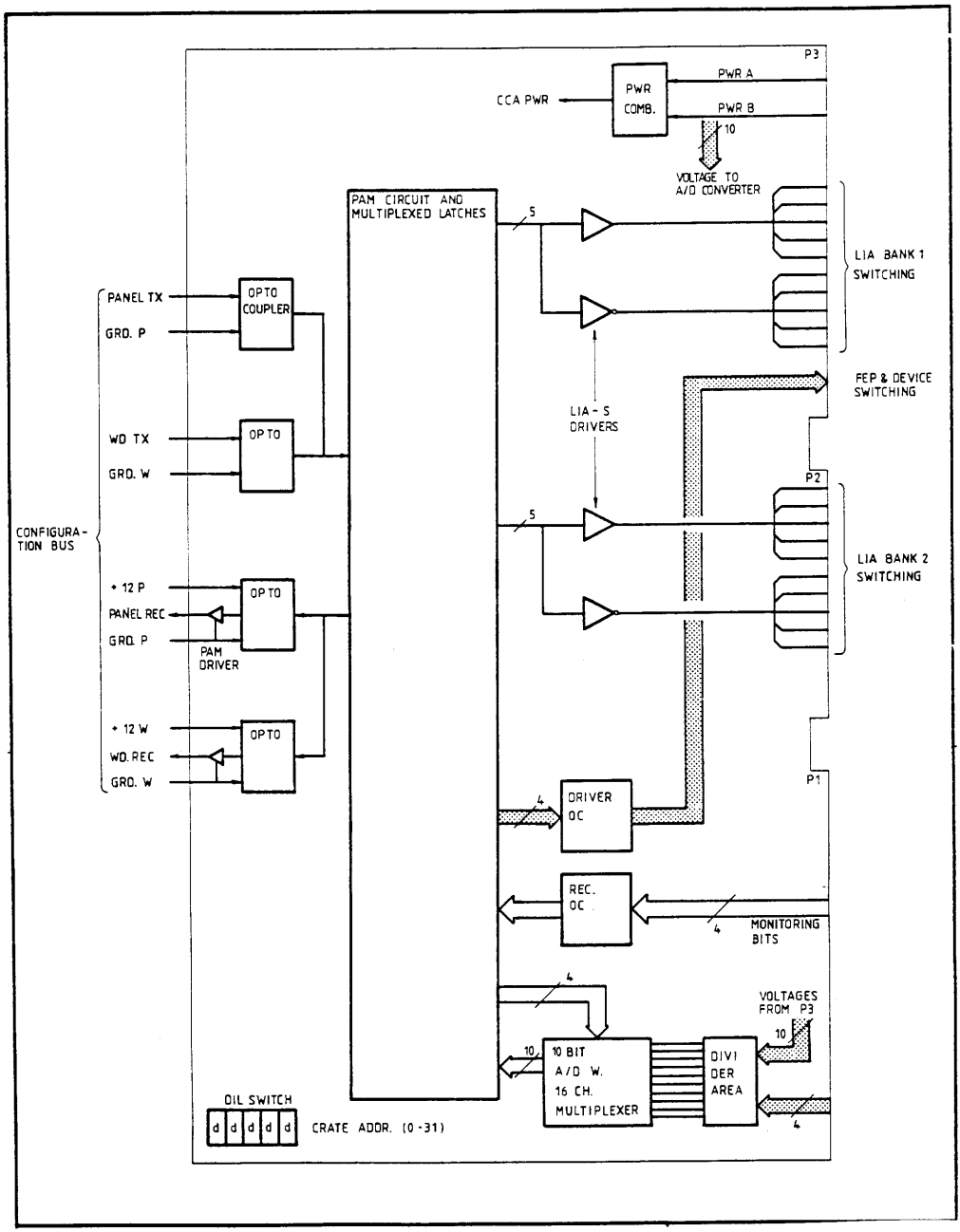


Figure 7.7.1-1

CRATE CONFIGURATION ADAPTER (CCA)

Block Schematic

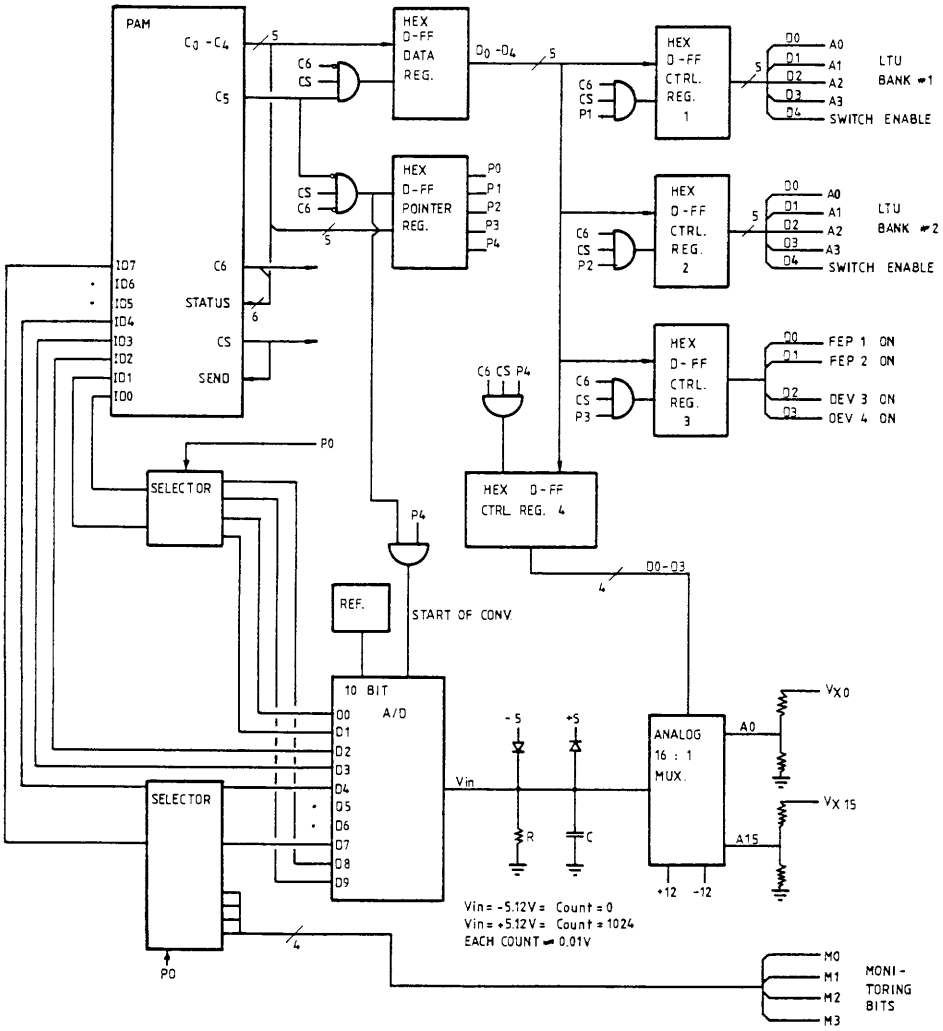


Figure 7.7.1-2
CRATE CONFIGURATION ADAPTER (CCA)
PAM CIRCUIT & MULTIPLEXED LATCHES

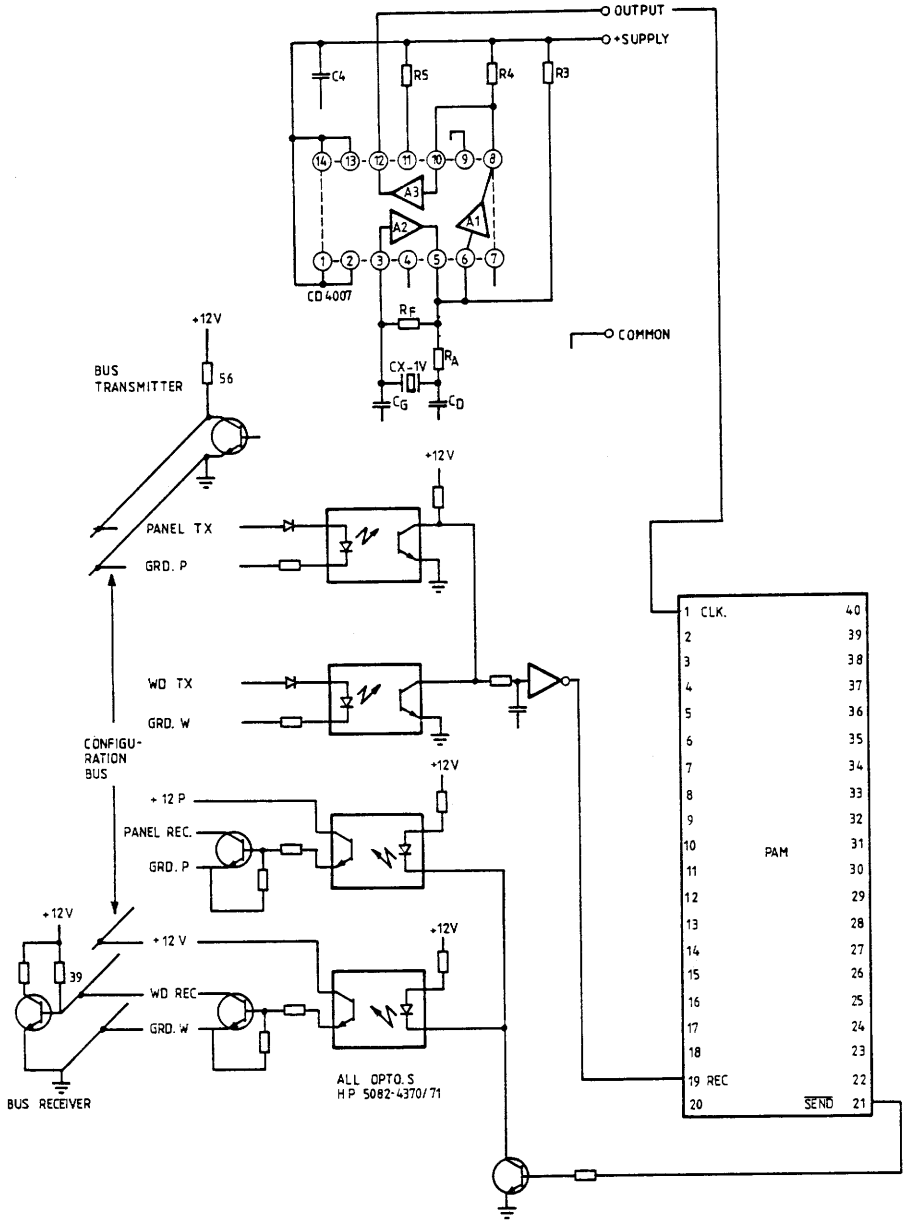


Figure 7.7.1-3
CCA OPTOCOUPLER I/F
TO CONFIGURATION BUS AND CLOCK CIRCUIT

4) Check that correct data was received by CCA in returned status.

5) C_6, C_5 = 1X strobe "data" into "pointed" at ctrl register.

6) Check that the C_6 = 1 was received by CCA in returned status.

- A/D Measurement

1) Using procedure above, load ctrl. reg. 4 with analog ch. No. (D_0-D_3), I_{in} starts to settle on $C_{,}$. and conversion is started on each command strobe (CS) to pointer register.

2) Reset pointer register, P4: C_6, C_5 = 00, send pointer P4 = C4 = 0, this stops conversion and ensures that at least one conversion has taken place. For longer settling times of conversions, other transmissions can take place between 1) and 2) as long as pointer register bit P4 is remained set.

3) Load pointer register with P_0 = 0; this returns upper 8 bit of 10 bit conversion (D_2-D_9) in ID0 to ID7.

4) Load pointer register with P_0 = 1; this returns lower 2 bits of 10 bit conversion (D_0-D_1) in ID6 and ID7 as well as digital input monitoring bits (M_0-M_3) in ID0-ID3.

- Digital Input Monitoring

As for point 4 under A/D measurement above.

CCA conventions for PAM signals:

	$P_4 \cdots P_0$	
LTU Bank 1	0001X	C_0-C_3: LTU no. to be replaced by spare C_4: Switch enable
LTU Bank 2	0010X	C_0-C_3: LTU no. to be replaced by spare C_4: Switch enable
Other devices	0100X	C_0: FEP1 on, C_1: FEP2 on C_3: Dev3 on, C_4: Dev4 on
A/D channel select	1000X	C_0-C_3 Analog channel no: 0 +24V-A 1 +12V-A 2 + 5V-A 3 -12V-A 4 -24V-A 5 +24V-B 6 +12V-B 7 + 5V-B 8 -12V-B 9 -24V-B 10 not used 11 not used 12 not used 13 not used 14 V_{ref} 15 ground A/D start of conversion is enabled
A/D return lower 8 bit of 10 bit A/D in ID0-ID7.	00XXXX0	
A/D return upper 2 bit of 10 bit A/D in ID0 and ID1 (A/D start conversion must have been issued previously), 4 digital input monitoring bits M_0-M_3 are returned in ID_4-ID_7	00XXXX1	
Load DATA reg.	01XXXX	
Load POINTER reg.	00XXXX	
Strobe content of DATA reg. into CTRL reg. pointed at by POINTER reg.	1XXXXX	

CHRISTIAN ROVSING A/S

CCA conventions for PAM signals

Scale		Approved		Issue	1
Date	80-10-16	E		Date	80-10-16
Drawn	ASM/MZL			Approved	
Parts no		30-1155			
Print no		Sheet 1 of 1 sheets			

The following V_x resistors and R division ratios are implemented to give natural number presentation:

Range	Ratio	Range of V_x (count: -512 to +512)	PS voltages measured	Multiplier for natural representation	Measurement increments
1	$V_{in} = 1/2V_x$	+/-10.24V	+5V	2	0.02V
2	$V_{in} = 1/4V_x$	+/-20.48V	+/-12V	4	0.04V
3	$V_{in} = 1/8V_x$	+/-40.96V	+/-24V	8	0.08V

The analog signals are converted to digital in twos complement notation, 10 bit are used.

IO_7	IO_6	IO_5	IO_4	IO_3	IO_2	IO_1	IO_0
D_9 sign+/-	D_8	D_7	D_6	D_5	D_4	D_3	D_2

IO_7	IO_6	IO_5	IO_4	IO_3	IO_2	IO_1	IO_0
D_1	D_0	X	X	M_3	M_2	M_1	M_0

12 of the 16 analog lines are used for measuring the output voltages of the crate supplies, ground and V.ref.

The remaining, unassigned 4 analog lines have the input range 2 (+/-20.48V). The can for example be used for measuring temperatures in the crate.

7.7.2 Crate ConfigurationAdapter (CCA) in PU and CU's

The CCA normally inserted in module position 1 in the Adapter Crate (PU or CU rear magazine). Below the interface and functions of the CCA in PU and CU's is described.

- CCA in PU

Functions:

- To monitor the DC Power Supply voltages via Adapter Crate backplane connector J3, and eventually other analog and digital signals via Adapter Crate backplane interface connector J1 (see Table 7.7.2a).

- To control the PU-Adapter enable line (J3, backplane bus line) by one of its four device switching outputs on J3 (dev. on). Pulling the PU adapter enable line down, disables all external signals from the PU (Datachannel, Suprabuses etc.), by disabling of linedrivers directly. The other three device switching outputs are free for application defined device switching.

Interface in PU:

Adapter Crate backplane connector J3 (see chapter 3, section 3.9).

Adapter Crate backplane connector J2 normally not used.

Adapter Crate backplane connector J1 (see table 7.7.2a).

- CCA in CU

Functions:

- To monitor the DC Power Supply voltages via Adapter Crate backplane connector J3 and eventually other analog and digital signals via Adapter Crate backplane connector J1.

- To control N+1 redundancy switching of LTUs, either one bank of up to 16+1 LTUs via Adapter Crate backplane connector J3 or two banks of up to 8+1 LTUs via Adapter crate backplane connectors J3 for the one bank and J2 for the second bank.

  Figure 7.7.2-1 illustrates and describes cabling or wirewrap continuation of Adapter Crate backplane bus lines J3 and J2 position, with the different bank combinations (1x(8+1), 1x(16+1) or 2x(8+1)).

- To control ON/OFF switching of other devices (e.g. Mainframe I/F's etc.) by the four device switching outputs on J3 (device on J3 bus line or wirewrap connection).

Interface in CU:

Adapter Crate backplane connector J3 (see chapter 3, section 3.9).

Adapter Crate backplane connector J2 (see chapter 3, section 3.10 and table 7.7.2b).

Adapter Crate backplane connector J1 (see table 7.7.2a).

Table 7.7.2a.

CCA, J1 Signal Pin Positions

ADAPTOR CONNECTOR J1

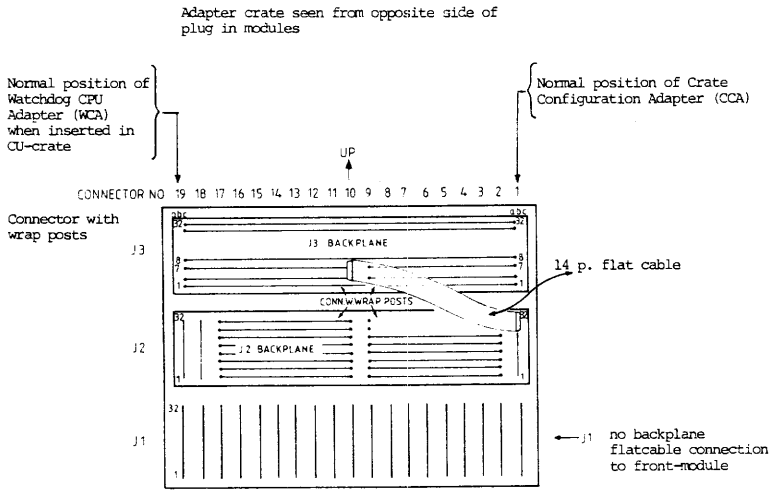
A	not used ↓ B	C	
GND	32		GND
AN3 (Analog inp.)	31	(Analog inp.)	AN4
GND (3)	30		GND (4)
AN1 (Analog inp.)	29	(Analog inp.)	AN2
GND (1)	28		GND (2)
	27		
	26		
	25		
	24		
	23		
	22		
	21		
	20		
	19		
	18		
	17		
	16		
	15		
	14		
	13		
	12		
	11		
	10		
	9		
	8		
	7		
	6		
	5		
M3 (dig. inp.)	4	(dig.inp.)	M4
GND	3		GND
M1 (dig. inp.)	2	(dig.inp.)	M2
GND	1		GND

Table 7.7.2b

CCA, J2 Signal pin positions.

ADAPTER CONNECTOR J2

a	b	c
Switch inhibit	32	Grd
Grd	31	Grd
A3	30	A3
A2	29	A2
A1	28	A1
A0	27	A0
Switching enable	26	Grd
	25	
	24	
	23	
	22	
	21	
	20	
	19	
	18	
	17	
	16	
	15	
	14	
	13	
	12	
	11	
	10	
	9	
	8	
	7	
	6	
	5	
	4	
	3	
	2	
	1	



CU Adapter Crate normally comes equipped for 2 banks of up to 8 LTU's + 1 spare LTU. In case only one bank of up to 16 LTU's + 1 spare is wanted, then the flatcable is removed and backplane J3, pins a1-a7 and c1-c7 is wirewrapped between connector No. 9 and 10, likewise is backplane J2, pins a1-a32 and c1-c32 wirewrapped between connector No. 9 and 10.

CHRISTIAN ROVSING A/S			
CU Adapter Crate MOUNTING OF BACK PLANES FIG. 7.7.2-1	Scale	Approved	Issue
	Date 80-10-13		Date 80-10-13
	Drawn ASM / MM		Approved
	Parts no	30 - 1151	
	Print no	Sheet of sheets	

7.8 Configuration Bus Used with PAM Process Modules (Process Ctrl & Monitoring)

As an extension for medium size process control the PAM range of process modules (see table overleaf) can be used with the MCP system and configuration bus, interfacing to the CR80 via the WCA and WPU. Only a brief overview of the many possibilities available with the PAM product line is given here, please request the detailed PAM product documentation for further information.

The powerful standard range of PAM process modules includes 8 digital input, 7 digital output modules as well as single bit input and output. The analogue range include 8 channel input modules like 0-10V, 0-20mA, RTD temperature measurement as well as a 7 channel thermocouple module. The analogue modules are isolated against the Configuration bus as well as other input channels. Conversion is 8 bit dual slope to ensure problem-free operation even in noisy industrial environments. The MCP system and PAM modules are interconnected using the WPU (or WPC) configuration bus lines (Receive/Transmit) replacing configuration bus flat cable with twisted-wire 0,75 square mm installation cable. Thus the bus can be extended to a distance of up to 3000 meters. Signal communication as well as power distribution for the PAM modules go over the same two wires. The analogue modules which are of the dual slope integrating type need a 24V AC supply for their isolated front-ends. The 24V AC can be supplied locally or via an extra pair of wires in the bus cable.

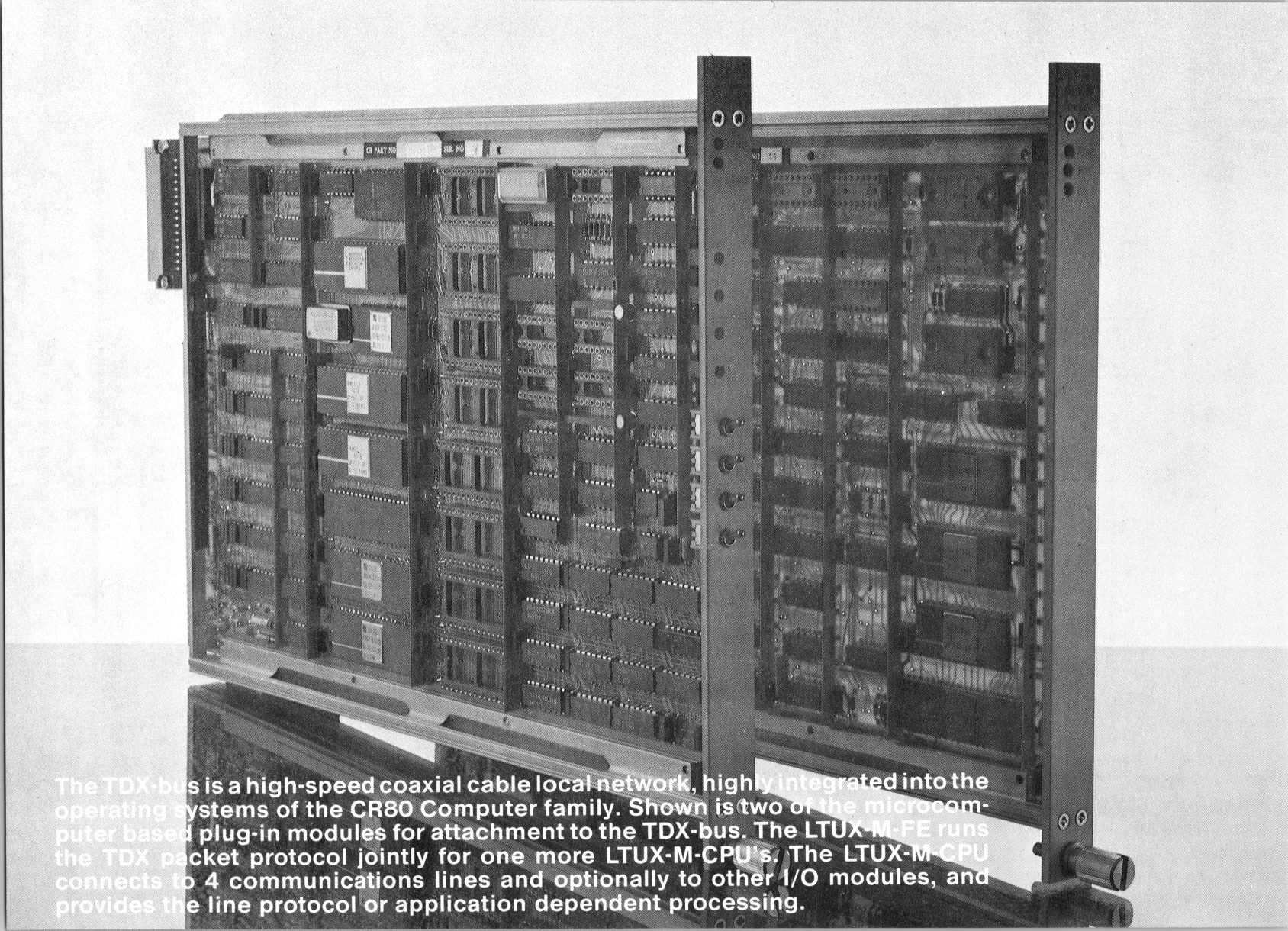
An FSK unit enables modules to be connected via APL telephone lines. The modules are polled from the MCP system, which during the poll cycle updates analogue and digital output modules and collects the analogue and digital input data.

In this way it is easy and inexpensive to upgrade a CR80 computer to do medium size process control like complete energy management for large buildings or factories just to mention one important application.

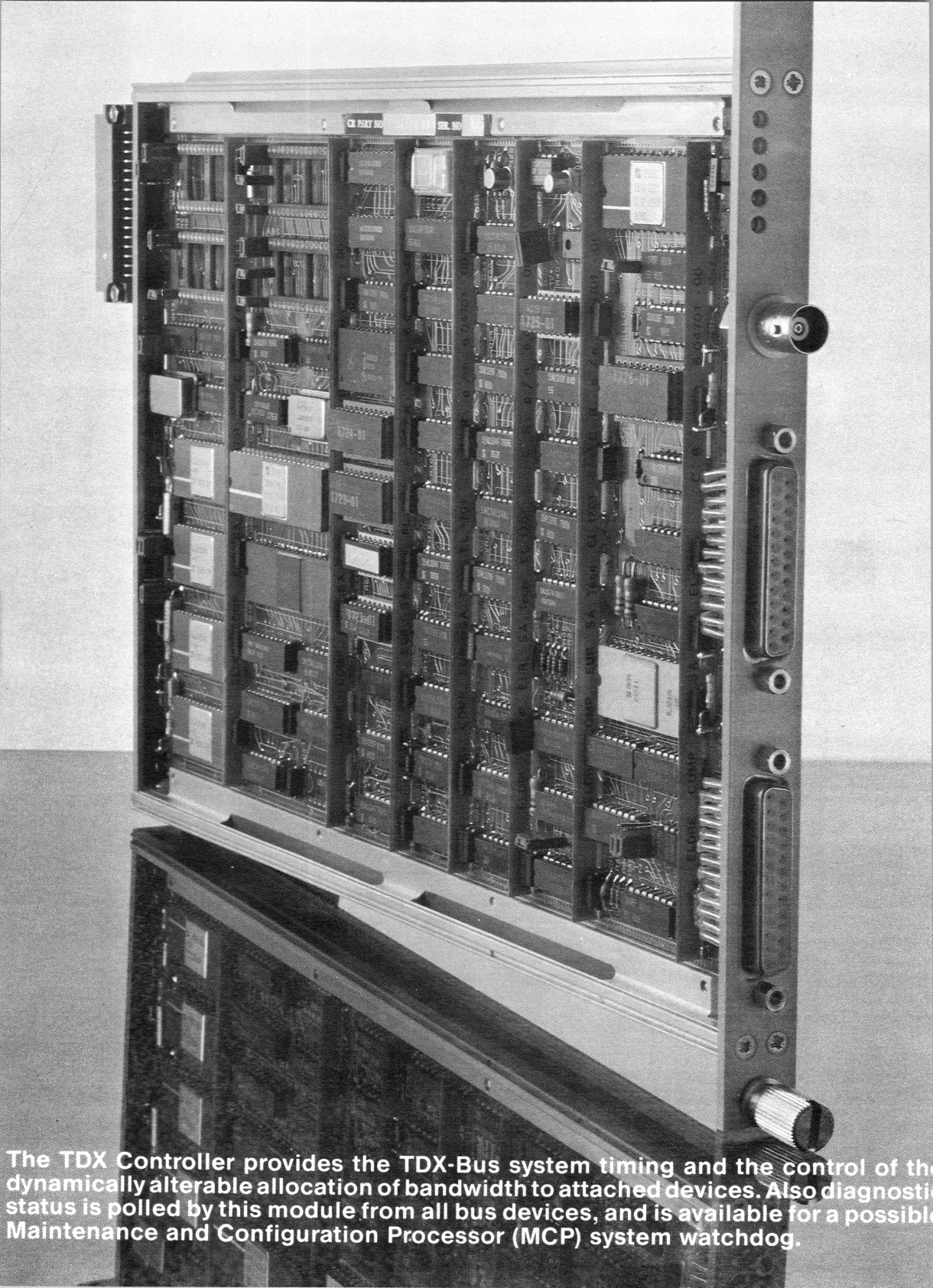
Type	Description
PAM 310	1 digital input (alarm)
PAM 320	8 digital inputs (alarms)
PAM 330	7 digital outputs
PAM 340	1 digital input + 1 digital output
PAM 345	4 digital inputs + 3 digital outputs
PAM 351	8 analogue inputs, 0-20mA
PAM 360	1 analogue output, 0-20mA
PAM 371	8 analogue inputs for Pt-100 sensors
PAM 372	Room temperature monitor with built-in sensor
PAM 372D	Room temperature monitor (PAM 372) + 2 digital inputs and 2 digital outputs
PAM 373	7 analogue inputs for thermo-elements
PAM 374	Temperature monitor, 1 channel excl. sensor + 2 digital inputs and 1 digital outputs
PAM 375	1 pulse frequency input
PAM 380	ID-card reader
PAM 382	Electrically operated door latch
PAM 385	ID-card without picture
PAM 390	PAM-FSK Tranceiver (APL Lines)
CMU 310	CMU, Alarm and Access, Turnkey Central Unit
CMU 320	CMU, Alarm, Turnkey Central Unit
CMU 330	CMU, Process, Turnkey Central Unit
PMU 340	Printer

Table 7.8a: PAM range of standard Process Control and Monitoring modules.

NOTES:



The TDX-bus is a high-speed coaxial cable local network, highly integrated into the operating systems of the CR80 Computer family. Shown is two of the microcomputer based plug-in modules for attachment to the TDX-bus. The LTUX-M-FE runs the TDX packet protocol jointly for one more LTUX-M-CPU's. The LTUX-M-CPU connects to 4 communications lines and optionally to other I/O modules, and provides the line protocol or application dependent processing.



The TDX Controller provides the TDX-Bus system timing and the control of the dynamically alterable allocation of bandwidth to attached devices. Also diagnostic status is polled by this module from all bus devices, and is available for a possible Maintenance and Configuration Processor (MCP) system watchdog.

8. TDX BUS SUBSYSTEM FOR THE CR80 COMPUTERS

8.1 Introduction

The TDX system (Telecommunication Data eXchange) is based on a high bandwidth serial coaxial cable pair interconnecting a range of TDX Devices, which makes up a high performance front-end local network highly integrated in the operating system of the CR80 minicomputer family (mapped and unmapped systems).

The TDX Local Network highlights easy connection of different data communication equipment and makes future extensions and changes of computer installations easy to do without having to move and install new cables.

The TDX Protocol is a full Error Detecting and Correcting distributed full duplex HDLC like protocol, and as all critical components can be dualized, a very fault tolerant system is achieved.

The TDX Local Network can be equipped with standard protocols for interfacing to different data communication equipment e.g. X25 LAP and LAP B, SDLC, BSC, TC500, CO2, CO3 etc., as well as custom defined protocols.
Attachment of distributed Process Monitor and Control Equipment is possible as well as optical links, crypto-equipment etc.
The following pages illustrates the versatility of the TDX system by applications in the areas of Communication and Process Control.

Based on the TDX Front End Local Network concept is the X-Net (see chapter 14) which is a stand-alone local area network especially designed for interconnecting different data communication equipment.

8.1.1 Front End Local Network Examples

8.1.1.1 CAMPS

CAMPS - Computer Aided Message Processing System is a Message Entry System based on the CR80 minicomputer with the TDX Network as a front end system.

CAMPS Centers communicates with other datanetworks and systems such as SCARS II (Strategic Command and Alert Reporting System) and ACE CCIS (Command Control Information System).

There are naturally high demands for reliability and security in the CAMPS system. These demands are met by the hardware, firmware and software as an entity.

The hardware is built to be fault tolerant, so that a breakdown of a subsystem causes another subsystem within the machine architecture to take over.

The DAMOS based software design includes automatic recovery and restart procedures.

The system architecture results in highly distributed processing, storage and terminal management. The major subsystems contain multiple CPU's and numerous microprocessors are distributed throughout the configuration.

Overleaf (Fig. 8.1.1.1-1) is shown the CAMPS system configuration in an overview.

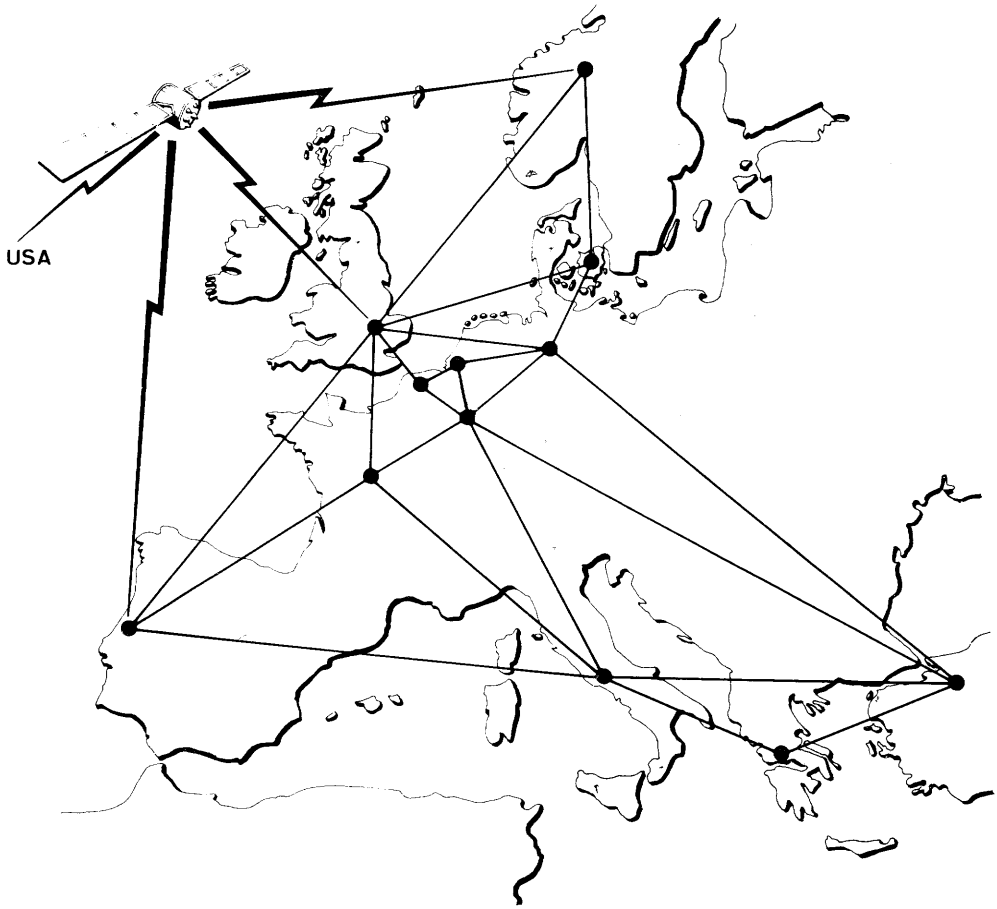
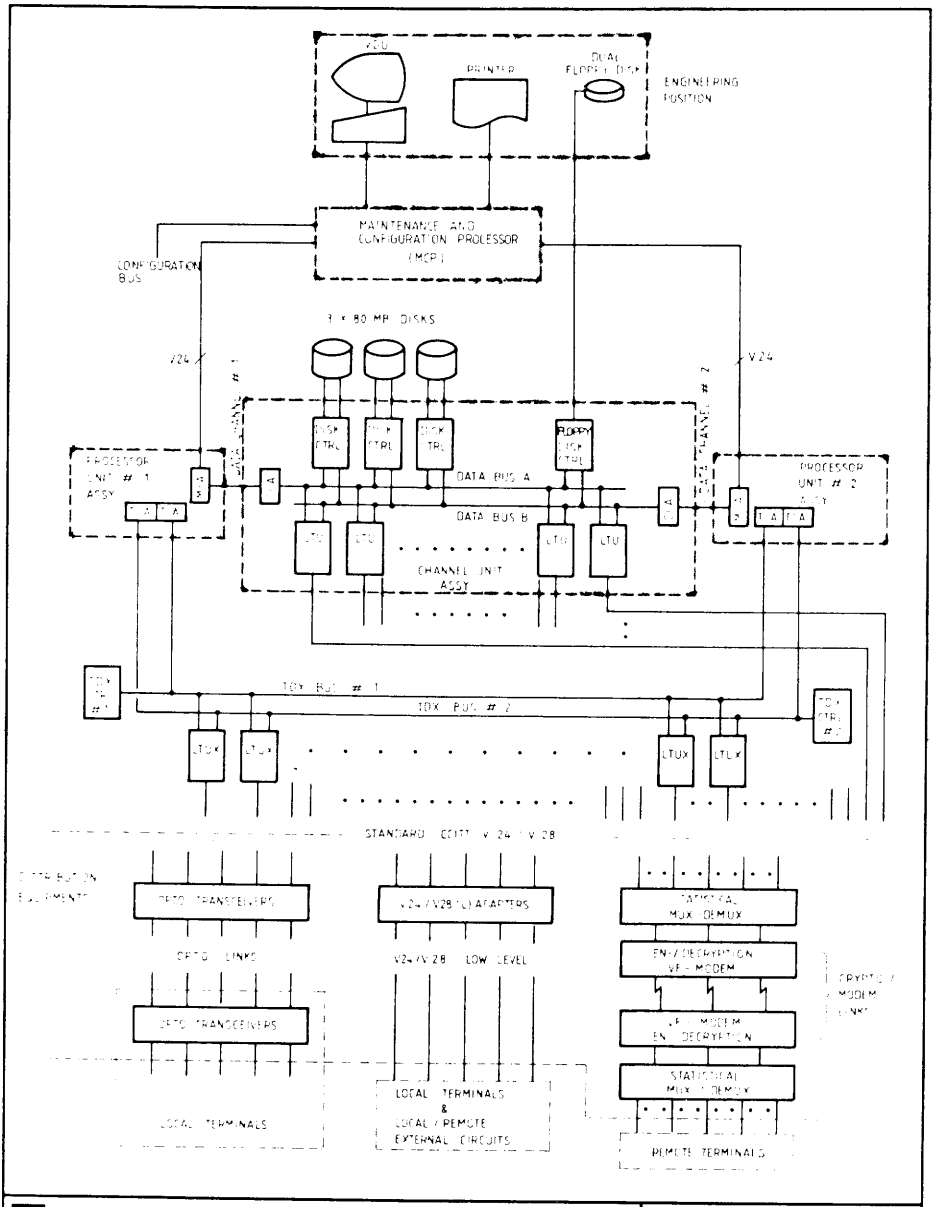


Figure 8.1.1.1-1
CAMPS NETWORK

A unique element of the configuration is the TDX system. In essence, the TDX handles entire complex of terminals and communication lines with a minimum attention by the central processors resulting in increased processor throughput.

Overleaf (Fig. 8.1.1.1-2) is shown CAMPS hardware configuration.



CHRISTIAN ROVSING A/S

CAMPS H/W Configuration

Fig 8.1.1.1-2

3 - 2694

8.1.1.2 FIKS

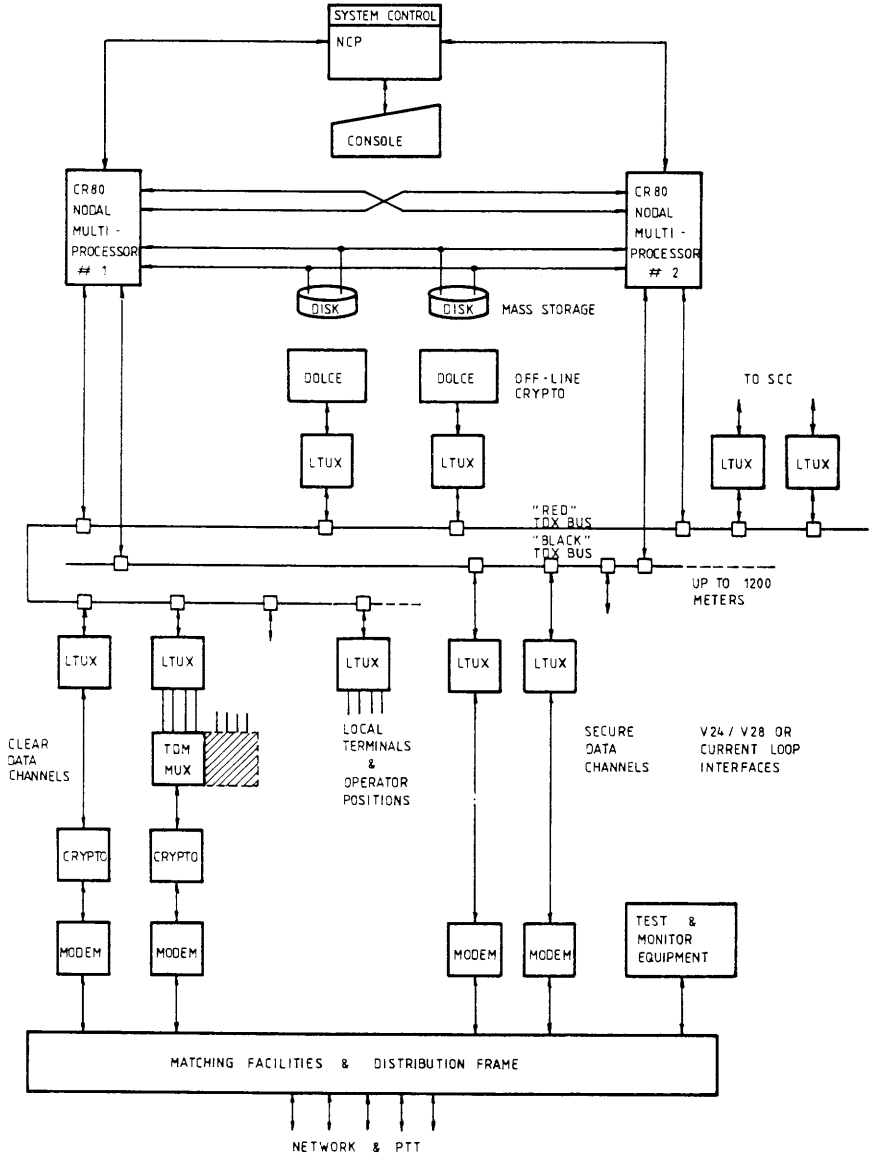
FIKS, the Danish Defense Integrated Communication System, is an integrated and fully automated message switching and data transfer communication system to be used by the Danish Armed Forces. It consists of a multinode network geographically distributed throughout Denmark. As initially structured, 8 nodes will be arranged in a grid configuration and interconnected via 18 full duplex trunks operating at 9.6 kilobaud.

Message and data traffic will be interchanged between users under control of computerized nodal switching centres. Message users at remote terminals are served through COMCENTERS, some of which are colocated at the nodes. FIKS is sized to handle a throughput of 25,000 messages per busy hour including messages entering the network, multiple distribution of messages, retrievals, service messages and a 25% allowance for growth.

Data users, continuous or discontinuous, will exchange information through FIKS. Typical data users are tactical data systems which relate to air defense, air traffic control, intelligence and command nets.

Data traffic rates vary from low speed 50, 100 and 300 baud, to medium speed 600, 1200 and 2400 baud.

The services in the whole network will be delivered from 18 TDX systems and 18 CR80 computers.


SYSTEM CONTROL
NCP
CONSOLE
CR80 NODAL MULTI - PROCESSOR # 1
CR80 NODAL MULTI - PROCESSOR # 2
DISK
DISK
MASS STORAGE
DOLCE
DOLCE
OFF-LINE CRYPTO
TO SCC
LTUX
LTUX
LTUX
LTUX
"RED" TDX BUS
"BLACK" TDX BUS
UP TO 1200 METERS
LTUX
LTUX
LTUX
LTUX
LTUX
CLEAR DATA CHANNELS
TDM MUX
LOCAL TERMINALS & OPERATOR POSITIONS
SECURE DATA CHANNELS
V24 / V28 OR CURRENT LOOP INTERFACES
CRYPTO
CRYPTO
MODEM
MODEM
MODEM
MODEM
TEST & MONITOR EQUIPMENT
MATCHING FACILITIES & DISTRIBUTION FRAME
NETWORK & PTT

8.1.1.3 DORA

The DORA computer system (see Fig. 8.1.1.3-1) will be used by the Danish Radio for their daily production and automated transmission of television news and sports broadcasts. The system consists of a dualized TDX system, which integrates 3 CR80 computers, 17 VDUs and some 40 various pieces of video equipment via parallel I/O ports analogue/digital and digital/analogue converters distributed within an overall distance 1-2 kilometers.

The aim of the system is to provide better and more efficient working conditions for the staff during the preparation, editing and transmission of the broadcasts. This is achieved in two ways:

- During the preparation and editing of the broadcasts a large amount of information of various kinds needs to be distributed among the studios and editorial offices. This can conveniently and safely be done via the TDX system and terminals connected to the computer from where information and commands can be called up and executed on-line, in real-time.

- Moreover, by letting the computer take over the critical functions encountered during the transmissions, the staff will be relieved from the more stressing part of the work.

The three computers are used for:

- Editing of broadcast programs from the attached videosources
- Real time control of video sources during transmission
- Facility booking.

Each interface connected to the TDX bus contains its own microprocessor. The division of control between these microprocessors and the CR80 is such that the intricate time critical control of the device is exercised by the microprocessor, whereas overall system control is carried out by the CR80.

DORA is an example of a complex process monitor and production control system, the successful implementation of which is dependent upon a system architecture with a carefully balanced distribution of processing power throughout the system.

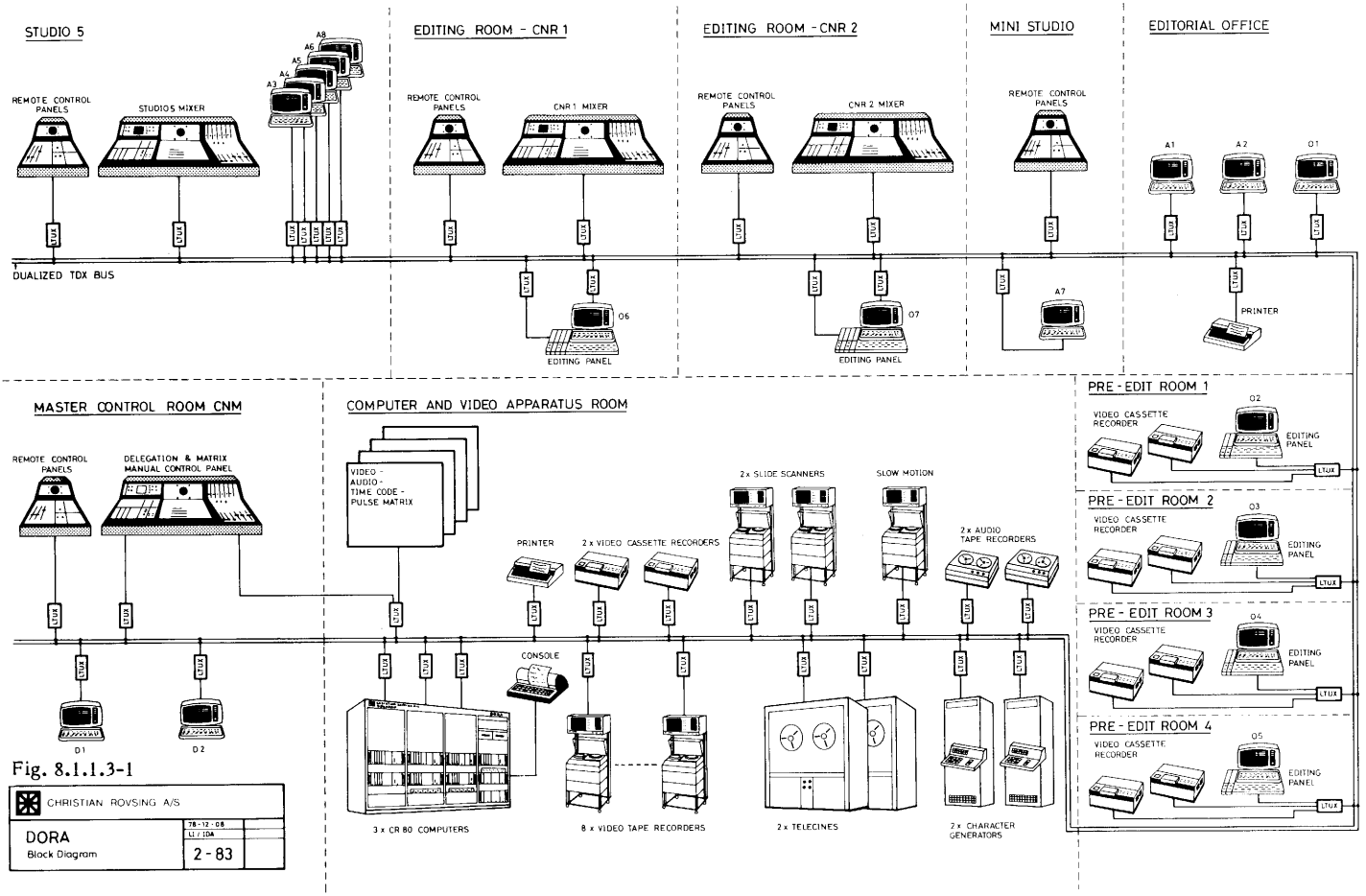


Fig. 8.1.1.3-1

CHRISTIAN ROVSING A/S		
DORA Block Diagram	78-12-08 LI/IDA 2-83	

8.2 TDX Front End Local Network - System Description

Below the main characteristics of the TDX Bus is given.

8.2.1 System Overview

• bitrate:	1.8432 Mbit/sec (optionally external clock).
• medium:	shielded twisted pair coaxial cable, base band signalling.
• Maximum cable length:	1.3 km without Amplifier/Branching Unit 2.50 km with Amplifier/Branching Unit
• topology:	branching routed tree
• data link layer:	distributed duplex HDLC like protocol
• fault tolerance:	dualization of all critical devices are possible.
• addressing:	each attached device has a unique address from 0 to 254 and each device has sub-addresses giving addressig possibilities on several hundred dataconnections each monitored by the protocol.

8.2.1.1 Topology and Configuration

The TDX system consists of a number of Units connected to the TDX-Bus. Each unit consists of one ore more Devices and each device has a unique TDX address. Devices can consist of several subdevices.

Different data terminal and data communication equipment can be attached to one Device. Devices are grouped in Domains. Each domain must be controlled by a CR80 Computer

The TDX System has following schematic configuration.

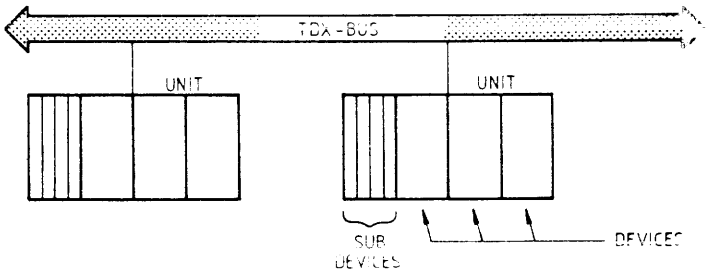


Fig. 8.2.1.1-1 TDX UNITS, DEVICES AND SUBDEVICES

TDX Units can be either a communication interface unit or a CR80 interface unit (STI/TIA).

TDX Devices are one of following types:

- STI/TIA (CR80 Interface)
- Controller (Synchronizing unit, has TDX-address 0)
- LTUX-S (interface between TDX-bus and 4 serial links)
- LTUX-M (interface between TDX-bus and 4-12 serial links optionally X-21 links; furthermore subdevices can be attached, see below).
  The LTUX-M is optionally available with an increased no. of internal datastreams; the addressing in the TDX system is like STI/TIA addressing - see 8.2.1.3

STI/TIA is in the device address range from 1-12, LTUX Devices is in the device address range from 13 to 254, DUMMY DEVICE is address 255 which must not be physically implemented on the TDX-Bus.

TDX subdevices are one of following types:

Front-End Subdevices:

- LTUX-M-FE (interfaces the micro-bus and the TDX-Bus - see below)

CPU Subdevices:

- LTUX-M-CPU (interfaces 4 serial links to the micro-bus)
- LTUX-M-CPU with X21 (interfaces 4 X21 links to the micro-bus)

Slave Subdevices:

- PIO-X (parallel I/O board interfacing to the micro-bus)
- ADC-X (analogue/digital converter board interfacing to the micro-bus)
- DAC-X (digital/analogue converter board interfacing to the micro-bus)

The subdevices are interconnected via the micro-bus and a device consisting of multiple subdevices has the following configuration:

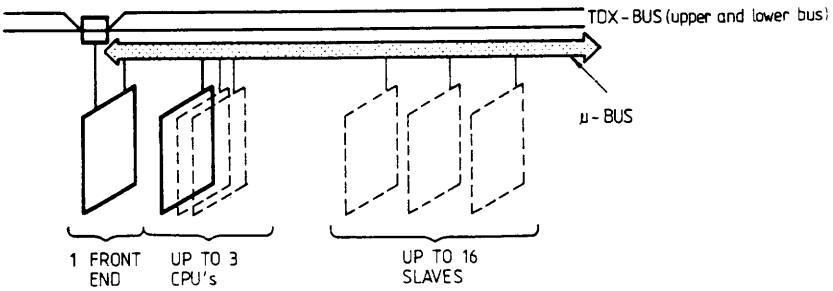


Fig. 8.2.1.1-2 SUBDEVICES CONNECTED VIA THE MICROBUS

All devices (except the controller) transmit on the upper bus and receive on the lower bus shown overleaf (Fig. 8.2.1.1-3).

TDX BUS:

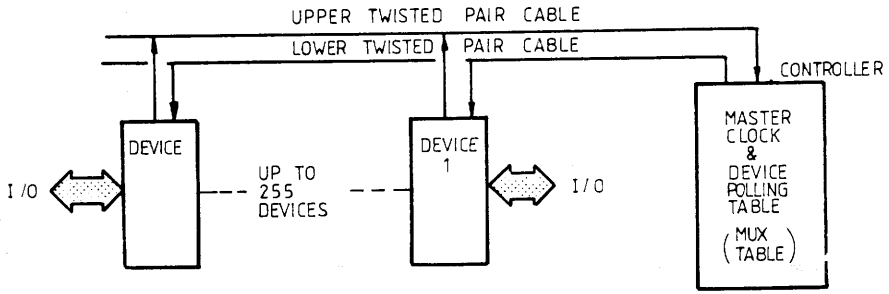


Fig. 8.2.1.1-3 TDX BUS

Each device has a number of internal logical data channels. Two logical data channels can communicate point to point with full TDX protocol. Two interconnected logical channels constitutes a Datastream.

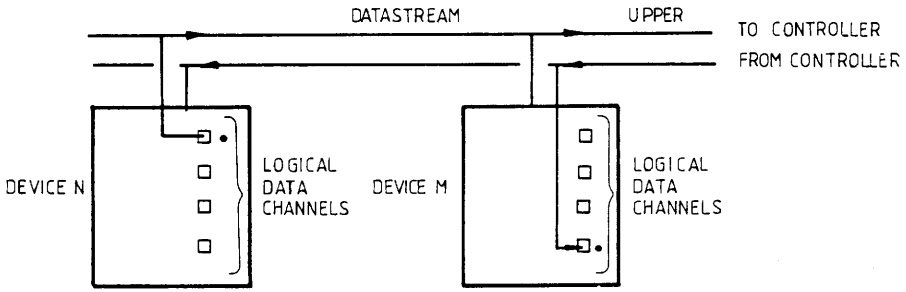


Fig. 8.2.1.1-4 DATASTREAM

Datastream can be established and dismantled by request from higher levels

Overleaf is found a typical TDX system configuration.

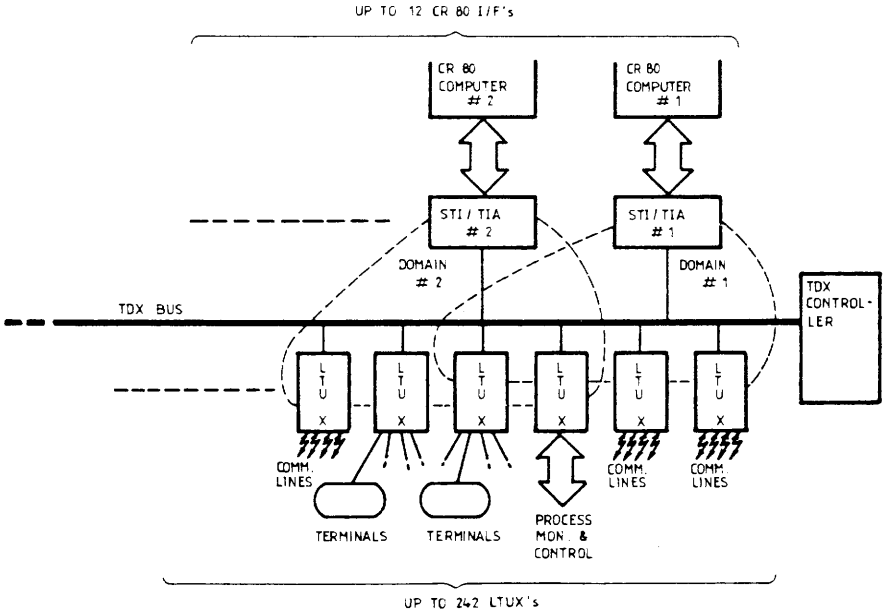


Fig. 8.2.1.1-5
DOMAIN PARTITION OF DEVICES
ON TDX LOCAL AREA BUS

8.2.1.2 Synchronization of TDX Traffic

The TDX Controller outputs a continuous bit stream of 1.8432 Mbit/sec on the lower bus. This stream is divided into 6400 time slots of 288 bit each per second.

Each time slot on the lower bus contains a standard HDLC frame with control information (5 bytes), DATA to be transferred (16 bytes), and CCITT-16 (2 bytes) cyclic redundancy check (CRC). The HDLC frame starts at the beginning of a time slot and takes up maximally 236 bit of the 288 bit in the time slot, the remaining bits being all "ONES" except from bit No. 240 to 248 when all "ZERO'es".

The TDX controller inserts as first byte in the HDLC frame on the lower bus a unique address from a MUX table (hereafter called MUX-No) which is scanned according to the required bandwidth.

FLAG	MUX	IDENTIFIER (CR - ID)			COMMUN-CATIONS BYTE	SEQ. NO. & BYTE-COUNT	16 - BYTE DATA FRAME	CRC	FLAG
		DATATYPE	HOST/MODE	DEVICE NO.					
(8)	(8)	(4)	(4)	(8)	(8)	(8)	(128)	(16)	(8)

Fig. 8.2.1.2 TDX FRAME FORMAT

All devices with their unique address on the TDX Bus look at the MUX No., and if address and MUX-No are identical this TDX Device has the use of the upper bus to transmit data at the end of the frame on the lower bus, provided that the lower bus CRC shows no errors, thereby ensuring that only one device will transmit on the upper bus at any time.

8.2.1.3 TDX Bus Frame Format and Addressing

The two TDX Bus frame formats are shown overleaf (Fig. 8.2.1.3-1) for lower and upper bus. They only differ on two points, namely that the lower bus includes MUX No. M in its data field, and the upper bus has an ABORT byte (all ONE'S) as preamble.

Both upper and lower bus frames are standard HDLC FRAMES with bitstuffing in order to have a unique synchronization byte (FLAG = 01111110). Bitstuffing is done between the beginning FLAG and ending FLAG, each time five ONE'S are met in the data stream, a zero is inserted. The zero is then removed at receiving end, thereby restoring the original data.

The bit stuffing (depending on number of subsequent ONE'S in data stream) makes the length of frames variable - on the lower bus between 200 and 236 bit, and on the upper bus between 200 and 235 bits.

LOWER BUS FORMAT

Following the flag each byte has the following function:

Byte No. 1, Mux No. M:
This byte is inserted by the Controller to signal that the station with the unique address M is to transmit on the upper bus at the end (actually at bit count 241) of the frame on the lower bus.

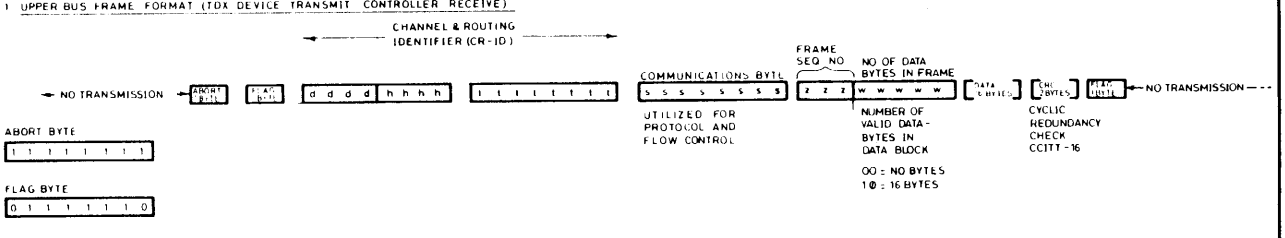
CHRISTIAN ROVSING A/S

TDX BUS
Frame Format
Upper & Lower Bus

Fig. 8.2.1.3-1

Scale
Date 81-11-17
Drawn ASM/GFL
Parts no
Print no
Sheet 1 of ... sheets
Approved
Issue
Date 81-11-17
Approved

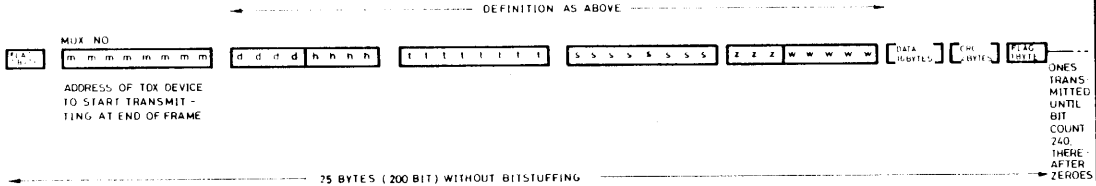
1 UPPER BUS FRAME FORMAT (TDX DEVICE TRANSMIT CONTROLLER RECEIVE)



25 BYTES (200 BIT) WITHOUT BITSTUFFING

235 BIT WORST CASE WITH BITSTUFFING

2 LOWER BUS FRAME FORMAT (CONTROLLER TRANSMIT, TDX DEVICE RECEIVE)



25 BYTES (200 BIT) WITHOUT BITSTUFFING

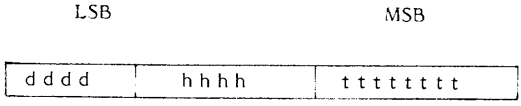
236 BIT WORST CASE WITH BITSTUFFING

Byte No. 2 and 3 (CR-ID)

Byte no. 2 and 3 are used in the addressing scheme of the TDX-System and they are named CR-ID (Channel Routing Identifier).

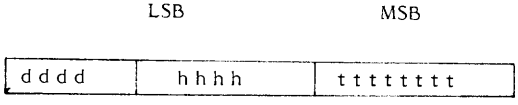
Four different interconnections of TDX Devices are available

A. From a LTUX to a LTUX. The CR-ID has following format:



h: E*
d: Datatype (O-F)*
t: Address of destination Device

B. From a LTUX to a STI/TIA. The CR-ID has following format:



h: STI/TIA Address O-C* (Destination)
d: Datatype (O-F)*
t: Address of Source Device

* Hexadecimal

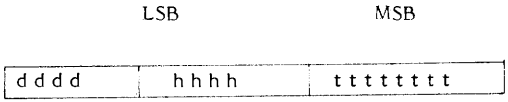
C: From a STI-TIA to a LTUX. The CR-ID has following format:

h: F*

d: Datatype (O-F)*

t: Address of destination Device

D: From a STI/TIA to a STI/TIA. The CR-ID has following format:



h: STI/TIA Address O-C* (Destination)

d: Datatype (O-F)* (Datastream no.)

t: STI/TIA Address O-C* (Source)

E: Dummy frames

1) from devices (upper bus): CR-ID:=FFFF*
(preferably the remaining header bytes and the data bytes shall contain zeroes)

2) from Controller (lower bus): CR-ID:=0000*
remaining bytes in frame shall contain all zeroes.

* Hexadecimal

Byte No. 4, COMMUNICATION BYTE:

This byte is used for the TDX Packet protocol control.

Byte No. 5, NO. OF DATA BYTES IN FRAME:

BITS 5-7: inserted by the originating station of the frame to sequentially number frames, modulo 8 in a communication with a specific receiving station, in order that the receiving device can request retransmission when a jump in SEQ. NO. occurs. This is used for Error Detection and Correction, as in following example: A CRC error occurred in frame with SEQ.NO. n; this frame was therefore thrown away by the receiving device. At correct recieval of frame with SEQ. No. n+1, the receiving device detects an increase in SEQ. No. of 2, from previous correctly received SEQ. No. n+1. The Link-level of the receiving device accordingly requests retransmission.

BITS 0-4:

As the number of bytes in the DATA field transmitted is fixed at 16 (128 bits), the five least significant bits of this byte indicate the number of actual DATA bytes in a possible partly filled data field (hexadecimal: ØØ - 1Ø, decimal: Ø - 16)

Byte No. 6 to Byte No. 21, DATA:

This is the information field containing data to be communicated from originating station (source) to receiving station (destination).

Byte No. 22 & No. 23, CRC:

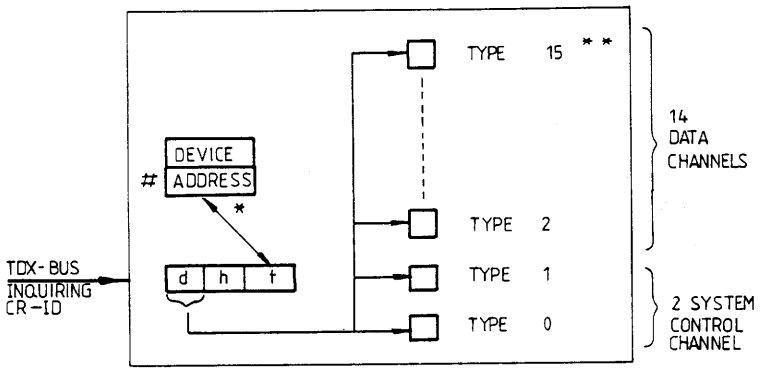
The Cyclic Redundancy Check (CCITT-16) is inserted by the originating station and utilized by the receiving station to verify the correctness of the total received frame.

Trailing "ONE'S" after Frame

Following the frame, the controller inserts "1"s until bitcount of time slot reaches 241 and thereafter "0" until end of time slot. The change from 1 to 0 is used for starting transmission on the upper bus by the selected TDX Device (MUX No.) at an exact point in the time slot.

8.2.1.3.1 LTUX and STI/TIA Channel Separation

Each TDX-Device uses the CR-ID to separate the ingoing traffic in Channels. (There is not necessarily correlation between Channels and physical datalinks). The following figures (8.2.1.3.1-1 and 8.2.1.3.1-2) show the channel separation performed by a LTUX and a STI/TIA device.

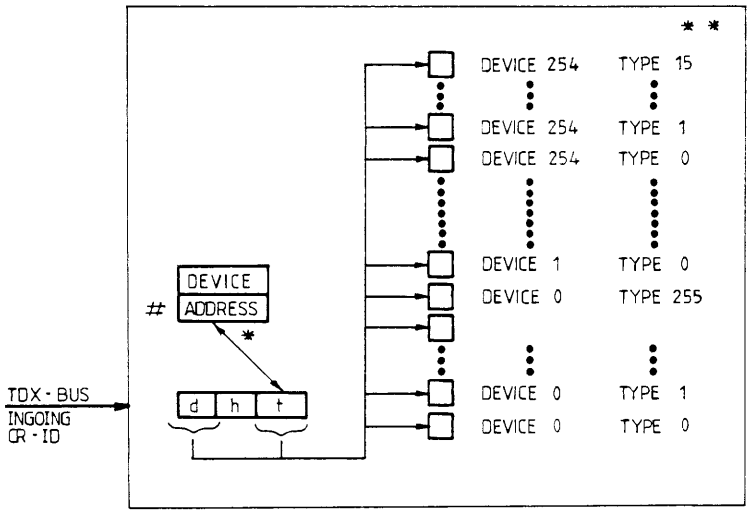


* Global device address identification

** LTUX-S has only 10 channels

Fig. 8.2.1.3.1-1 LTUX SEPARATION

Fig. 8.2.1.3.1-2
STI/TIA SEPARATION



* Global address identification

** The addressing scheme allows 4096 channels, but the number of active channels is limited to 180 ea. STI (valid for STI with 32K Central RAM).

The STI/TIA has addressing space for 2 system control channels per Device, but as the traffic on the channels is very low, and the protocol on these channels is a datagram protocol, only two system control channels are used on the same time in the STI-TIA.

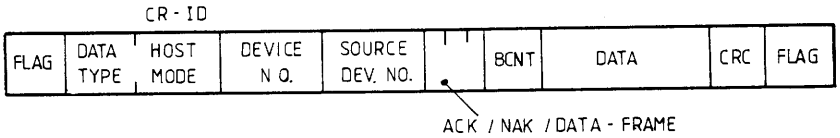
8.2.1.4 Data Transfer

Two different forms of data transfer take place in the TDX-System:

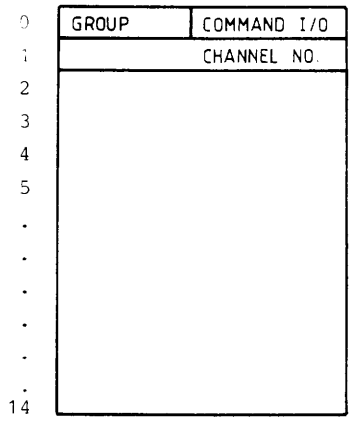
- datagram messages, which are transferred on channels with datatype 0 or 1.
- datapackets, which are transferred between two open channels (except channels with datatype 0,1); two such channels constitute a Datastream.

- Datagram

Datagrams on channel 1 are sent to the TDX-controller as bandwidth requests or diagnostic. Datagrams on channel Ø (between LTUX's or/and STI/TIA's) are restricted to be single TDX-frames on max. 15 bytes with the following format:



The Flag, CR-ID, BCNT and CRC are described in 8.2.1.3. The Source dev. no. identifies the device to where an ACK/NAK must be routed.
The ACK/NAK bits are placed in the SEQ. no. field of the ordinary frames. The 3 bits are used to give acknowledge, not acknowledge, and indicate if the frame contains data. This information is used at protocol level.
The DATA-contents is read by an interpreter. The format of the DATA-field is as follows:



The GROUP-no. must be 0 or 1 for commands to the system interpreter. The channel no. defines one of 16 channels that is related to the command.

The data field must be setup individually for each command.

Following command-IDs are available:

Group, Commd. ID	Operation	Data
0,1	Open TDX-channel	CRID, Send -and Receivetimer
0,2	Close TDX-channel	
0,3	Read Memory	Bytecount, Address
0,4	Read Diagnostic	Bytecount, Address
0,5	Write Memory	Bcnt, adr, memory-data
0,6	Write diag.	Bcnt, Adr, memory-data
1,X	Command response	completion code

When a command received from the TDX-bus is executed by the Local Interpreter, it sends a command response packet to the remote device. The completion code Ø indicates a successfully completion.
Code 2ØH is responded to an illegal command.

Datagrams with group.no. different from 0,1 are moved transparently to the ingoing- and outgoing queues of channel Ø. The source-device no. is inserted as byte 15 for use by higher levels.

- Datapackets

Datapackets are transferred between two open channels. An open channel is a channel with information about the address of the remote channel, whereby source CR-ID is needless information in the packet.

Datapackets consists of one or more TDX-frames which are encapsulated and decapsulated under full error detecting and error correcting protocol (see section 8.3).

8.2.2 TDX System - Functional Description

The TDX System is a dedicated Front-End Network for the CR80 minicomputer and is a highly integrated part of the I/O System in both mapped and unmapped systems.
The TDX System transfers Logical Data Units (LDU) between CR80 minicomputers and/or LTUX, see fig. 8.2.2-1 below.

The LDU consists of Packets which are a TDX retransmission unit. A packet consists of one or more Frames which are a transmission unit in the TDX system. Prior to sending a LDU, the datastream between two datachannels must be created via datagrams on the system control channel. After transfer of one ore more LDU's the datastream can be dismantled, whereby a dynamic configuration scheme is achieved.
Overleaf (Fig. 8.2.2-2) is shown an example of LDU segmenting, and in the following is given a detailed description of the data formats and dataflow in the TDX system.

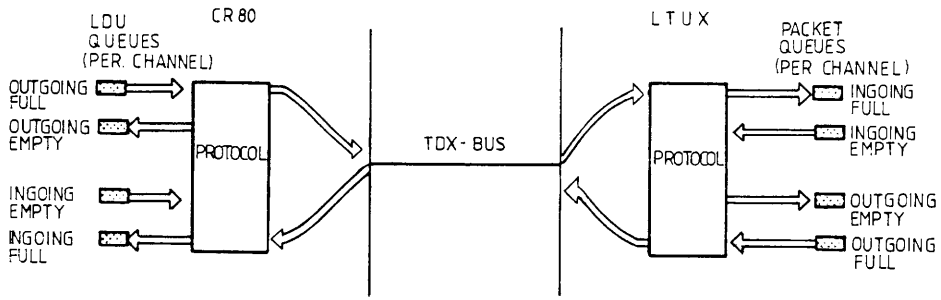


Fig. 8.2.2-1 LOGICAL DATA UNITS (LDU)

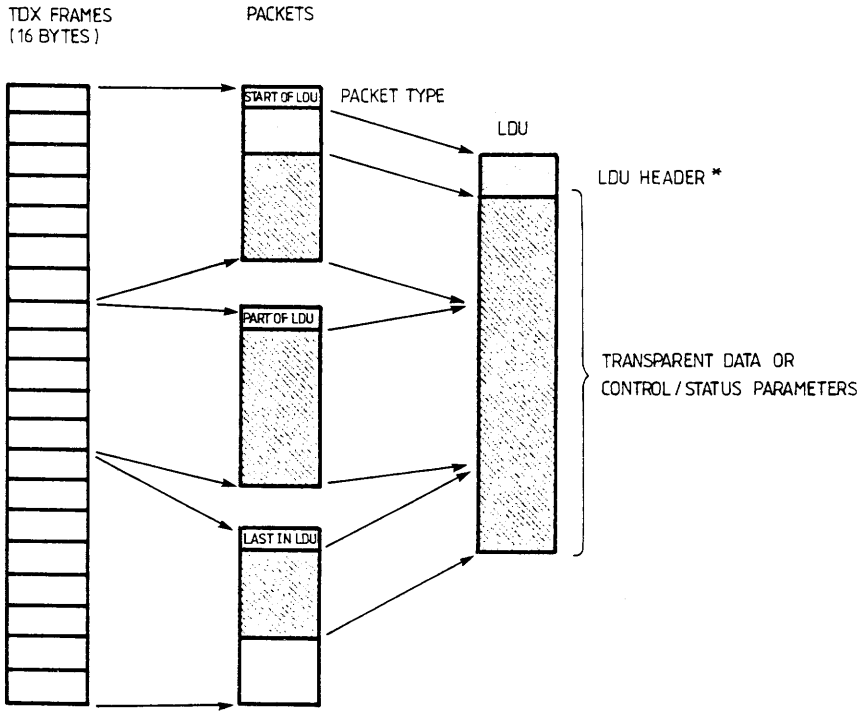


Fig. 8.2.2-2 LDU SEGMENTING

* The LDU header contains either a LDU sequence number (in case the LDU is built of packets of type "data") or a control/status id (in case the LDU is built of packets of type "control/status").

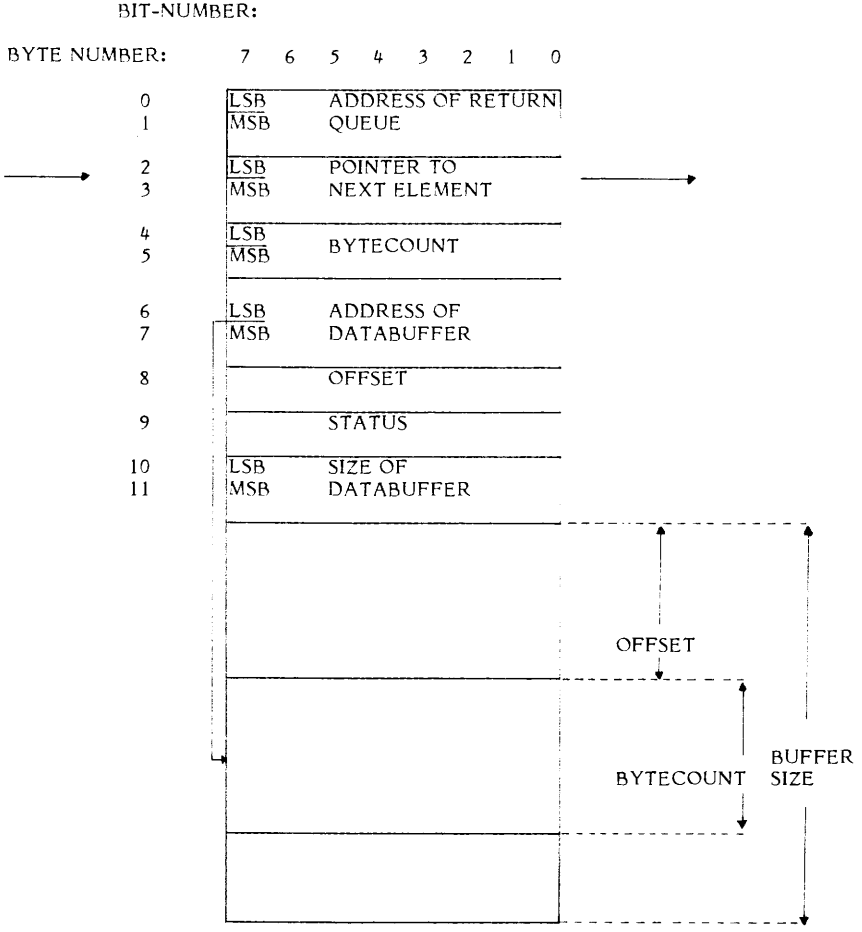


Fig. 8.2.2-3 FORMAT OF PACKET USED IN LTUX

Data Communication Format

The following information types are available for the application programmes in the LTUX.

Ingoing full:

1. Data
   The information is grouped into Logical Data Units (LDU'S) each consisting of a set of buffers and identified through an unambiguous sequence numbering. The LTUX must respond to all Logical Data Units delivered by the CR80 with a completion status for the transmission.

2. Control
   The following control commands are implemented:

   1. Input request
      This command gives the LTUX the permission to deliver the next received Logical Data Unit to the CR80. The LTUX must answer all input requests either by a delivery of the requested Logical Data Unit or by a status message.

   2. Cancel input request
      This command directs the LTUX to delete an identified input request from the internal queue. If the input request is being processed when the cancel is received, the processing is terminated. If the processing of the input request has been completed when the cancel is received, no action is taken.
      If the handling of the input is actually influenced by the cancel command, this is reported in a status message identifying the input request.

3. Cancel output

   This command directs the LTUX to delete an identified Logical Data Unit from the transmission queue. If the LDU is being transmitted when the cancel is received, the transmission is terminated. If the transmission of the LDU has been completed when the cancel is received, no action is taken. If the transmission of the LDU is actually affected by the cancel command, this is reported in the completion status for the LDU.

4. Report status request

   This command causes the LTUX to report status to the CR80 in a status buffer. Status requests are able to distinguish between different types of status specified by a parameter.

5. Open Channel

   This command directs the LTUX to initialize the channel. A number of user defined parameters may be included in the command.

6. Close Channel

   This command directs the LTUX to reinitialize and close the channel.

Outgoing full:

1. Data

   The information is grouped into Logical Data Units (LDU's) each consisting of a set of buffers with an unambiguous identification corresponding to a related input request.

2. Status

   The following status messages are implemented:

   1. Transmission status

      This message reports the completion status of a Logical Data Unit delivered by the CR80 for transmission. The following completion codes may be reported:

Transmission OK
Transmission cancelled
Transmission error

Reception of the transmission status is an asynchronous event.

2. Reception Status
   This status is a completion code for received LDU's. When the device-handler receives 'Reception status' the LDU is terminated prematurely. This status is transferred asynchronous.

3. Interrupt
   This message reports asynchronously an user defined event (e.g. the reception of a 'break' from the communication line).

4. Channel Status
   This message reports the status specified by the command 'Report Status request'.

Buffer-Type

The first byte in the buffer data field contains information about which part of the LDU the buffer contains.

The buffer-type specifies:

bit 1,0

11	-	the buffer is an entire LDU
01	-	the buffer is start of LDU
00	-	the buffer is part of LDU
10	-	the buffer is the last of a LDU

bit 2

0	-	transparant data
1	-	control/status parameters

LDU-Header

The first byte in a LDU is denoted LDU-Header. The contents of the LDU-header depends of the buffer type.

Transparant Data

In case the buffer is 'start of LDU' or 'entire LDU', the buffer includes a LDU-Header byte with a LDU sequence no. (0-255) in front of the LDU data.

Control/Status Parameters

In case the buffer is 'start of LDU' or 'entire LDU', the buffer includes a LDU-Header byte with the control/status type.

The following control commands are available:

Code	Control-types	Parameters (P_1,P_2..)
0	input request	LDU seq. no.
1	cancel input request	LDU seq. no.
2	cancel output	LDU seq. no.
3	report status request	status type
4	open channel	user defined parameters
5	close channel	N.A

The following status types are available:

Code	Status-types	Parameters (P_1,P_2..)
0	transmission status	LDU seq. no., state
1	reception status	LDU seq. no., state
2	interrupt	
3	channel status	state, user defined parameters

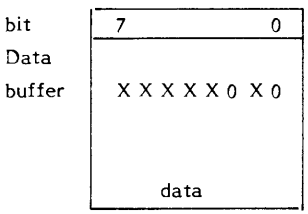
Control commands flow in the direction from the CR80 towards the LTUX and status information flows from the LTUX to the CR80.

Specification of States

Status Type	State	Code
Transmission status	TX OK	0
	TX cancelled	1
	TX error	2-255
Reception status	RX error	1-255
Channel status	Status type	0-255

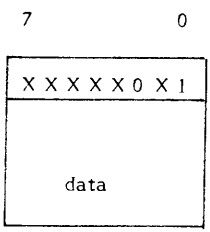
Examples:

1:



Data are transparent data in the middle or end of a logical data unit (LDU).

2

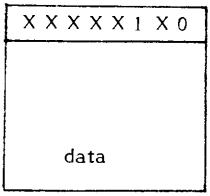


buffer type

LDU seq. no.

Data are transparent data starting a LDU or an entire LDU.

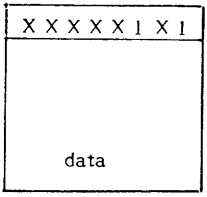
3:



buffer type

Data are control/status parameters in the middle or ending of a LDU.

4:



buffer type

command/status type

Data are control/status parameters starting a LDU or an entire LDU.

8.2.2.1 TDX Controller

Introduction

The TDX Controller is designed to control and monitor a single or dualized TDX system and perform the following tasks:

- Synchronize communication on the upper bus by inserting a MUX-No. in the HDLC frame on the lower bus.
- Answer a bandwidth request and allocate bandwidth according to the request.
- Poll and appended devices to collect diagnostic information.
- Communicate with a Watchdog (in a dualized TDX Controller Configuration).
- Select one of two upper buses to optimize performance in a dualized bus sytem.

All frames received from the "upper" bus are transmitted on the "lower" bus delayed one frame. Only if the CRC of a received frame is not correct or if the frame is destinated to Host No. Ø (the Controller itself), the frame will not be swapped to the transmitter buffer. When a received frame is destinated Host No. Ø, it is loaded to the controller processor which is managing the Mux table. The received frame may contain a request for a changed bandwidth to a given TDX device (BW-request).

The synchronization is achieved by inserting as second byte int he HDLC frame on the lower bus, a device No. taken from a Mux table (MUX No.) that is scanned according to the speed level assigned to each device of the TDX system.

FLAG	MUX	IDENTIFIER (CR-ID)			COMMUN-CATIONS BYTE	SEQ. NO. & BYTE-COUNT	16-BYTE DATA FRAME	CRC	FLAG
		DATATYPE	HOST/MODE	DEVICE NO.					
(8)	(8)	(4)	(4)	(8)	(8)	(8)	(128)	(16)	(8)

Functional Description

All devices with their unique device No. on the TDX-BUS look at the Mux No. byte, and if it is identical to its device No., this device has the use of the upper bus, to transmit data, at the end of the frame on the lower bus, provided that the lower bus CRC check shows no errors. This ensures that only one device will transmit on the upper bus at any time.

The bandwidth allocation is determined by the Mux table which is changeable (dynamic). A request for a changed bandwidth to a specified device received on the upper bus is accepted if the system bandwidth is large enough or rejected if the system bandwidth is too small. The answer (ACK, NACK) is sent to the requesting device, when a free time slice occurs on the lower bus. The dummy device FF (which is inserted in the MUX table, to allow the Controller access to the lower bus in the following frame) has a minimum bandwidth on 100 bit/sec. giving a free time slot to answers at least every 1.28 sec.

The diagnostic information is collected by polling each device appended to the system. If an answer is not received within 4 scans in the Mux table, a retransmission is executed. After three requests not answered, the device is perceived as not appended to the system.

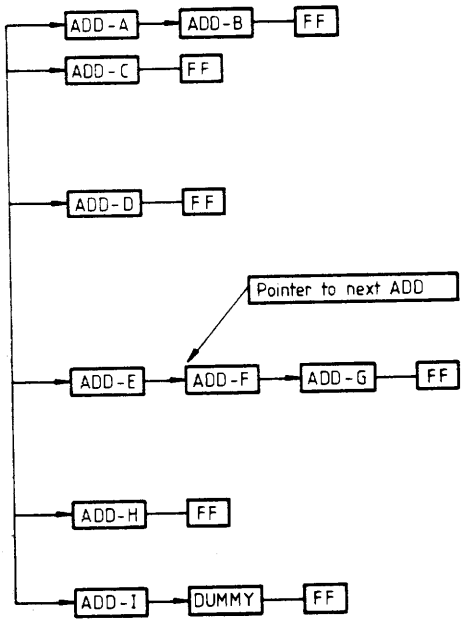
The upper bus select feature is achieved by counting errorfree received frames from the upper bus (both frames destinated to the controller and frames swapped to the lower bus) each time the Mux table has been read completely. If this count is less than 1:4 of the previous count, the bus is switched. This implies that one device may be removed every 1.28 sec. without changing buses, but removing a number of devices instantly causes a switch of upper buses.

Bandwidth Allocation

The bandwidth allocated to each device in the TDX system is determined by the MUX-table. This table is configured as shown below with 14 speed levels.

Bandwidth (b/sec)	Speed level
819200	14
409600	13
204800	12
102400	11
51200	10
25600	9
12800	8
6400	7
3200	6
1600	5
800	4
400	3
200	2
100	1

MUX-Table Configuration (Example)



ADD: Device-Address (1-254)

DUMMY: Device-Address 255 (dummy device)

FF: Flip-Flop used to implement the correct average bandwidth.

Each time a frame is transmitted on the lower bus, a device-address is read from the MUX-table in the location given by the "pointer to next ADD" and used as MUX-No. The "pointer to next ADD" is updated to the next "ADD", which is either the next in the chain or - in case it was the last "ADD" in the speed level chain - one of two possibilities: the first "ADD in the highest speed level or the first "ADD" in the next lower speed level dependent on the state of the FF, which is complemented by each test. The MUX-table shown on previous page will give following bandwidth-allocation (please notice that not used speed levels are not encountered).

AB - ABC - AB - ABCD - AB - ABC - AB - ABCDEFG
AB - ABC - AB - ABCD - AB - ABC - AB - ABCDEFGH
AB - ABC - AB - ABCD - AB - ABC - AB - ABCDEFG
AB - ABC - AB - ABCD - AB - ABC - AB - ABCDEFGHI (DUMMY)

This gives a relative bandwidth on following:

Address	Relative Bandwidth
A,B	32
C	16
D	8
E,F,G	4
H	2
I, DUMMY	1

This entire cyclus is repeated continously and is called a scancyclus.

A complete scancyclus has a duration of:

$$T_{scan} = \sum_{Vsp=1}^{Vsp=14} 2^{Vsp-1} \cdot NO_{Vsp} \cdot 156 \text{ usec.}$$

where Vsp is the virtuel speed level, which is the speed level achieved by compressing all active speed levels to be in sequence from 1 to the actual maximum (in the example on the previous page Vsp = 1,2,3,4,5,6). NO_{Vsp} is the number of device addresses on the actual speed level.

The average bandwidth of a device is expressed by

$$BW_{average} = \frac{2^{asp-1}}{\sum_{Vsp=1}^{Vsp=14} 2^{Vsp-1} \cdot NO_{Vsp} \cdot 156 \text{ usec.}}$$

where asp is the actual speed level (virtuel) of the device considered.

As an example the average bandwidth of the device with address D is found (1 frame has a duration of 156 usec):

$$BW_{average} = \frac{8 \text{ frames}}{(2^0 \cdot 2 + 2^1 \cdot 1 + 2^2 \cdot 3 + 2^3 \cdot 1 + 2^4 \cdot 1 + 2^5 \cdot 2) \cdot 156 \text{ usec}}$$

$$\approx 493 \text{ frames/sec}$$

Notice: The bandwidth is not equally distributed and to use the above expression for $BW_{average}$ memory chips with access-times less than 250 nanosec. must be used.

8.2.2.2 STI/TIA

Introduction

In the range of TDX-devices the STI makes the high performance end interfacing of a CR80 minicomputer to the TDX-bus system.

The STI is a high bandwidth device which is able to interface to other STI or LTUX connected to the TDX-bus. It may address 4096 logical lines through the TDX-system, and each line may have bandwidth allocated individually. The current implementation of STI-handler and STI serve up to 180 channels running actively in parallel. Through the STI the CR80 minicomputer is able, dynamically, to establish and dismantle all logical channels originating from and belonging to the Domain of the STI. The CR80 is also able to make dynamically change of the bandwidth-assignment on the TDX-bus.

By connecting to a STI several TIAs and SBAs it is possible to interface the CR80 with both TDX-buses and SUPRA-buses via the same STI. The maximum total number of TDX-buses and/or SUPRA-buses connected to a single STI is limited to 8. The STI will be addressed with the same HOST-number on each bus in a given configuration. For details of interfacing to the SUPRA-bus (S-net) please refer to chapter 2.

The STI serves the TDX-packet protocol, which guaranties errorfree transmission of data.

Functional Description

The STI is based on 2 processors:

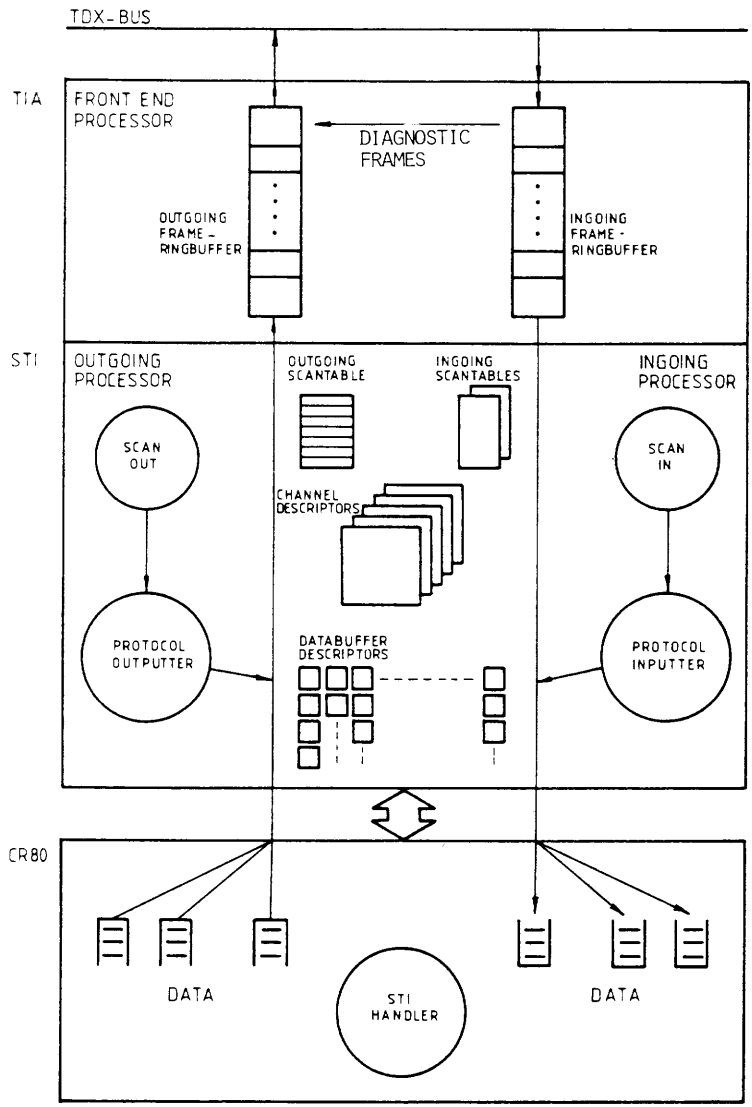
- The Ingoing processor, which moves data from the front end (TIA or SBA) to the CR80 memory. The data-traffic is controlled by the TDX-packet protocol. As the data is divided into may logical channels this processor also demultiplexes traffic coming from the TDX-bus. Data delivered to the CR80-computer is by the TDX packet protocol ensured errorfree. When a complete packet is received it is reported to the STI-handler by chaining the related data-buffer descriptor (DBD) into the ingoing completion queue. Transmission-errors, which are unrecoverable by the protocol are reported as completion codes in the DBDs.

- The Outgoing processor, which moves data from the CR80 memory to the Front end. Data sent to the TDX-bus is controlled by the outputter-part of the TDX packet protocol. Beside data transfer and protocol, this processor also scans all the logical channels set up by the CR80 computer, and multiplexes their data into one single stream delivered in the outgoing front end ringbuffers. When a packet is correctly transmitted, it is reported in the outgong completion queue. Transmission-errors, which are unrecoverable by the protocol are reported as completion codes in the DBDs.

Overleaf (Fig. 8.2.2.2-1) is shown the dataflow through the STI and the datastructures in the central-RAM, which is shared between Ingoing Processor, Outgoing processor and the STI-handler.

Fig. 8.2.2.2-1

DATAFLOW OVERVIEW IN THE STI



afs 8

Logical Data Units

The CR80 application or system programs request transfers through the TDX- or SUPRA-bus in data-blocks called Logical Data Units (LDUs). The STI-handler choppes up the LDUs into packets, which are delivered to the STI by databuffer descriptors (DBDs). The TDX protocol resident in the STI transmits the packets as a number of frames and accumulates received frames into packets.

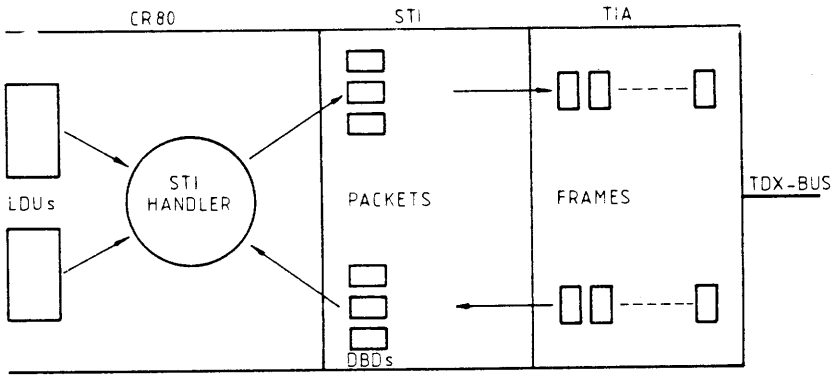


Fig. 8.2.2.2-2 CR80, STI/TIA INTERFACE

The STI handler returns a LDU-transmission as errorfree, when all the related packets are successfully transmitted and returned from the STI-protocol. The size of the LDUs is not limited, but with the current implementation of the STI-handler and STI packet sizes must be less than 64Kbytes. For a given TDX-channel the maximum packet size is defined as a parameter at channel creation-time.

Performance and Limitations

The required average timeconsumption (for STI-processing) pr. TDX-frame is shown overleaf (Fig. 8.2.2.2-3 and 8.2.2.2-4) as a function of number of frames in the transferred TDX-packet. The access-time to the TDX-bus is considered as neglegible. The times may be reduced with app. 20% by using memory chips with access-times less than 250 nanosec. in the STI.

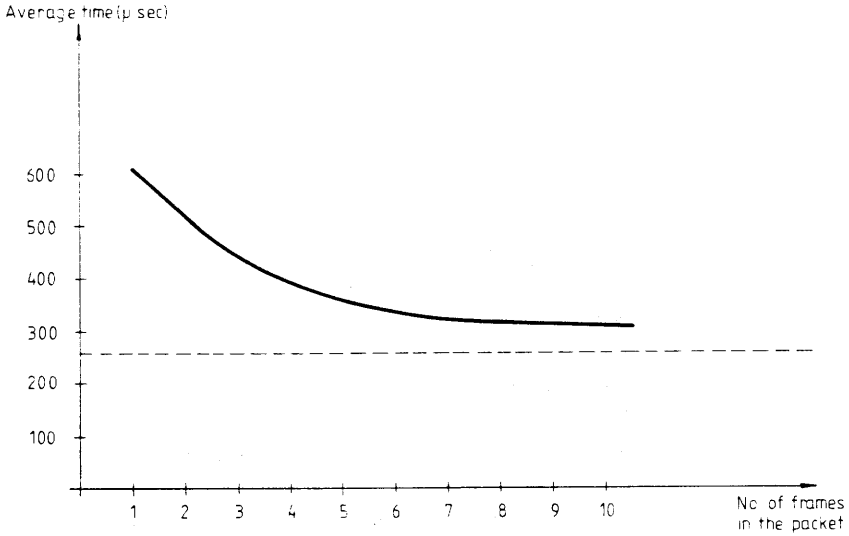


Fig. 8.2.2.2-3 TIMECONSUMPTION PER INCOMING FRAME IN STI

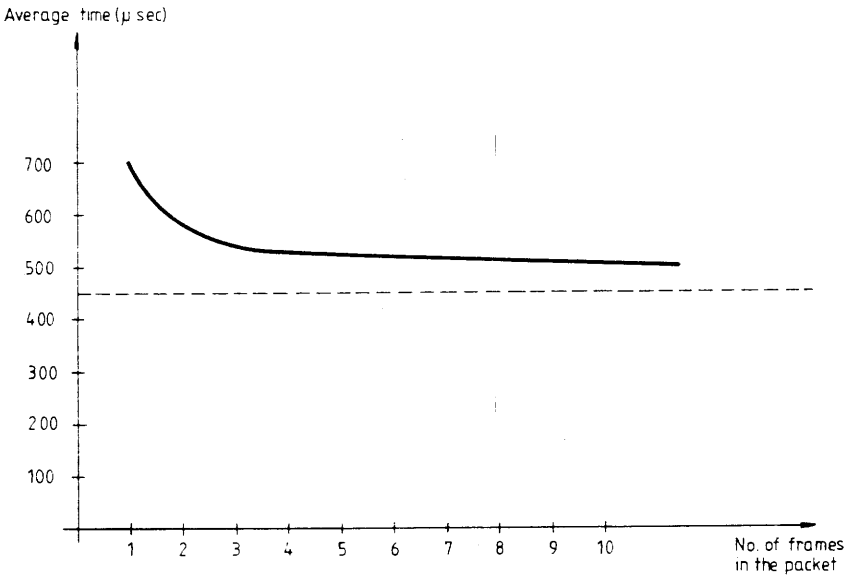


Fig. 8.2.2.2-4 TIMECONSUMPTION PER OUTGOING FRAME IN STI

afs.8

8.2.2.3 LTUX-S

Introduction

The LTUX-S device is the standard interface between the TDX Bus and four serial lines with communication lines, terminals, modems, etc. connected.

The LTUX-S is built around a standard microprocessor using a high speed Large Scale Integrated circuit for interfacing and low level protocol handling of the TDX Bus, and standard microprocessor LSI's for interfacing the user applications. The LTUX-S provides up to 8 full duplex logical data channels from the user application to the TDX Bus, each with full TDX protocol, with the microprocessor supporting both application processing of the TDX interface, TDX protocol and TDX channel set-up. The sum of bandwidth assigned to the channels through the LTUX is dynamically changeable by request to the TDX controller.

Functional Description

Overview

The interface to the TDX Bus is performed by a state controller. The Bus I/F receives frames from the TDX Bus and performs CRC-check, as well as transmits frames from the LTUX to the TDX Bus inserting frame flags and CRC bytes.

The received frame is stored in a FIFO 1 and on the fly the CR-ID is investigated for match with the LTUX device No. and the appropriate bits in the CR-ID. If a match is detected and the CRC is correct, the frame is routed to another FIFO2, interrupting the microprocessor for reading the frame. The interrupt branch (FRAME-routine) transfers the frame from FIFO 2 to a free ringbuffer, and the Ingoing Scan Process transfers

the frame to a data buffer associated to one out of 10 protocol descriptors identified by the CR-ID (data type) and simultaneously executes the TDX protocol on the frame. From the Protocol Descriptor the data buffer is fetched by a higher level (i.e., application S/W).

The transmitted frame comes from a data buffer fetched from a Protocol Descriptor by the Outgoing Scan Process and routed to a FIFO 3 directly while executing the TDX Protocol on the frame.

Overleaf (Fig. 8.2.2.3-1) is shown the data flow in the LTUX-S. Routing the frame from FIFO 3 to the upper TDX BUS is made upon a match by the send permission detector between the MUX. No. transmitted with the frames on the lower TDX BUS and the LTUX device No.

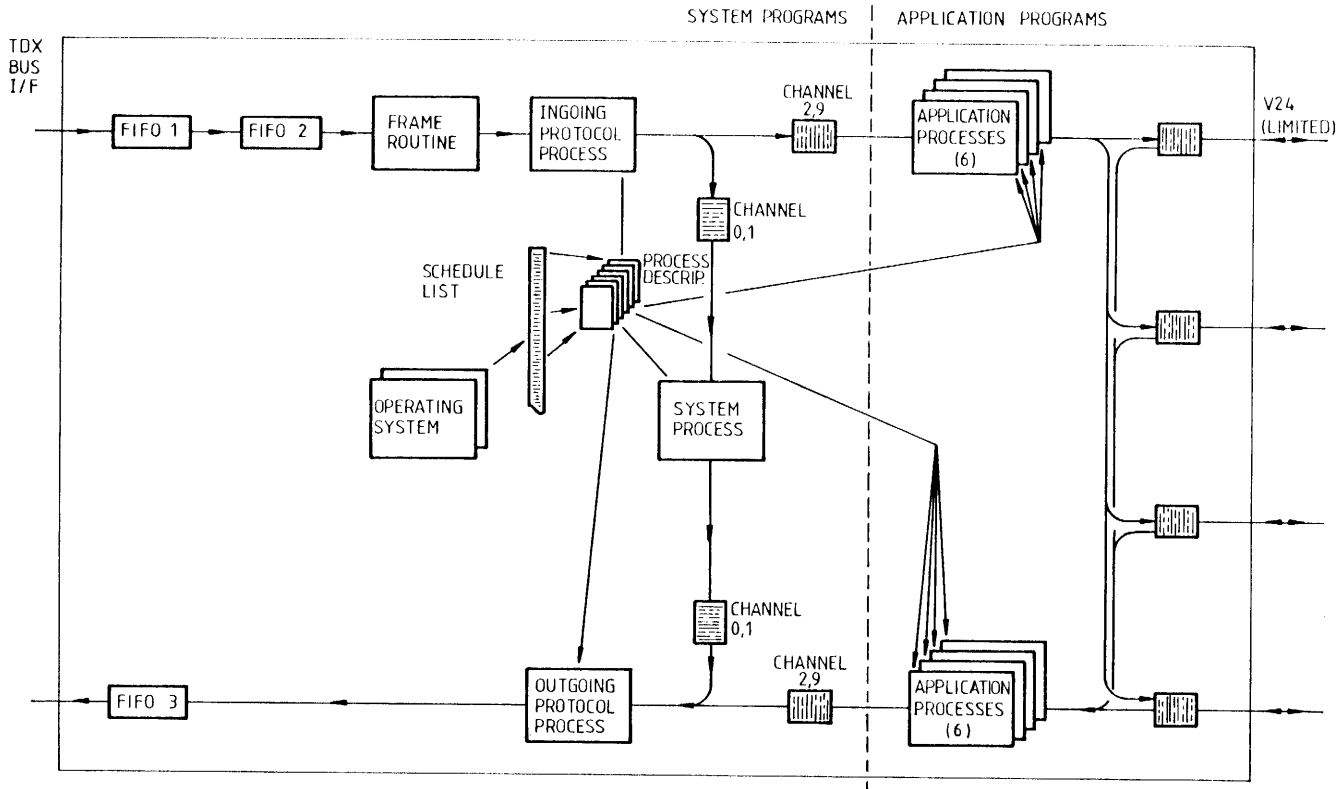


Fig. 8.2.2.3-1 LTUX-S DATAFLOW

Operating System

The operating system consists of a sequential monitor which in a predetermined order schedules 16 processes. Each process owns the CPU until a scheduling monitor-call is executed. Each process has a unique number between 0 and 15 which is used to identify the process. Processes can be activated, passivated and created via monitor calls, but all processes are initially in the schedule list.

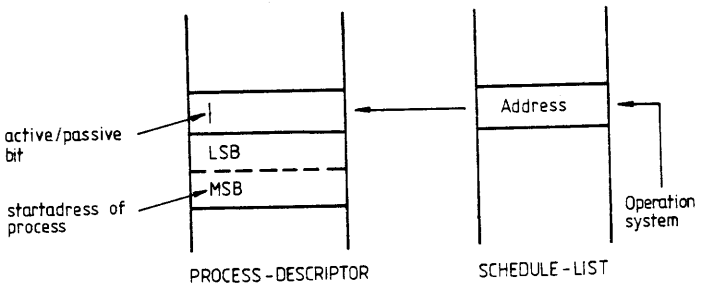


Fig. 8.2.2.3-2 LTUX OPERATING SYSTEM SCHEDULING

The creating primitive inserts the desired program counter (start address) in the process descriptor and activates the process (i.e. sets the active/-passive bit) whereby the operating system adapts the process. By passivating (i.e. resetting the active/passive bit) the process is omitted.

Notice: Each process must have a private stack which is not the system stack.

System Programs

The LTUX-S has four memory sockets of which two are used by the system programs:

Address 0-7FFH	(4k EPROM)	- used as system PROM
Address 4000H-47FFH	(4k EPROM)	- used as application PROM
Address 5400H-5700H	(1k RAM)	- used as system RAM
Address 5000H-53FFH	(1k RAM)	- used as application RAM
Address 6000H-67FFH	(2k RAM)	- used as application RAM

The system programs occupies processes no. 0,1,2,3 and performs all protocol handling on all channels. The interface to higher levels (application-levels) is based on four queues (for data channels 2-9) and a command handler (for command channels 0,1).

The four data queues are:

a. incoming full packets from the TDX Bus
b. incoming empty packets to bef illed by the TDX Bus.
c. outgoing full packets to the TDX Bus
d. outgoing empty packets from the TDX Bus.

Queues a and d have the TDX Bus as source
Queues b and c have the application-level as source.

An 8-bit cyclic timer is available (location 54A7H) with a time unit on 8 msec. This unit is used in flowcontrol-timers and receive/transmitter timers, too (see section 8.3.2).

Data Structure

The interface between the application level and the protocol level is based on queues with data buffers, where one databuffer contains one TDX-packet. The system databuffer is shown in section 8.2.2, and two queueing primitives transfers buffers between queues.
Each queue has a queue head associated with following structure

Relative address

0	Spare
1	No. of elements in queue
2	address of first element
3	
4	address of last element
5	

Application Programs - Environment

The application programs interfaces to a four serial channels which is programmable with regard to operation (sync, async, HDLC) and characteristics (5-8 bits/char, odd/even or no parity) etc.

The four serial channels can operate as either DTE or DCE with V24/V28 interface (limited) dependent on selected backpanel (see section 8.2.3.3.4).

Performance and Limitations

The required average time consumption per TDX frame is shown overleaf (Fig. 8.2.2.3-3, 8.2.2.3-4) as a function of no. of frames in the transferred TDX-packet. The access-time to the TDX-Bus is considered as negligible (refer to section 8.2.2.1 for further details).

Notice that incoming frames can be buffered in four levels but the interrupt driven frame-routine requires 182 micro sec. whereby two frames following immediately after each-other can not be received properly but causes a retransmission. This limitation can be reduced by using memory chips with access-times less than 250 nano sec. whereby an increase in processing power on 20% can be achieved.

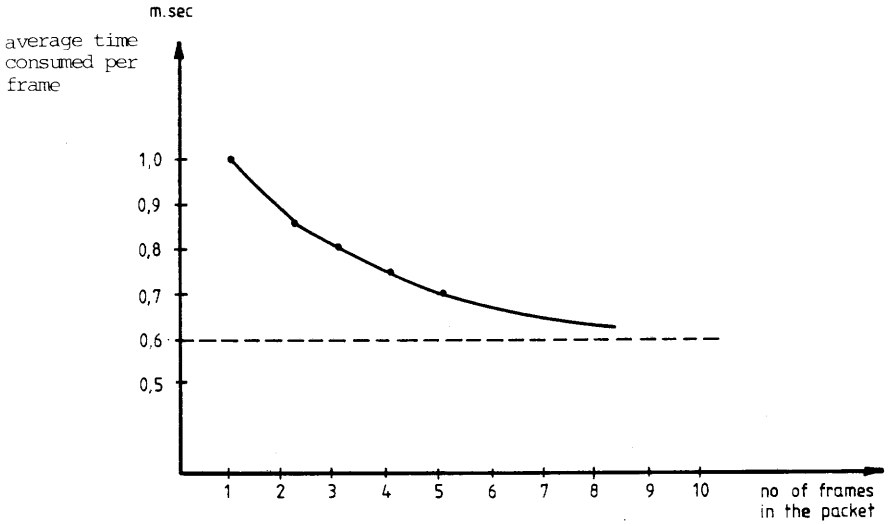


Fig. 8.2.2.3 - 3 TIMECONSUMPTION PER INCOMING FRAME

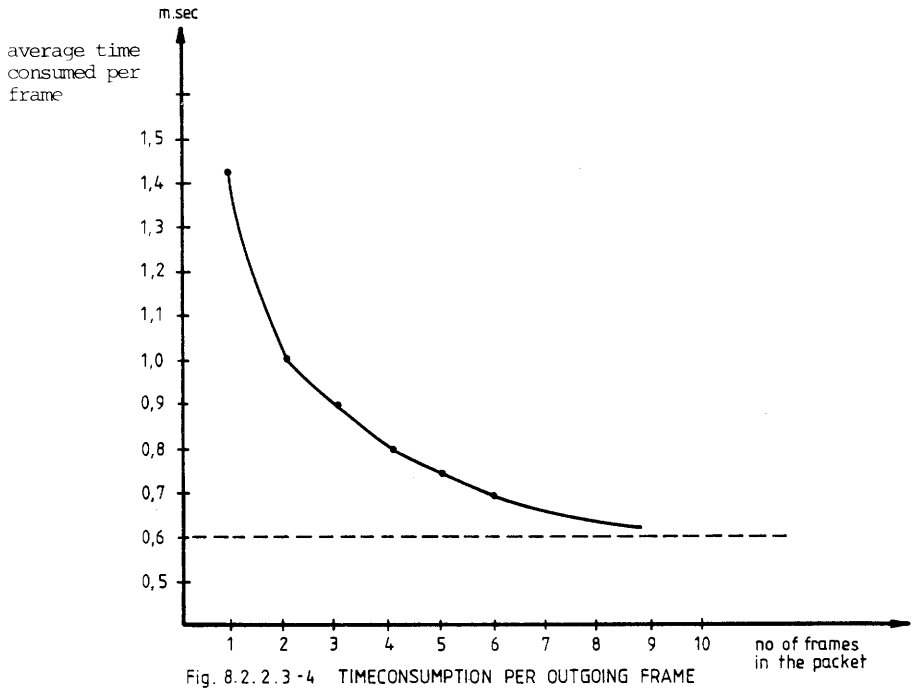


Fig. 8.2.2.3 - 4 TIMECONSUMPTION PER OUTGOING FRAME

afsnit 8

8.2.2.4 LTUX-M

Introduction

The LTUX-M device is a high performance interface between the TDX Bus and serial communication lines (terminal, modems etc) and process monitor and control equipment (production machines, monitor equipment etc.*).

The LTUX-M is based on from 2-4 microprocessorbased boards using Large Scale Integrated circuits for interfacing to the TDX Bus and peripheral equipment.

In the basic configuration the LTUX-M consists of two boards, a Front-End board (LTUX-M-FE) and a Main Processor Board (LTUX-M-CPU).

The LTUX-M-FE handles the complete TDX protocol and serves the interfacing to the TDX-Bus.

The LTUX-M-CPU, of which up to three can be connected to one LTUX-M-FE, each independently performs high level data processing as protocols (X25, BSC, 3270, TC500 etc) or process monitor and control programs.

The LTUX-M provides up to 14** full duplex logical data channels each with full TDX protocol with and accumulated data rate up to 120 kbit/sec. The bandwidth assigned to the LTUX-M device can be changed by request to the TDX controller.

* As an option X21 I/F is available.

** As an option up to 46 datachannels can be achieved by using STI/TIA addressing mode (see section 8.2.1.3).

Functional Description

Overview

The interface to the TDX Bus is performed by a state controller. The Bus I/F receives frames from the TDX Bus and performs CRC-check, as well as transmits frames from the LTUX to the TDX Bus inserting frame flags and CRC bytes.

For the following discussion on the LTUX-M dataflow, please refer to Fig. 8.2.2.4-1 overleaf. The received frame is stored in a FIFO 1 and on the fly the CR-ID is investigated for match with the LTUX device No. and the appropriate bits in the CR-ID. If a match is detected and the CRC is correct, the frame is routed to another FIFO 2, interrupting the microprocessor for reading the frame. The interrupt branch (FRAME-routine) transfers the frame from FIFO 2 to a free ringbuffer, and the Ingoing Scan Process transfers the frame to a data buffer associated to one out of 16 protocol descriptors identified by the CR-ID (Data type) and simultaneously executes the TDX protocol on the frame. From the Protocol Descriptor the data buffer is fetched by a higher level (i.e., application S/W).

The transmitted frame comes from a data buffer fetched from a Protocol Descriptor by the Outgoing Scan Process and routed to FIFO 3 directly while executing the TDX Protocol on the frame.

Transfer of data from each of the CPU-boards to the Front End board is performed via I/F-routines resident on each CPU-board. These routines are interrupt-driven as the Front-End board every millisec.* interrupts the CPU-boards to enforce the executing of the I/F-routines.

* the 1 millisec. interval implies a very small delay in the system but other timers can optionally be custom defined.

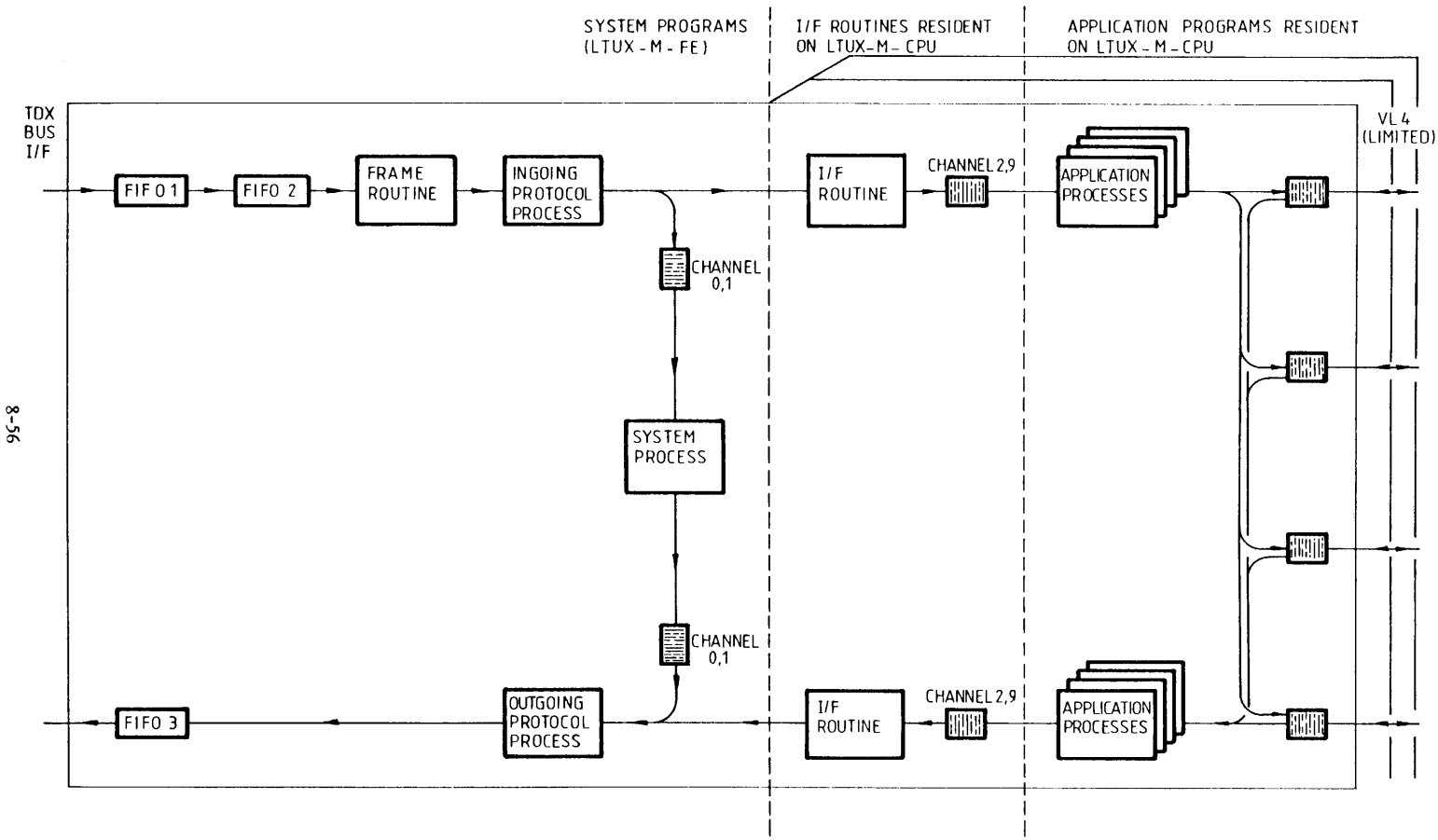


Fig. 8.2.2.4-1 LTUX-M Dataflow

System Programs

The LTUX-M has eight memory sockets of which two are used by the I/F-routines.

Four of the sockets are 28 pin and the remaining four are 24 pin whereby mix of 8, 4, 2 and 1 kbyte memory-chips is achieved (custom defined).

The interface to higher levels (application-levels) is based on four queues (for data channels 2-16) and a command handler (for command channels 0,1).

The four data queues are:

a. incoming full packets from the TDX Bus
b. incoming empty packets to bef illed by the TDX Bus.
c. outgoing full packets to the TDX Bus
d. outgoing empty packets from the TDX Bus.

Queues a and d have the TDX Bus as source
Queues b and c have the application-level as source.

Data Structure

The interface between the application level and the protocol level is based on queues with data buffers, where one databuffer contians one TDX-packet. The system databuffer is shown below, and two queueing primitives transfers buffers between queues.
Each queue has a queue head associated with following structure:

Relative address

0	Spare
1	No. of elements in queue
2	address of first element
3	
4	address of last element
5	

Application Programs - Environment

The application programs interfaces to four serial channels on each CPU board which is programmable with regard to operation (sync, async, HDLC) and characteristics (5-8 bits/char, odd/even or no parity) etc.

The four serial channels can operate as either DTE or DCE with V24/V28 interface (limited) dependent on selected backpanel (see section 8.2.3.3.4).

Furthermore slave modules are available via the microbus (see sections 8.2.3.3.7.4-6) whereby connectivity to a wide range of equipment is achieved.

Performance and Limitations

The required average time consumption per TDX frame is shown overleaf (Fig. 8.2.2.4-2 and 8.2.2.4-3) as a function of no. of frames in the transferred TDX-packet. The access-time to the TDX-Bus is considered as negligible (refer to section 8.2.2.1 for further details).

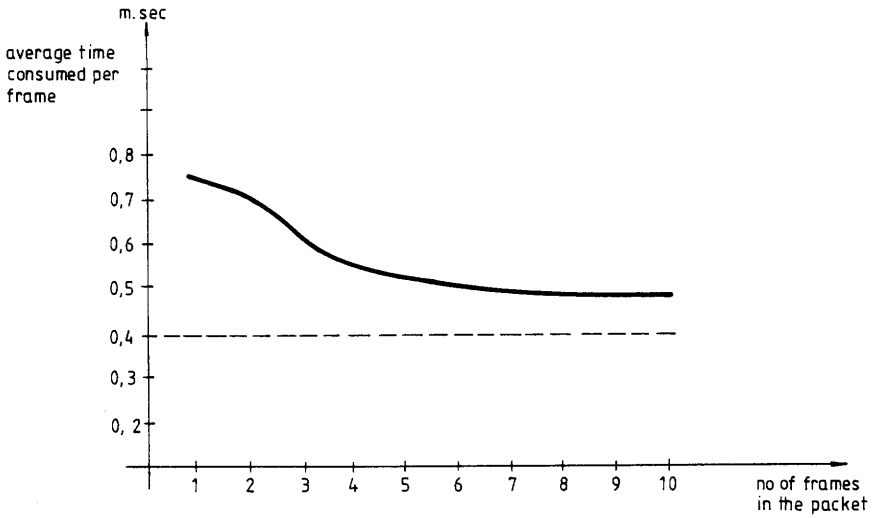


Fig. 8.2.2.4-2 TIMECONSUMPTION PER INCOMING FRAME

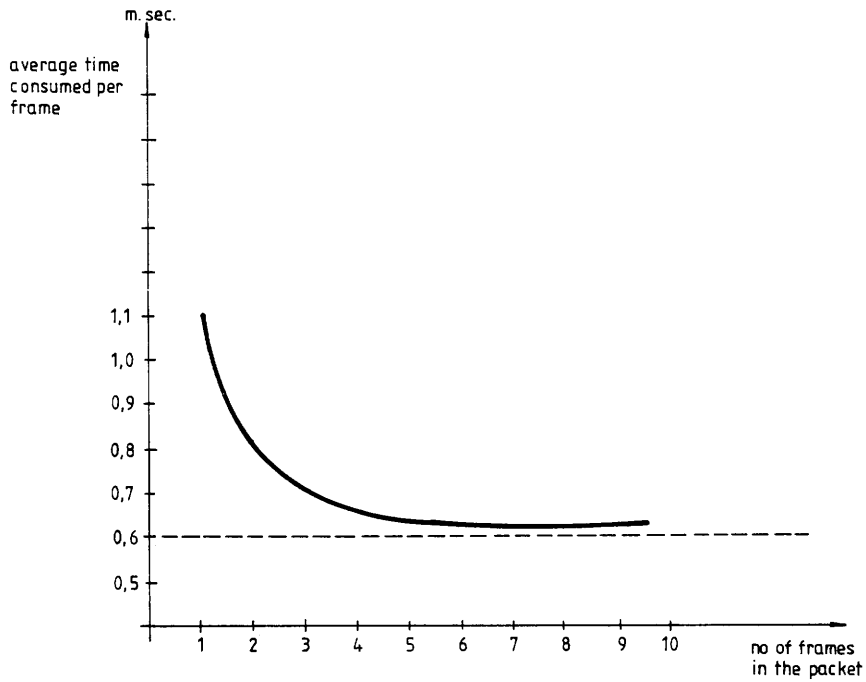


Fig. 8.2.2.4-3 TIMECONSUMPTION PER OUTGOING FRAME

afs. 8

8.2.3 System - Component Specification

8.2.3.1 TDX Bus

The TDX Bus configuration consists of one or more Sections of terminated coaxial cable pairs which can be interconnected. A local TDX system can be achieved by interconnecting LTUX-crates via flat cables (BTMX Bus).

The physical limits that restrict the system designer in configuring TDX Networks are given below. The drawing overleaf shows typical large configuration.
Explanation and calculation of the below fundamental constraints are found later in this chapter.

1. TDX Bus cable section is constituted of a coaxial cable pair terminated at each end, maximum coaxial cable length (Lsect.) of a section, depends on selected cable type:

TM3078	:	Lsect.= 800 meter	recommended types
RG111A/U or RG22 B/U	:	Lsect.= 1300 meter	

A maximum of 64 Wall Outlets must be attached to any section. There are no restrictions to distribution of Wall Outlets along the cable and different sections may use different cable types.

2. Maximum of 2 Amplifier and Branching Units (XAB) must be attached between the Controller and any TDX device, this allows for a maximum of 4 XAB's between any two devices.

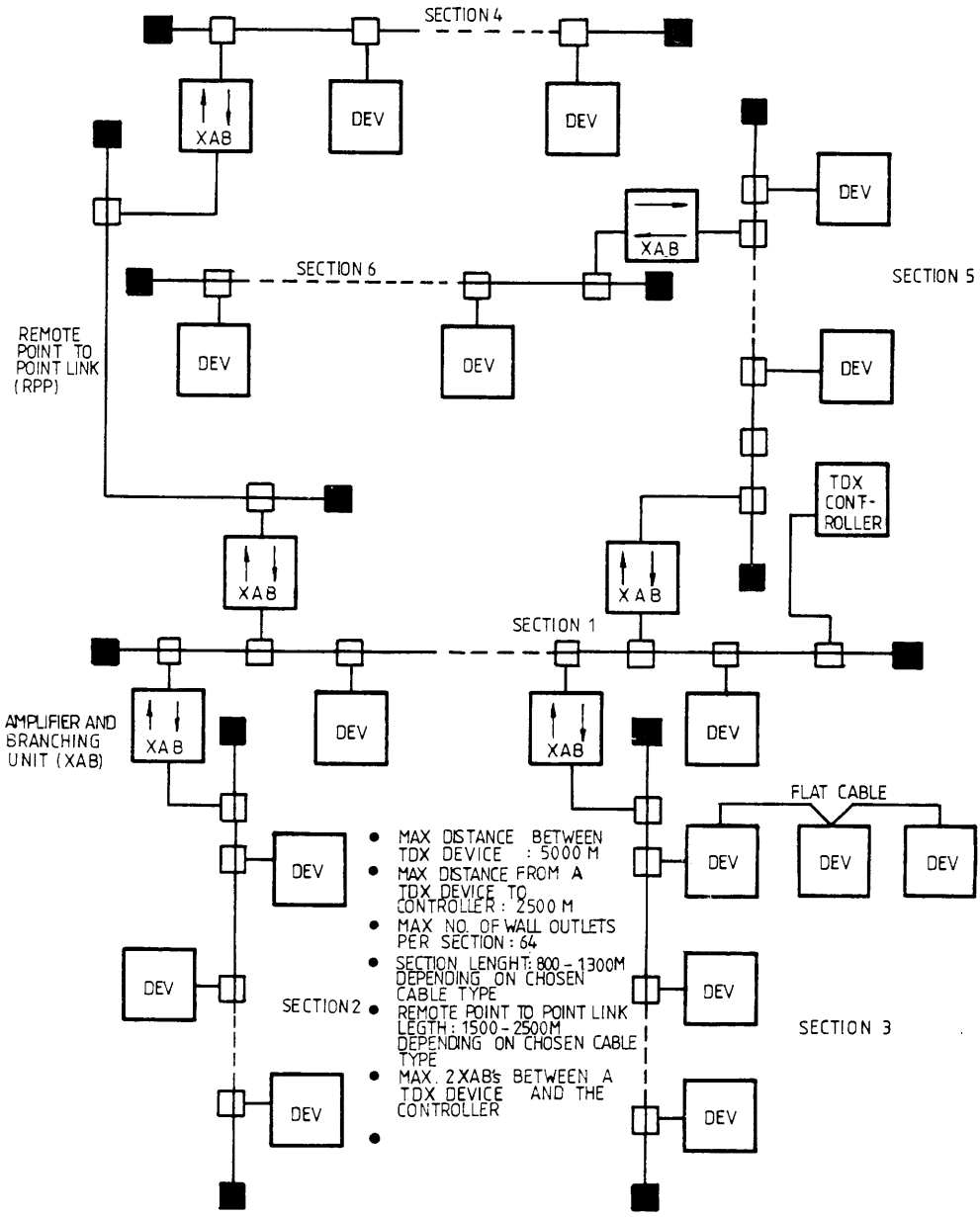
3. Maximum 2500 meters of coaxial cable length (including Point to Point links) between the controller and any TDX device, this allows for up to 5000 meters between devices.

4. Point to Point links, used for connecting remote sections in other buildings or floors within a large building, shall be within the following maximum length:

TM3078	:	1500 meter
RG111 A/U or GR22 B/U	:	2500 meter

A Point to Point link may use different cable type from the cable types used for the TDX Bus sections it connects to at each end.

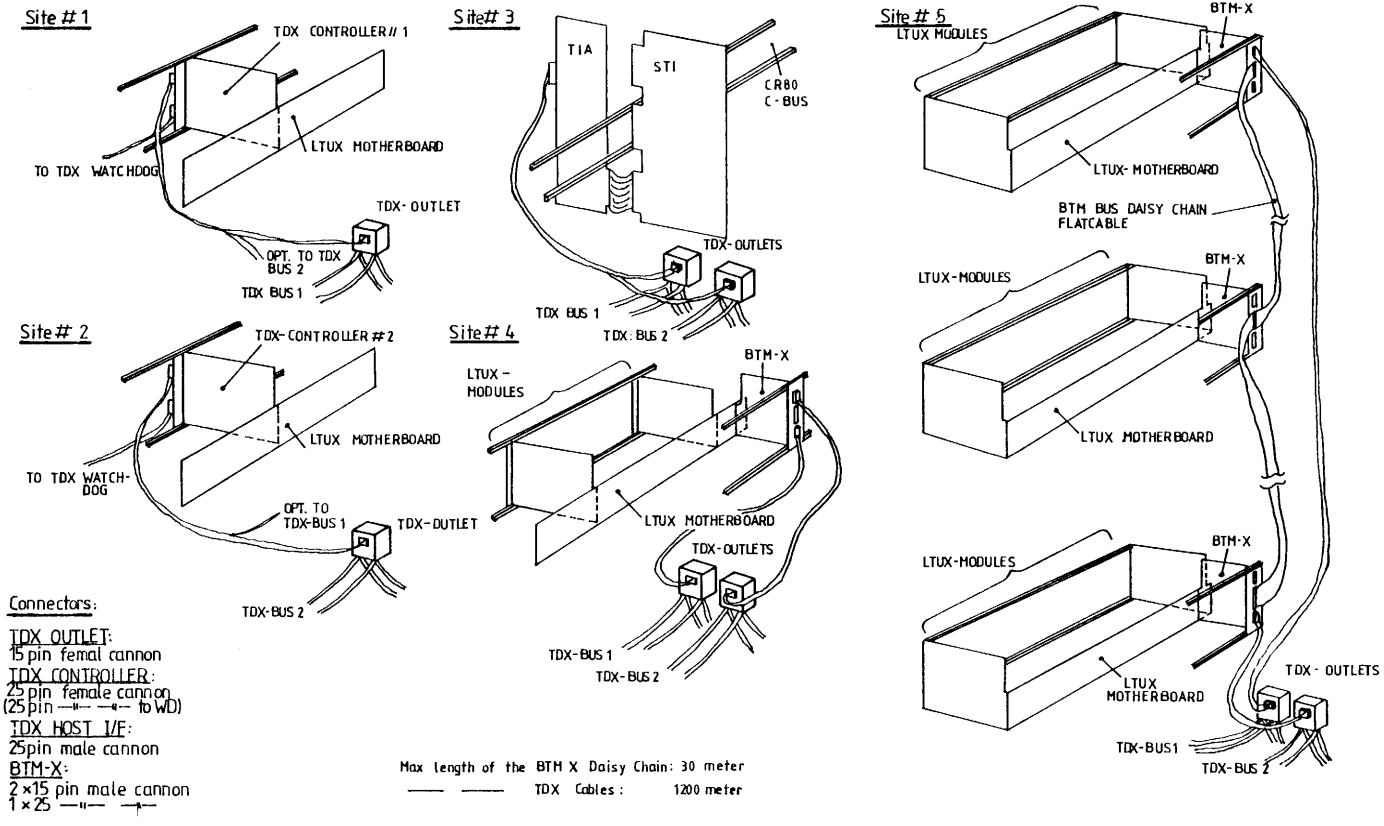
5. Maximum length of flat cable between TDX-Devices is 30 m.



TDX BUS CONFIGURATION OVERVIEW

afs.8

Example of TDX-system configuration with dualized TDX-buses and TDX controllers



8.2.3.1.1 Cables

Coaxial Cables

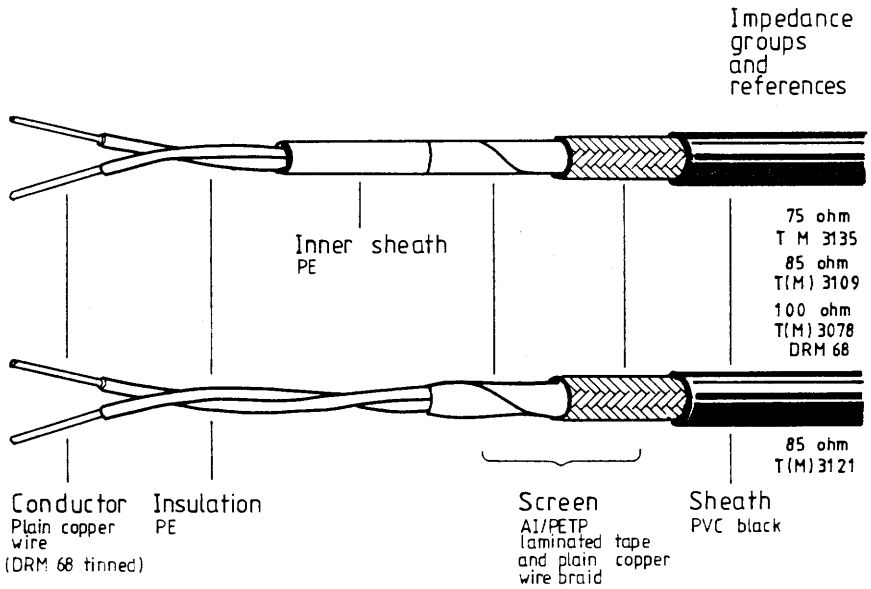
For distances up to 800 m the two core screened RF cable BICC T(M)3078 are used.
The characteristics of this cable are:

Char. imp.:	100
Nom. capacitance:	52 pf/m
Att. at 2 mhz:	2.18 db/m
diameter:	6.75 mm
Min. bending radius:	35 mm
Approx. weight:	69 kg/km
Max. operating temp.:	70°C
Insulation:	polyethylene

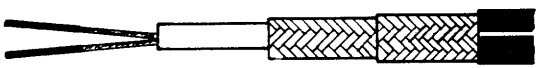
For distances up to 1300 m, the RG 22 B/U or the armed version RG 111 A/U are used. These cables comply with mil. spec. C-17D

Char. imp.:	95
Nom. capacitance:	53 pf/m
Att. at 2 mhz:	1.3 db/m
diameter:	10.8 mm RG 22 B/U 12.4 mm RG 111 A/U
Approx. weight:	180 kg/km RG 22 B/U 300 kg/km RG 111 A/U
Max. operating temp.:	70°C
Insulation:	polyethylene

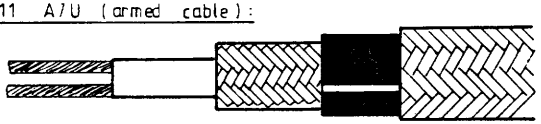
T (M) 3078:



RG 22 B/U (suitable for direct burial):



RG 111 A/U (armed cable):



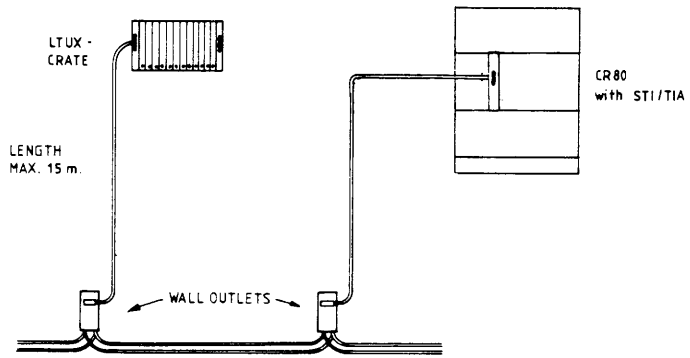
TDX Balanced, shielded twinlead cable.

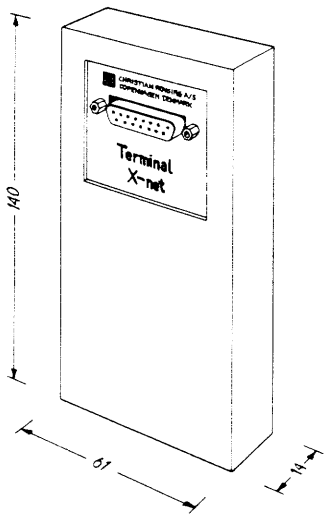
afs. 8

8.2.3.1.2 Wall Outlets

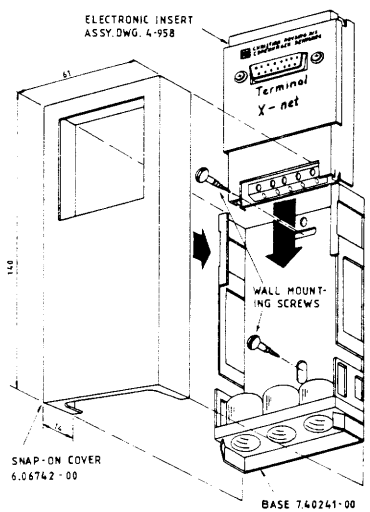
Connections to the TDX Bus coaxial cable pin are done via Wall Outlets (XWO-CR2510), as shown below (Fig. 8.2.3.1.2-1) and overleaf (Fig. 8.2.3.1.2-2).

Fig. 8.2.3.1.2-1 TDX BUS INSTALLATION





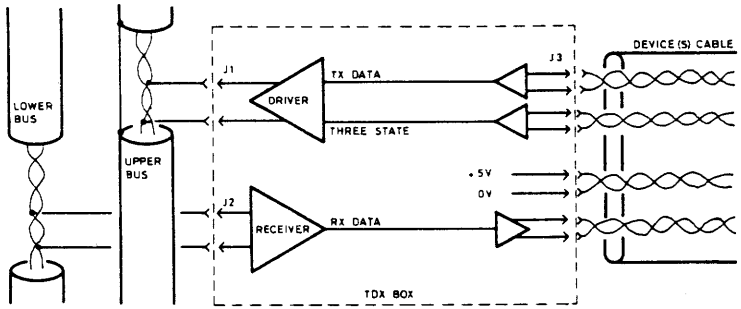
Wall Outlet



Exploded View

Wall Outlet

BLOCK SCHEMATIC



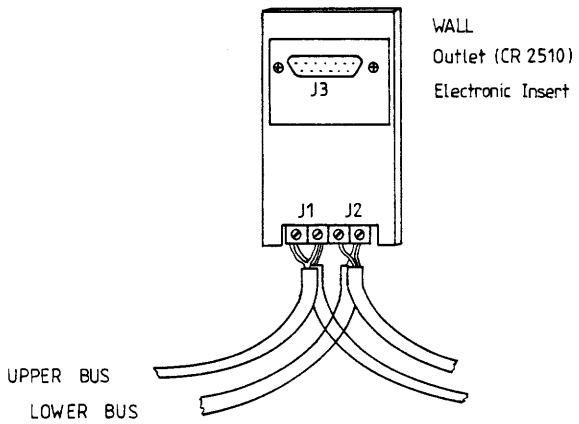


Fig. 8.2.3.1.2-2 WALL OUTLET, ELECTRONIC INSERT

Electrical Specifications

The signals have an electrical characteristics in accordance with RS 422, meaning that following level are applicable:

Logical 1: Signal+ $\geq$ 2.5V; Signal- $\leq$ 0.5V
Logical 0: Signal+ $\leq$ 0.5V; Signal- $\geq$ 2.5V

J1:(TDX Bus Driver)

These two screw terminals are used for termination of lower bus if TDX Controller or upper bus if other TDX modules are connected to the outlet (J3). In the end of the bus termination resistance (95 ohm) is mounted instead of one of the cables.

Driver amplitude when enabled (two cable basis):	>13Vpp
Driver (off-state resistance):	>24k ohm 10MHz

J2: (TDX Bus receiver)

These two screw terminals are used for termination of upper bus if TDX Controller or upper bus if other TDX modules are connected to the outlet (J3). In the end of the bus a termination resistor (95 ohm) is mounted instead of one of the cables.

Receiver sensitivity:	<130 mVpp
Receiver input resistance:	24k ohm

J3

This connector is used for interfacing the TDX modules to the bus by means of flat cable or multiwire cable.

Connector types: 15 pins.

8.2.3.1.3 Amplifier and Branching Unit (XAB)

The TDX Amplifier and Branching Unit (XAB CR2520-/0XX--/00) shown below, is used to interconnect TDX Bus Sections. The units are to be mounted on walls, in racks, etc., and the power supply unit is directly plugged into a Mains Power socket or alternatively the XAB can be supplied by +5V DC.

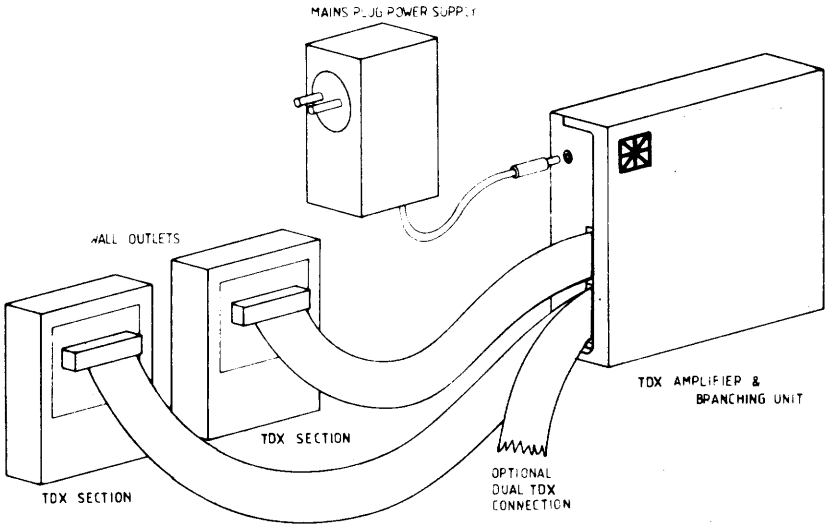


Fig. 8.2.3.1.3-1
TDX AMPLIFIER AND BRANCHING UNIT

8.2.3.2 CR80 Interface Unit

The CR80 Interface Unit consists of STI/TIA Devices and a cable between the Interface Unit and the TDX-Bus, as illustrated below in Fig. 8.2.3.2-1. The STI-TIA is described in details in section 8.2.2.2 and in the datasheets.

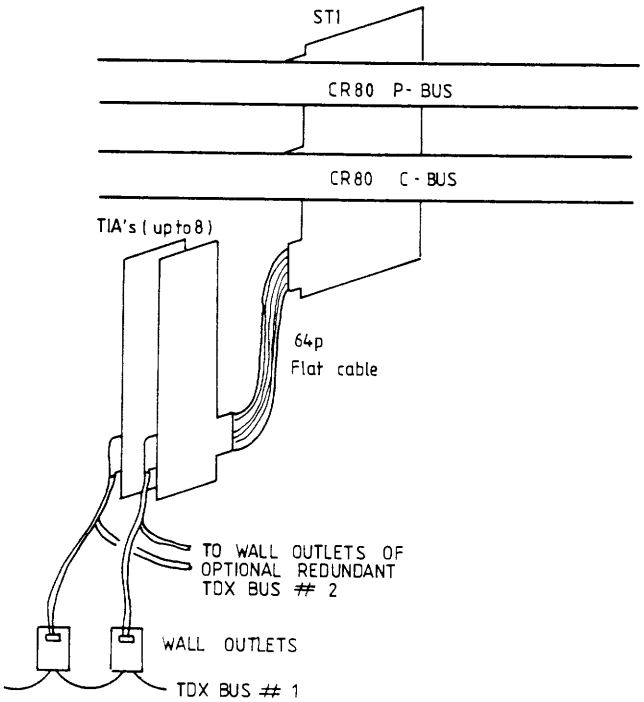


Fig. 8.2.3.2-1 CR80 STI/TIA INTERFACE TO THE TDX BUS

8.2.3.3 Communication Line Interface Unit

The Communication Line Interface Unit consists basically of a crate housing LTUX-devices/subdevices.

Following modules are used in the basic crate:

- power supply
- power panel
- bus termination module
- back-panels
- motherboard
- LTUX-devices

When the crate is equipped with subdevices, a micro-bus is required. This can be extended to other crates via an Active Micro Bus Extended (AMBE).

Overleaf is shown the mechanical configuration of the 19" crate used for rack-mounting.

8.2.3.3.1 Power Supply Module

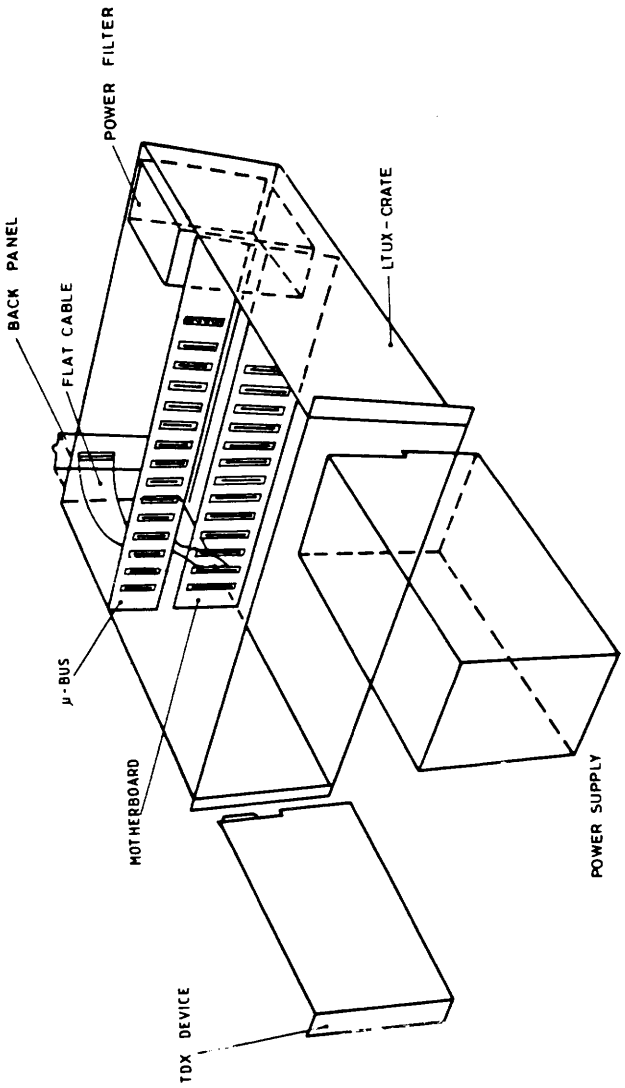
The standard Power Supply is a plug-in module allowing replacement within a few secondes. The unit which supplies the crate with DC voltages has the following characteristics:

Output: 5VDC : 32A
-12VDC : 1.2A
+12VDC : 2.8A

The mechanical specifications of the Power Supply are as shown in Fig. 8.2.3.3.1-1. For further details, please refer to the datasheets.

The module is inserted from the front of the crate and is fixed by means of one finger screw.

Figure 8.2.3.3-1
LTUX CRATE



8.2.3.3.2 Power Panel

The LTUX-crate is equipped with a Power Panel to which the mains power is interfaced.

The Power Panel contains a main power switch, fuse, filter and cabling for power distribution to the power supply connector.

The mechanical specifications are as shown in Fig. 8.2.3.3.2-1, and the interface specifications are as follows:

Voltage/freq.:	220V/50Hz (optional 110V/60Hz)
Input Power:	max. 250 VA
Power Plug type:	Tüchel T2260-00 (P/N 6.06709-01)

8.2.3.3.3 Bus Termination Module

Two types of Bus Termination Modules BTM-X or BSM-X can be used for LTUX-Crate micro-bus termination and TDX bus interface.
The Bus Termination Module, BTM-X, is a multi-function unit terminating the LTUX-Bus of the LTUX-Motherboard.

The BTM-X supports interfaces to a single TDX-Bus system as well as a dualized TDX-Bus Configuration and as an optional feature the BTM-X has a connector for the BTM-Bus interconnection of more crates in a single rack which means that one rack equipped with more crates only needs to contain one TDX Wall Outlet (- or two in systems with dualized TDX-Bus).

The mechanical specifications for the BTM-X are shown in Fig. 8.2.3.3.3-1.

The module is installed in the crate from the rear (Fig. 8.2.3.3.3-2) and is fixed by means of two finger screws.

The Front-Panel of the BTM-X contains two 15 pin connectors for TDX-Wall Outlet connections and one 25 pin connector for the BTM-X bus The Block diagram of the BTM-X is shown in Fig. 8.2.3.3.3-3.

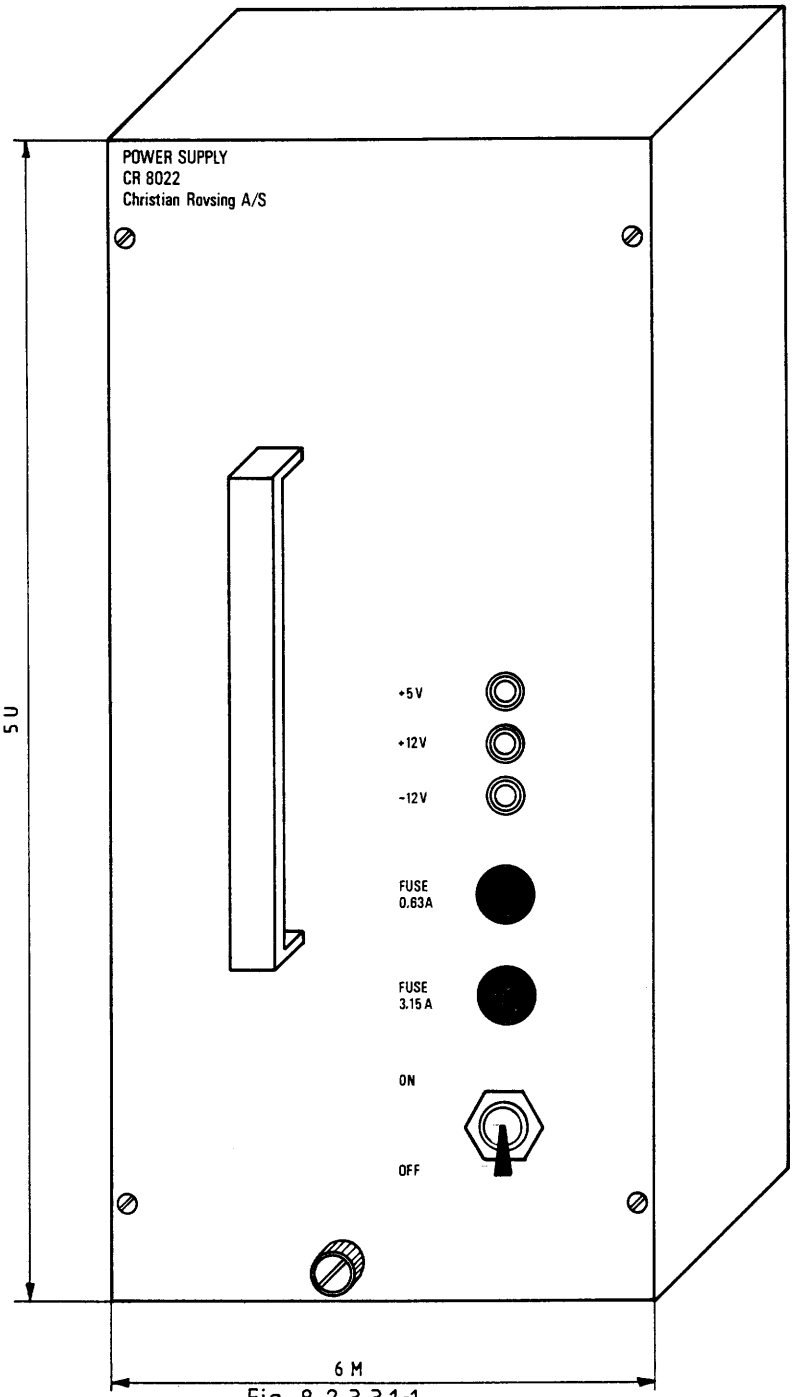


Fig 8.2.3.3.1-1
POWER SUPPLY
Front View

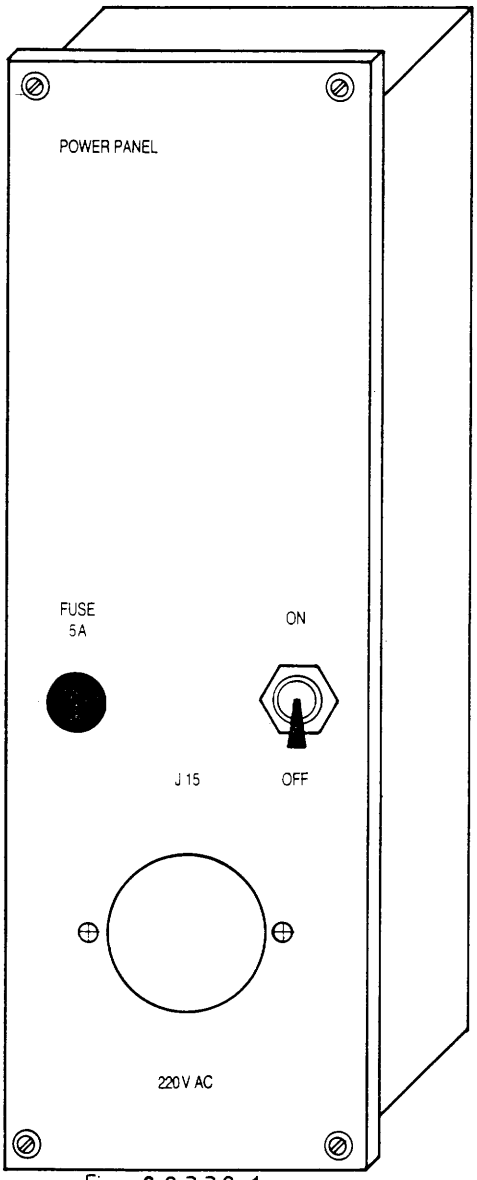


Fig 8.2.3.3.2-1
POWER PANEL
Front View

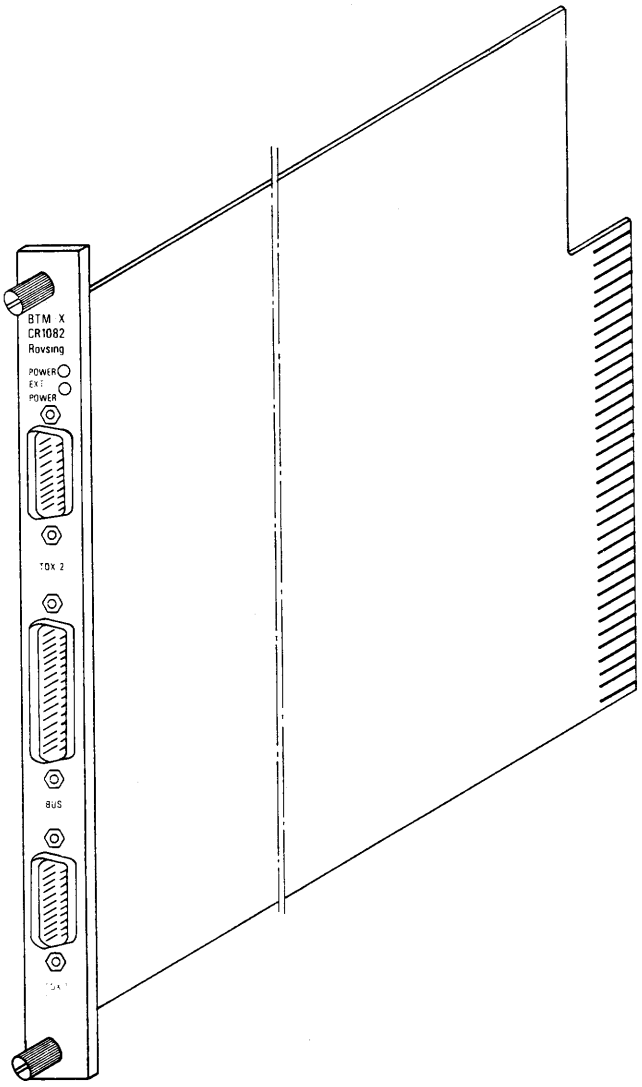


Figure 8.2.3.3.3-1
BTM-X FRONT VIEW

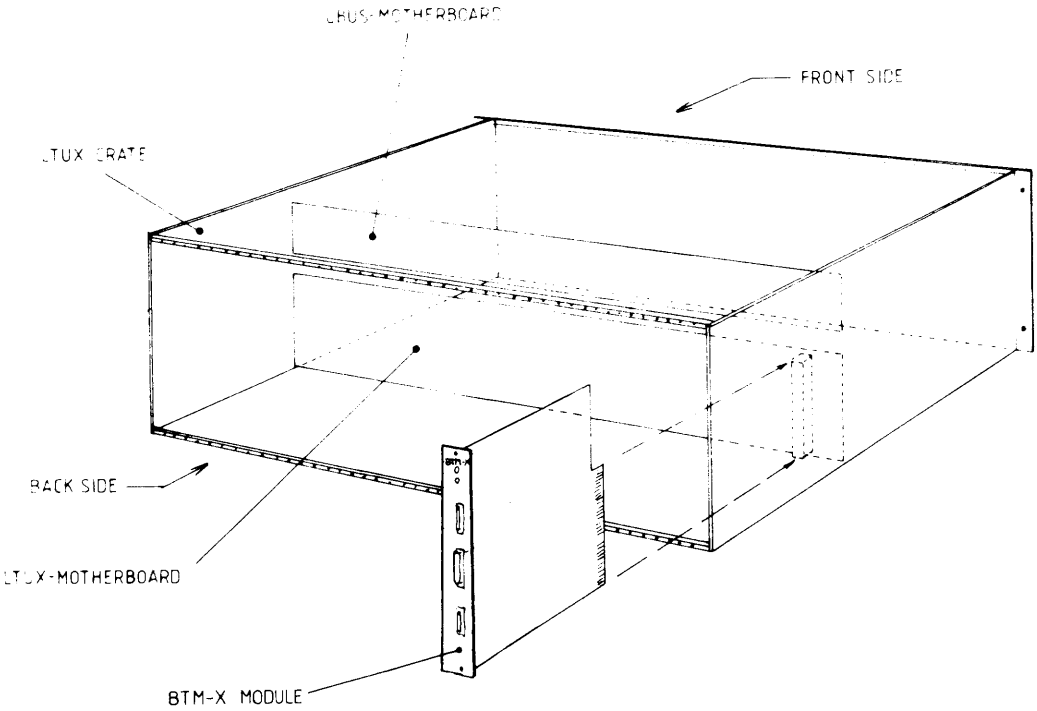
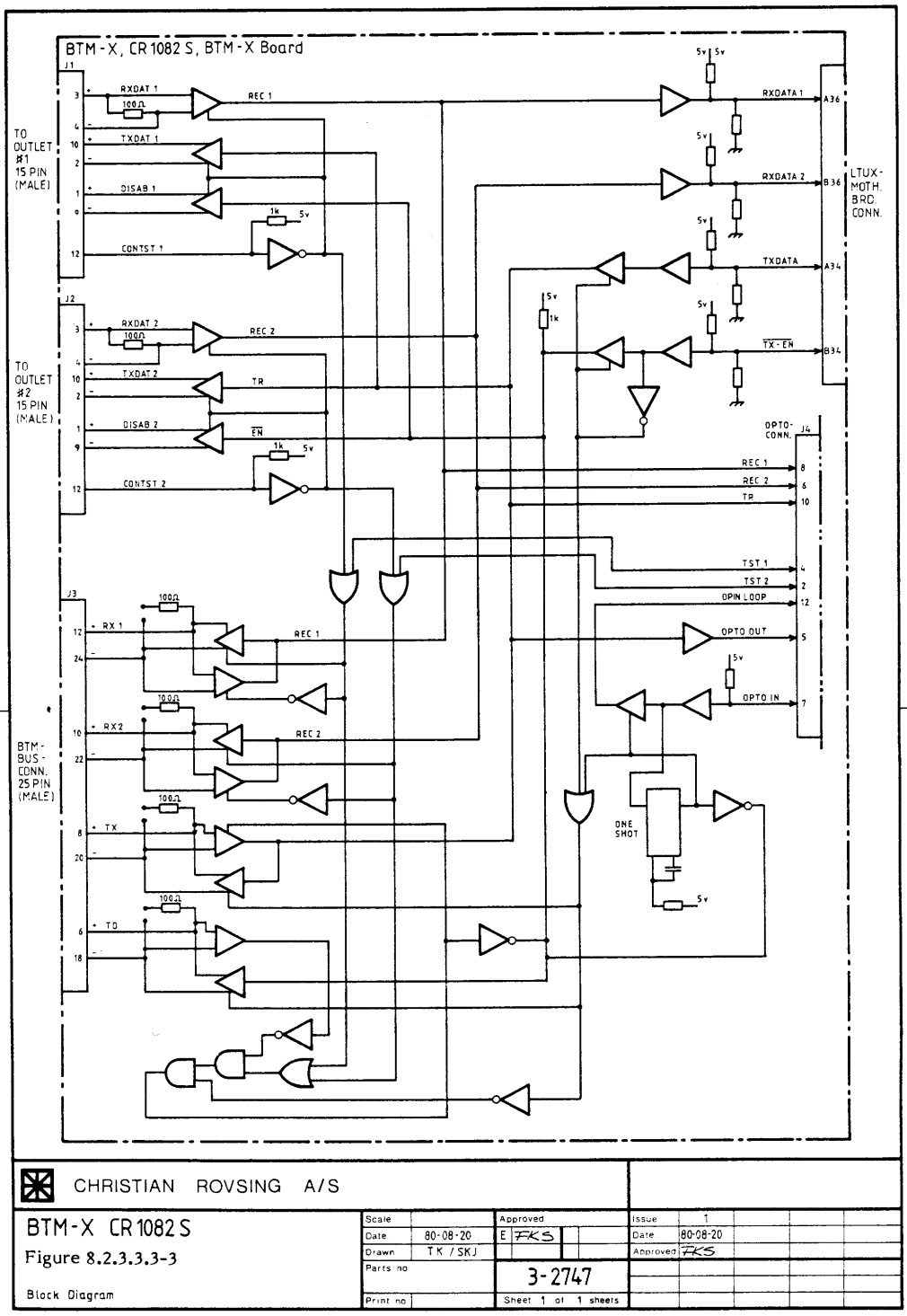


Figure 8.2.3.3.3-2
BTM-X INSTALLATION



CHRISTIAN ROVSING A/S

BTM-X CR1082S

Figure 8.2.3.3.3-3

Block Diagram

Scale		Approved		Issue	1
Date	80-08-20	E FKS		Date	80-08-20
Drawn	TK / SKJ			Approved	FKS
Parts no		3-2747			
Print no		Sheet 1 of 1 sheets			

The BSM-X, bus Termination Unit interfaces to the Configuration Bus (see chapter 7) enables monitoring and control of switch logic in dual TDX Bus Systems, by Maintenance and Configuration Processor (MCP) System. The BSM-X which can replace the BTM-X, is installed from the front of the LTUX Crate and connects to the BSM-X backpanel as shown below (Fig. 8.2.3.3.3-4).

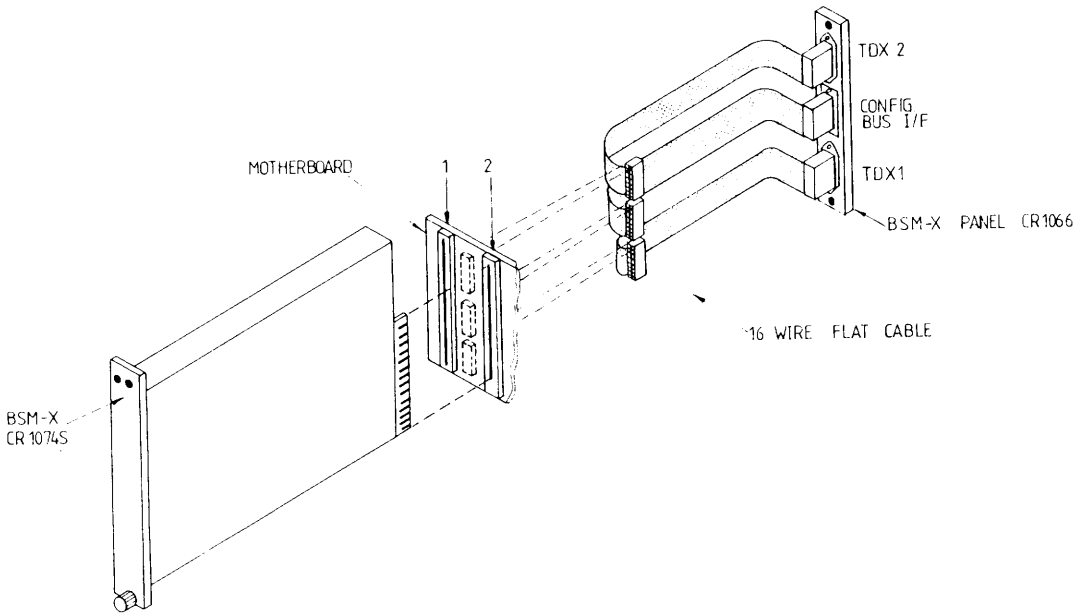


Fig. 8.2.3.3.3-4
CONNECTION OF BSM-X TO BSM-X PANEL

8.2.3.3.4 Back Panels for communication line termination

The Back Panels described in this section performs the physical interface for the line communication interfaces. They are internal in the crate connected to the individual LTUX-motherboard connector of their corresponding LTUX modules by means of 64p flatcables.

Two types of back panels (see fig. 8.2.3.3.4-1) are supported. One (type 1) performs a CCITT V24/DTE interface (see Table 8.2.3.3.4a and 8.2.3.3.4b) and the other one (type 2) performs a CCITT V24/DCE interface (see Table 8.2.3.3.4c and 8.2.3.3.4d).

The interface connectors of both types are 25 pin female Cannon connectors.

For further detailed information on these or other types of Back Panels please refer to Datasheets, Chapter 10.

The back panels are mounted on the rear of the crate by means of four 3 mm screws and connected to LTUX module as shown in Fig. 8.2.3.3.4-2.

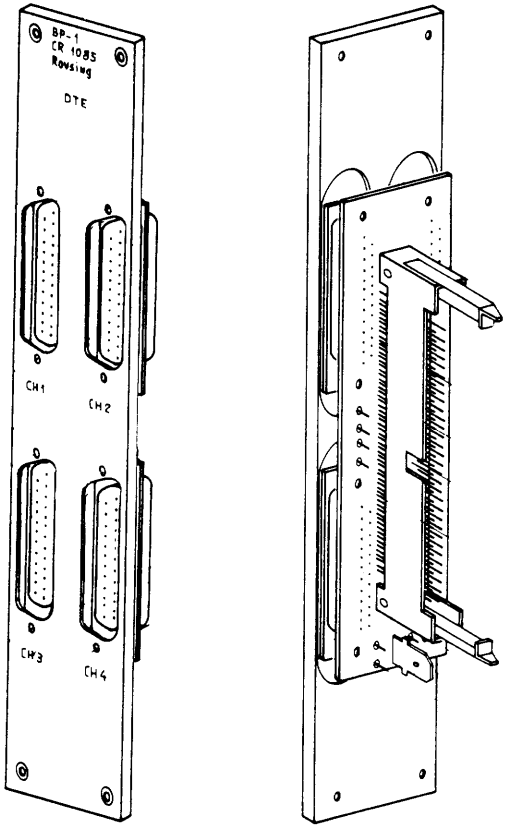


Figure 8.2.3.3.4-1
BACK PANEL
FRONT VIEW & REAR VIEW

LTUX-S:

Back Panel Type 1 (DTE)
25 pin Cannon Female Connector

Name	Circuit	LTUX-S DR./REC	Cannon Pin no.	Remarks
Pr. Gnd	101	Gnd	1	Protect Ground. Strapable.
Sign. Gnd	102		7	
TxD	103	Dr.	2	
RxD	104	Rec.	3	
RTS	105	Dr.	4	
CTS	106	Rec.	5	
DTR	108	Dr.	20	
DCD	109	Rec.	8	
TxC	114/113	Rec./Dr.	15	Selectable by straps
RxC	115	Rec.	17	

Table 8.2.3.3.4a

LTUX-M-CPU
Back Panel Type 1 (DTE)
25 pin Cannon Femal Connector.

Name	Circuit	LTUX-M-CPU DR./REC	Cannon Pin no.	Remarks
Pr. Gnd	101	Gnd	1	Protect Ground. Strapable.
Sign. Gnd	102		7	
DSRS	111	Dr.	23/11	
TxC	113/114	Dr./Rec.	15	Selectable by straps
DSR	107	Rec.	6	
S-S BY	116	Dr.	24	
RxC	115	Rec.	17	
SBY-I	117	Rec.	25/22	
DCD	109	Rec.	8	
CTS	106	Rec.	5	
RTS	105	Dr.	4	
DTR	108	Dr.	20	
TxD	103	Dr.	2	
RxD	104	Rec.	3	

Table 8.2.3.3.4b

LTUX-S
Back Panel type 2 (DCE)
25 pin Cannon Female Connector.

Name	Circuit	LTUX-S DR./REC	Cannon Pin no.	Remarks
Pr. Gnd	101	Gnd	1	Protect Ground. Strapable.
Sign. Gnd	102		7	
RxD	104	Rec.	3	
TxD	103	Dr.	2	
CTS	106	Rec.	5	
RTS	105	Dr.	4	
DSR	107	Dr.	6	
DSRS	111	Rec.	11/23	
TxC RxC-TxC	113/114-115	Rec./Dr.	15/17	Dr./Rec. Selectable by straps.

Table 8.2.3.3.4c

LTUX-M-CPU
Back Panel Type 2 (DCE)
25 pin Cannon Female Connector.

Name	Circuit	LTUX-M-CPU DR./REC	Cannon Pin no.	Remarks
Pr. Gnd.	101	Gnd	1	Protect ground. Strapable
Sign.Gnd	102		7	
DCD	109	Dr.	8	
TxC-RxC/ TxC	114-115/ 113	Dr./Rec.	15/17	Selectable by Straps.
RTS	108	Rec.	20	
SBY-I	117/125	Dr.	22/25	
S-SBY	116	Rec.	24	
DSRS	111	Rec.	11/23	
RTS	105	Rec.	4	
CTS	106	Dr.	5	
DSR	107	Dr.	6	
RxD	104	Dr.	3	
TxD	103	Rec.	2	

Table 8.2.3.3.4d

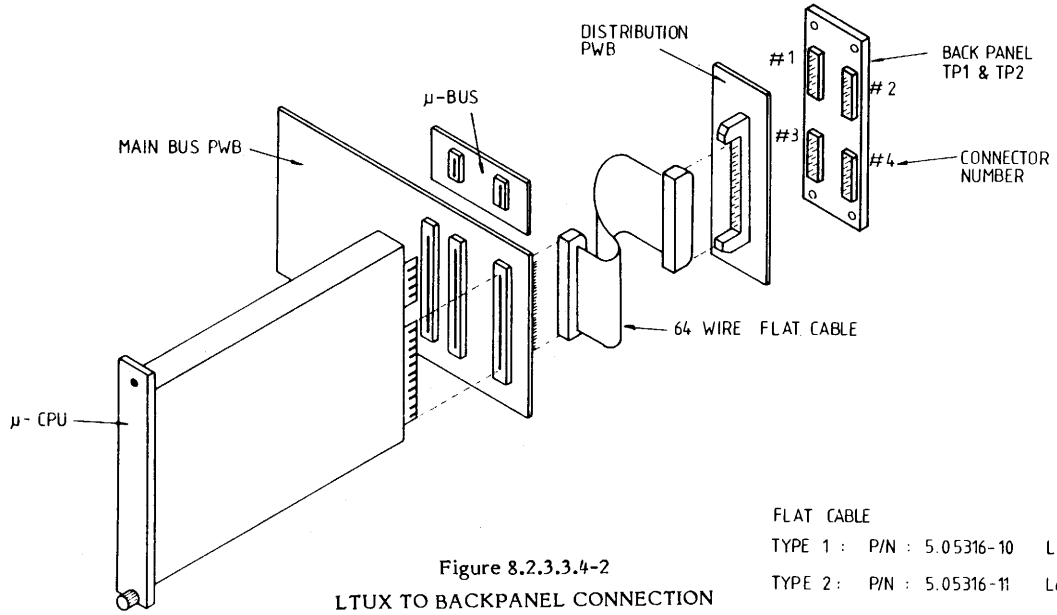


Figure 8.2.3.3.4-2
LTUX TO BACKPANEL CONNECTION

FLAT CABLE
TYPE 1 : P/N : 5.05316-10 Lenght : 280 mm
TYPE 2 : P/N : 5.05316-11 Lenght : 350 mm

8.2.3.3.5 Motherboard

The LTUX-Bus TDX signals and power distribution to the modules positioned in the crate are performed by the LTUX-Motherboard which is equipped with eighteen front-accessable 86 pin connectors and one rear accessable 86 pin connector as shown below (Fig. 8.2.3.3.5-1).

The motherboard provides a parallel bus structure meaning that the bus does not put any restrictions on the location of the LTUX modules.

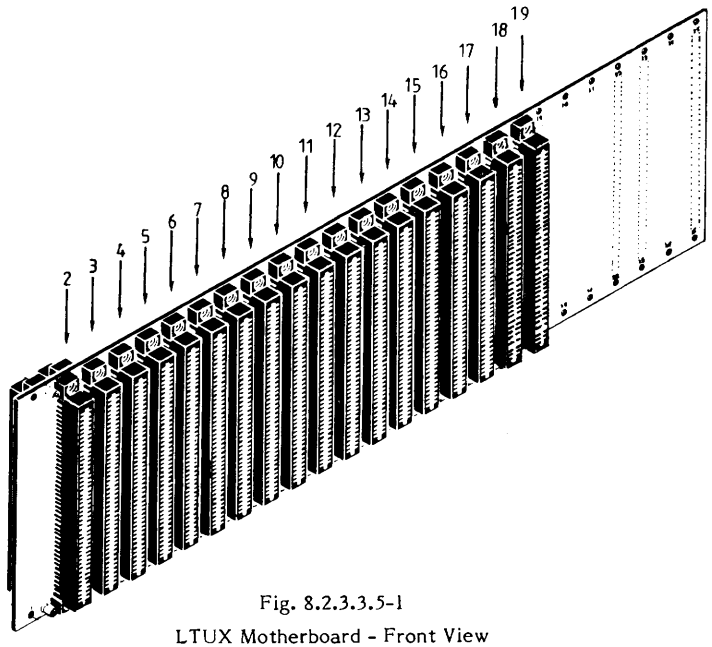


Fig. 8.2.3.3.5-1
LTUX Motherboard - Front View

8.2.3.3.6 Micro Bus

The micro bus provides a bus structure used when LTUX-M-CPU's communicate with a LTUX-M-FE or I/O modules.

The pin-assignment of the micro-bus is shown overleaf (Table 8.2.3.3.6a).

Table 8.2.3.3.6a
MICRO BUS
48 Pin Connector Pin Assignment

Pin no.	Signal Description	Pin no.	Signal Description	Pin no.	Signal Description
1a	ϕ	1b	GND (ϕ)	1c	AB0
2a	AB 1	2b	AB 2	2c	AB 3
3a	AB 4	3b	AB 5	3c	AB 6
4a	AB 7	4b	AB 8	4c	AB 9
5a	AB 10	5b	AB 11	5c	AB 12
6a	AB 13	6b	AB 14	6c	AB 15
7a	GND	7b	DB 0	7c	DB 1
8a	DB 2	8b	DB 3	8c	DB 4
9a	DB 5	9b	DB 6	9c	DB 7
10a	IEO [PROUT]	10b	GND	10c	IEI [PRIN]
11a	$\overline{\text{BUSRQ}}$	11b	$\overline{\text{WAIT}}$	11c	$\overline{\text{NMI}}$
12a	$\overline{\text{INT}}$	12b	$\overline{\text{RESET}}$	12c	EXT. CLK
13a	GND	13b	$\overline{\text{BAK}}$	13c	$\overline{\text{EXT. INT}}$
14a	RF	14b	$\overline{\text{M}}$	14c	$\overline{\text{WRC}}$
15a	$\overline{\text{RDC}}$	15b	$\overline{\text{IQ}}$	15c	$\overline{\text{MQ}}$
16a	GND	16b	$\overline{\text{MON}}$	16c	$\overline{\text{BUSY}}$ [INHIBIT]

8.2.3.3.7 LTUX-Modules

Different LTUX-modules are available for the TDX system, such as the LTUX-S, LTUX-M-FE, LTUX-M-CPU, and a range of process interface modules of the LTUX family.

The performance characteristics of the LTUX-modules are given in section 8.2.2,3-4 which mechanical and interface specifications are given in the following sections.

All LTUX-modules are front modules fixed by means of a single finger screw.

8.2.3.3.7.1 LTUX-S

The LTUX-S (see fig. 8.2.3.3.7.1-1 overleaf)is a standard TDX interface module interfacing the TDX system to four physical communications lines, terminals, modems etc.

The LTUX-S which is a front module interfaces to the power bus, the LTUX-bus and the I/O-area of the LTUX motherboard connector by means of a 86 pin edge connector. The front panel contains besides LED's four switches to disable/enable each of the four line communication interfaces.

The interface specification of the edge connector is found in Table 8.2.3.3.7.1a.

For further information please refer to datasheet, chapter 10.

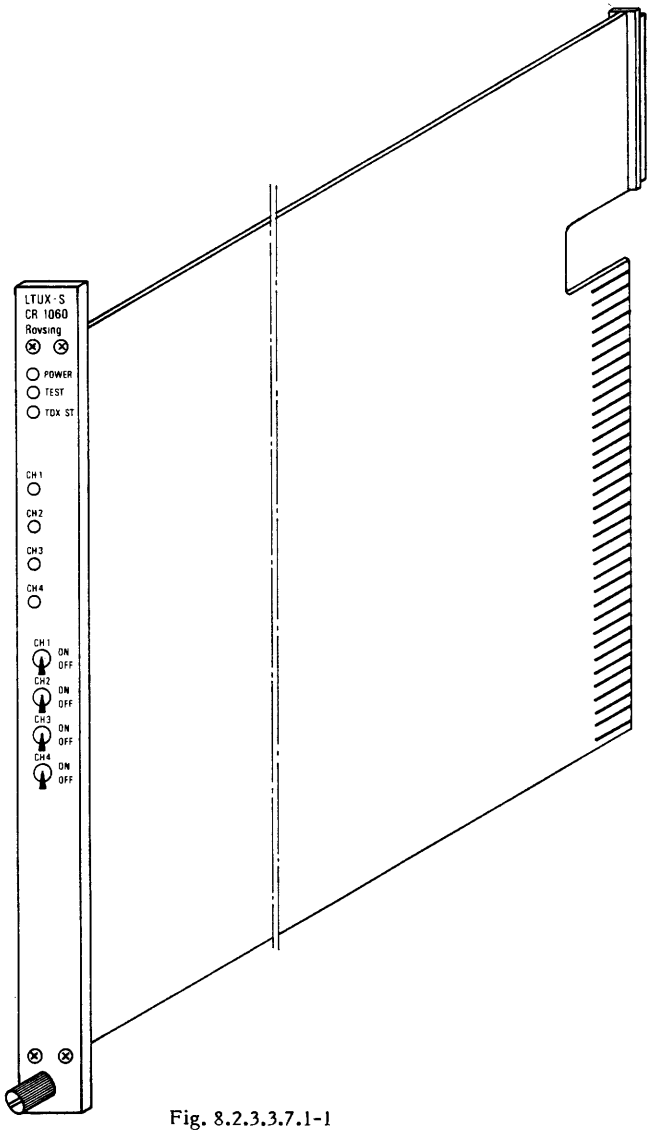


Fig. 8.2.3.3.7.1-1
LTUX-S
FRONT VIEW

Table 8.2.3.3.7.1a
LTUX-S
LTUX-BUS CONNECTOR, PIN LAY-OUT

	A			B
	GND		1	GND
CHANNEL 1	CIRC.NO.	101	2	CIRC.NO. 102
			3	113
			4	
		115	5	
		109	6	106
		105	7	108
		103	8	104
	Not used		9	Not used
CHANNEL 2	CIRC.NO.	101	10	CIRC.NO. 102
			11	113
			12	
		115	13	
		109	14	106
		105	15	108
		103	16	104
	Not used		17	Not used
	Not used		18	Not used
CHANNEL 3	CIRC.NO.	101	19	CIRC.NO. 102
			20	113
			21	
		115	22	
		109	23	106
		105	24	108
		103	25	104
	Not used		26	Not used
CHANNEL 4	CIRC.NO.	101	27	CIRC.NO. 102
			28	113
			29	
		115	30	
		109	31	106
		105	32	108
		103	33	104
	TDX TX DTA		34	TDX TX ENB
	TDX GND		35	TDX GND
	TDX REC1		36	TDX REC 2
	GND		37	GND
	+12V		38	+12V
	-12V		39	-12V
	+5V		40	+5V
	+5V		41	+5V
	GND		42	GND
	GND		43	GND

8.2.3.3.7.2 LTUX-M-FE

The LTUX-M-FE (Fig. 8.2.3.3.7.2 overleaf) is one module of the high performance TDX device called LTUX-M, which contains one LTUX-M-FE and one ore more LTUX-M-CPU's performing the line communication interfaces and the I/O-processing if process control interface modules are used.

The LTUX-M-FE which interfaces to the TDX-Bus, communicates with the LTUX-M-CPUs via the Micro Bus and a shared memory placed on the LTUX-M-FE.
The Micro Bus is interfaced via a 48 pin connector and the power Bus and LTUX-Bus of the LTUX-Motherboard connector is interfaced by means of a 86 pin edge connector.

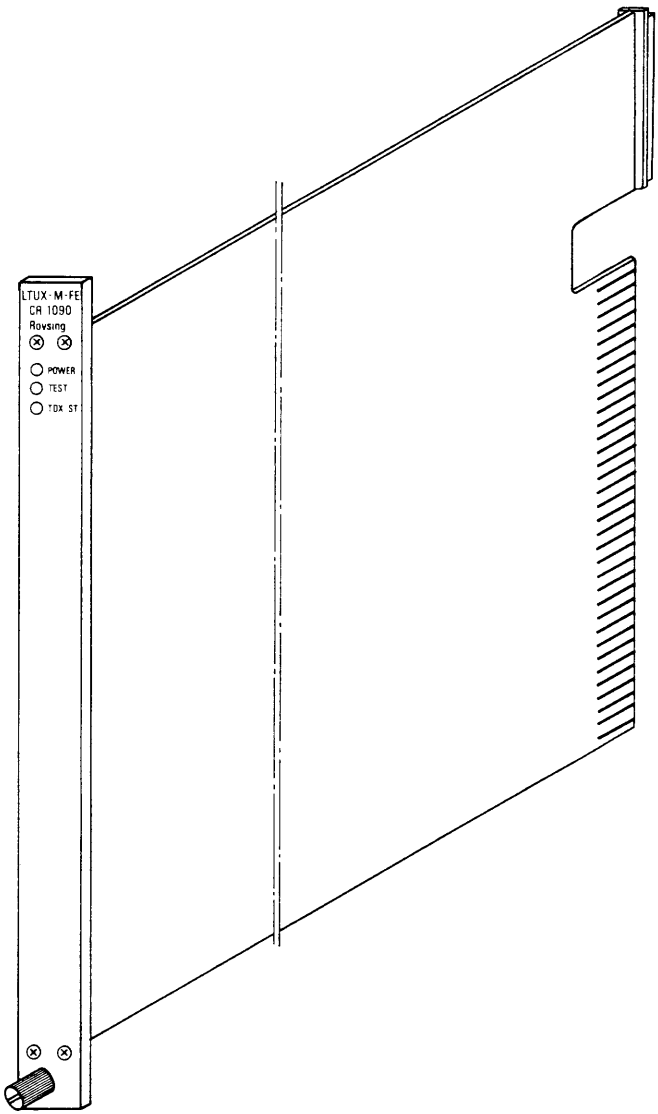


Fig 8.2.3.3.7.2 - 1
LTUX-M-FE
FRONT VIEW

8.2.3.3.7.3 LTUX-M-CPU

The LTUX-M-CPU (Fig. 8.2.3.3.7.3-1 overleaf) forms together with a LTUX-M-FE a high performance TDX-interface with four physical line communication interfaces of each LTUX-M-CPU (see Table 8.2.3.3.7.3a for pin layout).

The communication between the modules is done via the Micro bus on which the LTUX-M-CPU acts as a master, able to control the bus while the LTUX-M-FE is a bus slave module without any bus control.

The micro bus supports a multi-master configuration which means that more LTUX-M-CPU's may be installed with a single LTUX-M-FE.

For Process Control applications, Process Interface Modules may also be connected to the micro-bus.

The LTUX-M-CPU interfaces to the Micro Bus via a 48 pin connector and to the power bus and the I/O-area of the LTUX-Motherboard connector by means of a 86 pin edge connector.

The front panel contains besides LED's, four switches to enable/disable each of the four line communication interfaces.

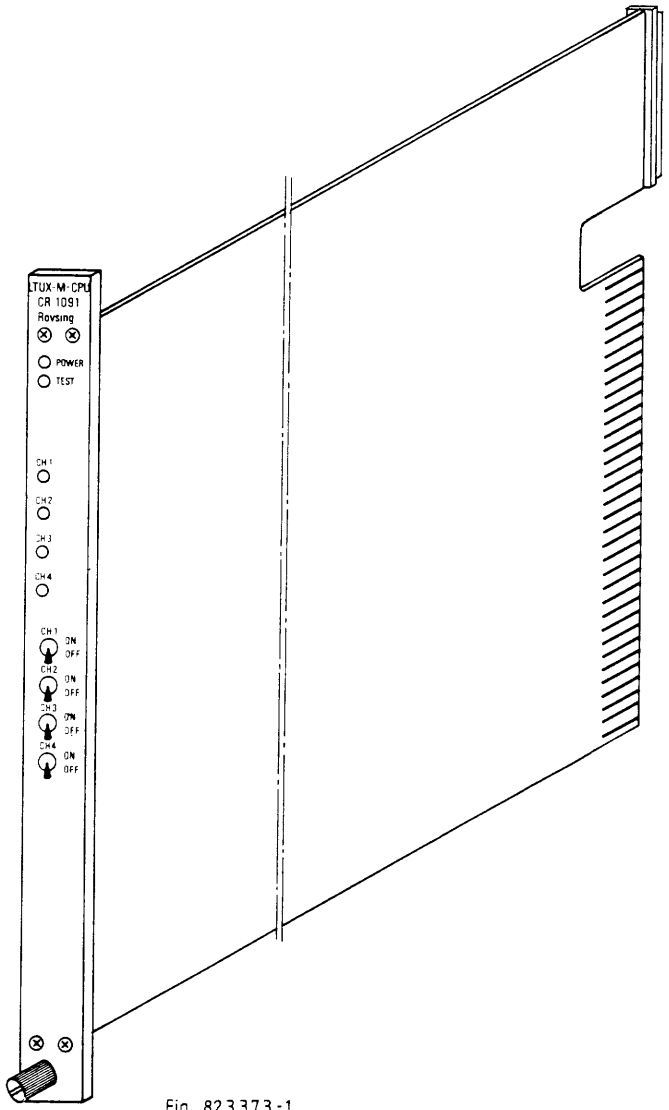


Fig. 82.3.37.3-1
LTUX-M-CPU
FRONT VIEW

Afsnit 8

Table 8.2.3.3.7.3a
CONNECTOR (LTUX-BUS) PIN LAYOUT FOR LTUX-M-CPU

		backpanel connector A	B
	GND	1	GND
CHANNEL 1	CIRC.NO. 102	2	CIRC.NO. 102
	111	3	114/113
	107	4	116
	115	5	117
	109	6	106
	105	7	108
	103	8	104
	Not used	9	Not used
CHANNEL 2	CIRC.NO. 102	10	CIRC.NO. 102
	111	11	114/113
	107	12	116
	115	13	117
	109	14	106
	105	15	108
	103	16	104
	Not used	17	Not used
	Not used	18	Not used
CHANNEL 3	CIRC.NO. 102	19	CIRC.NO. 102
	111	20	114/113
	107	21	116
	115	22	117
	109	23	106
	105	24	108
	103	25	104
	Not used	26	Not used
CHANNEL 4	CIRC.NO. 102	27	CIRC.NO. 102
	111	28	114/113
	107	29	116
	115	30	117
	109	31	106
	105	32	108
	103	33	104
		34	
		35	
		36	
	GND	37	GND
	+12V	38	+12V
	-12V	39	-12V
	+5V	40	+5V
	+5V	41	+5V
	GND	42	GND
	GND	43	GND

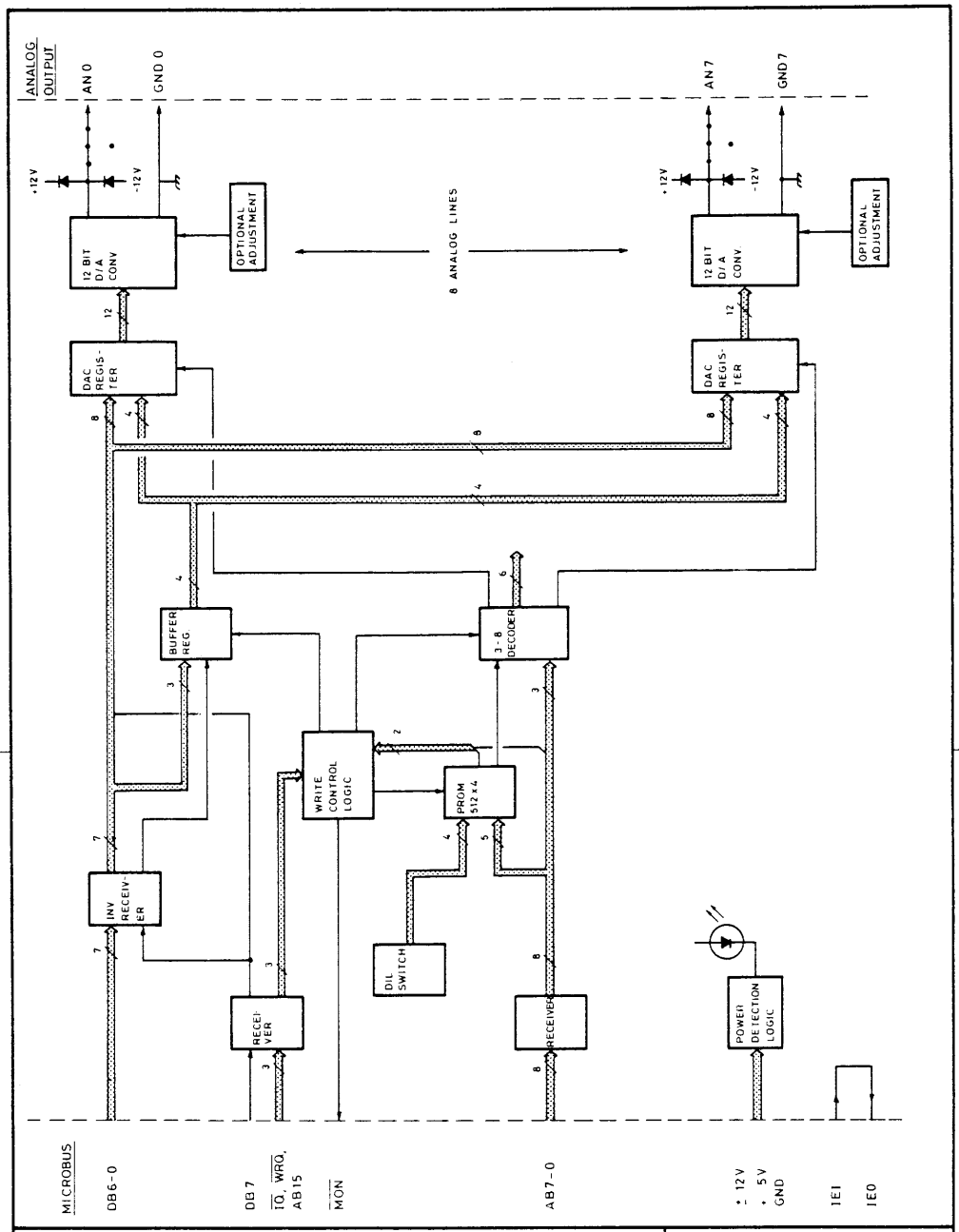
8.2.3.3.7.4 Digital to Analog Converter

This section describes the DAC I/F (Fig. 8.2.3.3.7.4-1) - Digital to Analog Converter which interfaces to the Micro-bus and with up to 8 Analog outputs.

Two versions are specified.

DAC-X,	CR1094S/000--/00	Two Channel D-A Converter.
DAC-X,	CR1094S/010--/00	Eight Channel D-A Converter.

- The one module DAC I/F board contains 12 bit D/A converters.
- The DAC I/F has normally two converters. Optionally eight converters are mounted.
- Without gain and offset adjustment the accuracy (including temperature drift) is 8 bit in worst case.
- With adjustment the accuracy is 10 bit in worst case.
- The outputs are short circuit protected.
- Two modes of addressing can be used, direct addressing and register addressing.



CHRISTIAN ROVSING A/S

DAC-X, CR1094

Digital to Analog Converter

Block Diagram Fig. 8.2.3.3.7.4-1

State		Approved	Issue 1
Date	79-11-29	E	Date 79-11-29
Drawn	AM / AMS		Approved
Parts no		3-2492	
Print no		Sheet 1 of 1 sheets	

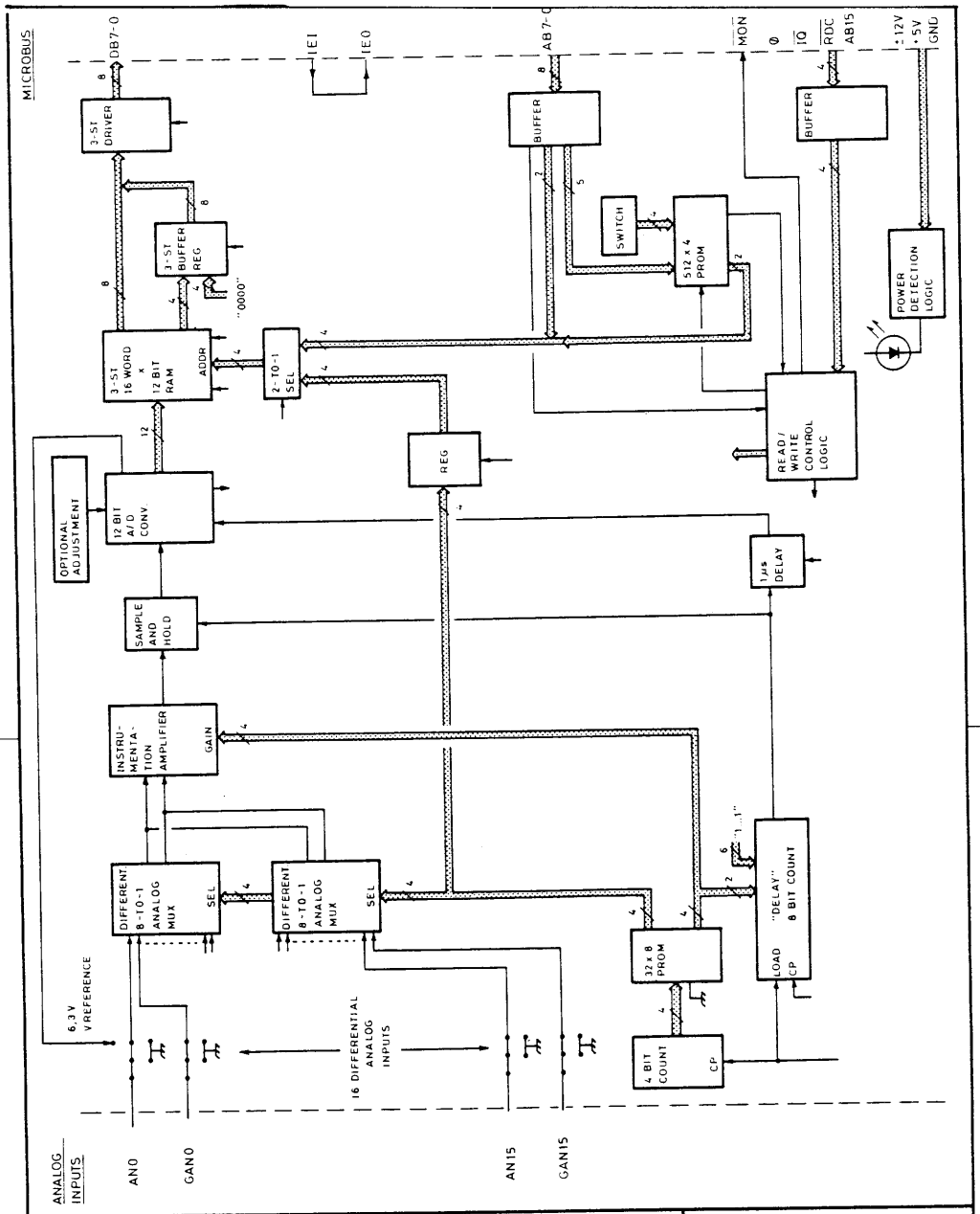
8.2.3.3.7.5 Analog to Digital Converter

The ADC-X Module (Fig. 8.2.3.3.7.5-1) is capable of receiving up to 16 differential analog signals. These signals are multiplexed to a programmable gain instrumentation amplifier. The sampling order and gain of the 16 analog signals are programmed in a PROM. The amplifier is then followed by a sample/hold circuit and a 12 bit A/D converter.

The analog inputs are continuously sampled and the digital value is updated in a 16 x 12 bit RAM. This RAM is readable from the micro-bus.

By means of straps it is possible to connect a reference voltage of 6.3V to analog inputs AN1 and AN0 respectively.

In this way offset error and gain error are software correctable during operation. A precision equal to the reference voltage precision is thereby achieved.



CHRISTIAN ROVSING A/S

ADC - X, CR1093

Analog to Digital Converter

Block Diagram Fig. 8.2.3.3.7.5-1

Scale		Approved	Issue 1
Date	79 - 11 - 29	E	Date 79 - 11 - 29
Drawn	AM / AMS		Approved
Parts no		3-2493	
Print no		Sheet 1 of 1 sheets	

8.2.3.3.7.6 Parallel I/O

The PIO-X (Fig. 8.2.3.3.7.6-1) is the general purpose Parallel In/Output module of the X-module family.

Communication between LTUX-masters and a PIO-X is via the Micro-bus. The parallel I/Os are connected by the standard I/O area of the LTUX bus connector.

Th PIO-X is capable of fullfilling nearly any application requirements for low power on/off controlling:

- 32 parallel input output lines
- Directions of all I/Os are software controlled
- Direction is selectable for groups of 8 in/output
- All inputs can be equipped with opto couplers (normal or low current type)
- All outputs can be equipped with reed Relays
- Each I/O has an adaptor field which makes it able to fulfil all interface requirements for low power controlling. As standard is following available:

CR1092S/000/00	32 TTL input
CR1092S/002/00	32 TTL input
CR1092S/003/00	32 TTL input
CR1092S/004/00	32 TTL input
CR1092S/005/00	32 TTL input
CR1092S/006/00	16 TTL input/16 TTL output
CR1092S/007/00	24 TTL output/8 TTL input
CR1092S/008/00	32 TTL output

CR1092S/008/00	32 TTL output
CR1092S/010/00	16 Relay output/16 Opto input
CR1092S/011/00	16 Relay output/16 Opto input
CR1092S/012/00	16 Relay output/16 Opto input
CR1092S/013/00	16 Relay output/16 Opto input
CR1092S/020/00	32 Opto input
CR1092S/021/00	32 Opto input
CR1092S/022/00	32 Opto input
CR1092S/023/00	32 Opto input
CR1092S/030/00	32 Relay output
CR1092S/031/00	24 Relay output/8 TTL input.

- PIO-X works in two addressing modes Register I/O addressing using all 16 addr. lines and Port addr. only using 8 addr. lines.

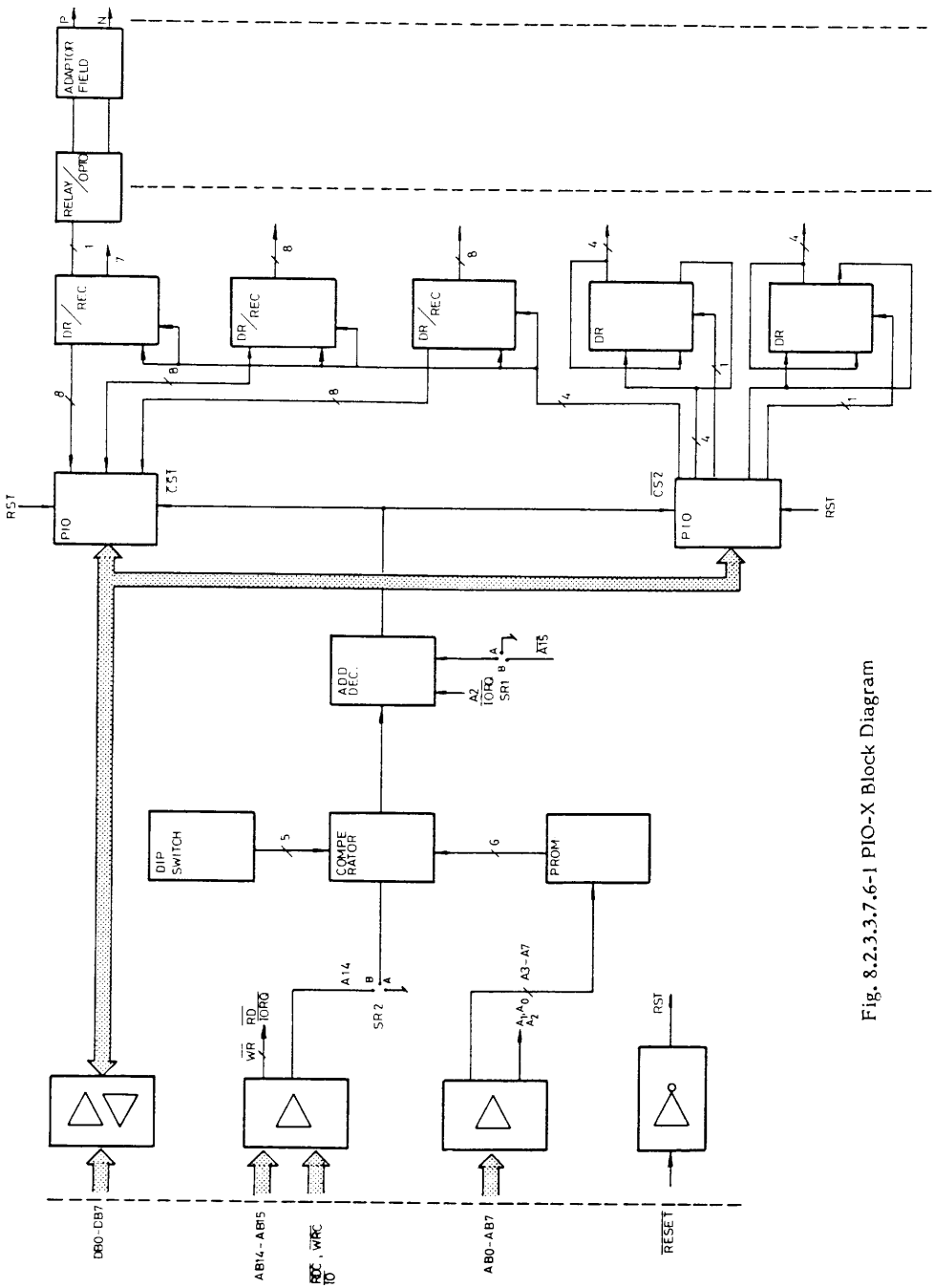
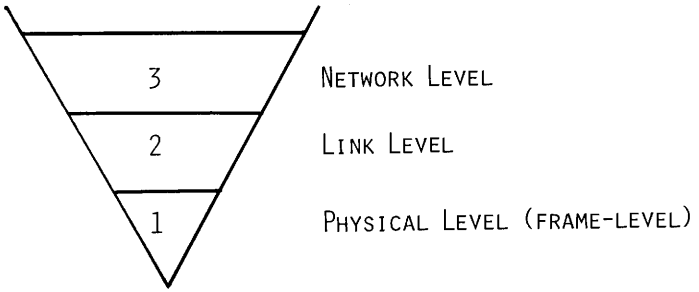


Fig. 8.2.3.3.7.6-1 PIO-X Block Diagram

8.3. TDX System - Protocol Specification

The architecture of the TDX Protocol is in analogy with the ISO Open System Model. The packet protocol is able to transfer packets with sizes from 0 bytes to 64K bytes. Data transferred are chopped into frames of 16 databytes encapsulated with protocol information in a HDLC-frame. The frames are decapsulated and collected via link-level, which performs the error-detection and correction.

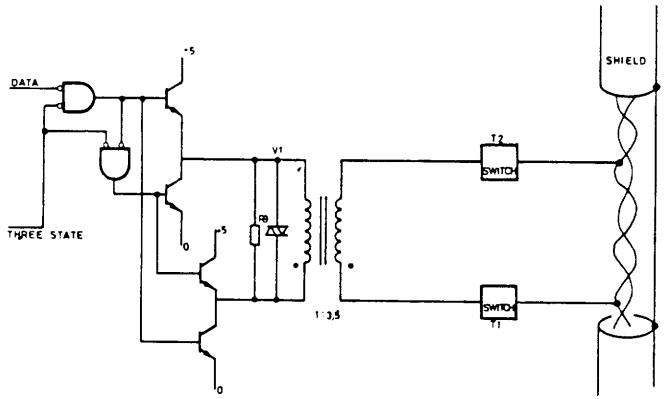


8.3.1 Physical Level (level 1)

The communication medium of the TDX Bus is two shielded twin-leaded cables. The interfaces to the cables are a balanced split phase driver and a differential receiver contained in the Wall Outlets, see figure 8.3.1-1 overleaf, distributed along the Cables.

The balanced split phase driver is capable of switching 12 volt over the transmission line load in less than 50ns.
The driver can be tristated against the twisted line pair by the MOS switches at the output, showing an impedance of 25K ohm in parallel with 50 pf for each switch against the line at 10 Mhz.

FLOATING THREE-STATE DRIVER:



FLOATING RECEIVER:

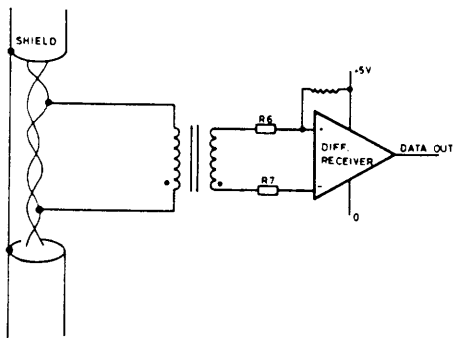


Figure 8.3.1-1 TDX WALL OUTLET TRANSCEIVER

The output of the driver is short circuit protected by on-resistance of MOS switches, transformer core characteristic and signals sensing circuit at the output.

The differential receiver has a worst case differential mode switching band of +/-40mV, which corresponds to +/- 120mV at twisted line side of line transformer. Both driver and receiver circuits are floating in relation to the TDX-cable, separated by galvanic isolation, making the circuits withstanding high common mode voltage pulses and DC voltages between TDX-cable and local ground of attached devices. The driver/receiver circuit is completely powered by + 5V and the transmitter is tristated against the cable when no power is applied.

Conservatively specifying +/- 240mV as safe min. value for switching of receiver (corresponding to 6 dB safety factor) the maximum tolerable attenuation between driver and receiver is:

$$ATT_{max} = 20 \log \frac{12}{0.24} = 34 \text{ dB}$$

At 2 Mhz (>1.8432 Mhz of bit clock) the attenuation of:

TM3078 cable:	2.18dB/100m
RG 111 A/U or RG22 B/U cable:	1.3 dB/100m

Attenuation by a three stated driver or a receiver, with impedance R_p between the twisted wire pair:

$$ATT_p \leq 10 \log (1 - \frac{8xZ_0}{R_p})$$
$$\leq 10 \log (1 - \frac{800}{24x10^3})$$
$$\leq 10 \log (0.96666)$$
$$\leq -0.147 \text{ dB}$$

Allowing 0.103 dB for mismatch etc. 0.25 dB is used for insertion loss pr. Wall Outlet (XWO) attached to the TDX-Cable. For 64 XWOS we have the total insertion loss:

$$ATT_p = 64 \times 0.25 \text{ dB} = 16 \text{ dB}$$

and maximum cable section length (L max.) with 64 XWO's attached to the TDX-Bus:

$$L_{max_{TM3078}} = \frac{34-16}{2.18} \times 100 \text{ m} = 825 \text{ m}$$

$$L_{max_{RG111A/U}} = \frac{34-16}{1.3} \times 100 \text{ m} = 1385 \text{ m}$$

From which follows the specification of max section length:

Lsect TM3078:	800 meter
Lsect. RG111 A/U:	1300 meter

And for Point to Point Links (no devices attached to cable:

LPP TM3078:	1500 meter
LPP RG111 A/U:	2500 meter

Clock Encoder and Decoder

Data and clock are communicated on the TDX Bus using the self-clocking differential split phase code SPL-D. SPL-D changes polarity at start of each bit cell, and in the middle of bit cells to indicate a zero. This allows data and clock to be transmitted via the same set of wires. Phase ambiguity is solved by the decoder when a "one" is inputted.

Therefore, on the lower bus where SPL-D is continuously transmitted, phase ambiguity is automatically solved. On the upper bus, where transmission is discontinuous, the ABORT byte (all ones) preceeding each frame solves the phase ambiguity.

Transmitting clock together with data on the busses have several advantages. The major one being that the bus delay has no significance for the correct decoding as clock and data are delayed equally. Also, no need exists to synchronize a transmitter and receiver.

Also that the TDX controller clock is continuously transmitted on the lower bus makes this available to all devices. One application for example being to use this clock for modems attached to the TDX-Bus thereby eliminating the problem of modem clocks being slightly out of phase, and the difference in transfer speed is thereby slowly accumulating in data buffers until overflow.

The following figures (8.3.1-2,3) show circuits and waveforms of the SPL-D encoder and decoder.

Frame Management

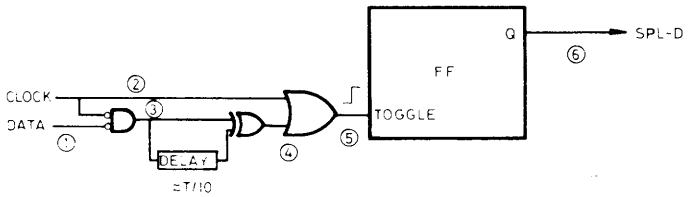
Transfer of frames between link-levels is carried out by means of two procedures, which handle all communication between the physical level and the link-level.

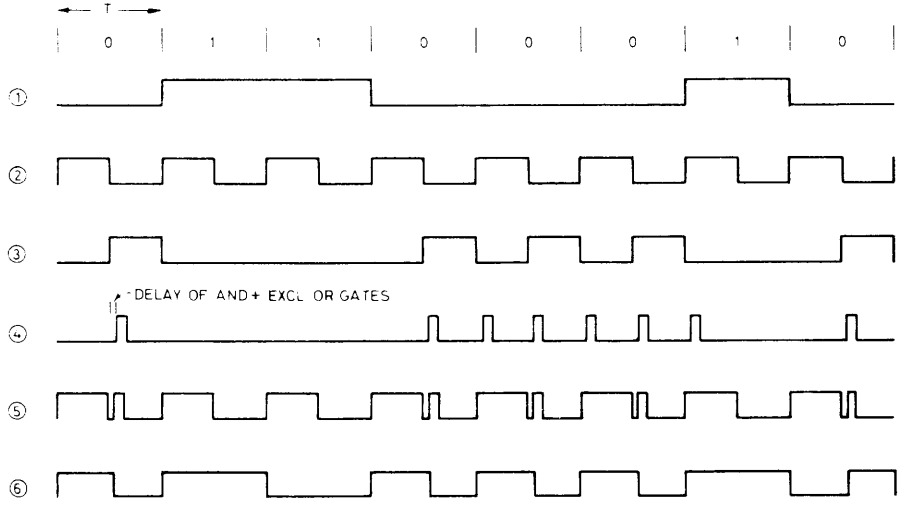
o Transmitter Procedure

PROCEDURE TFRAME (INTEGER D,H,T,S,Z,W; STRING (16))

This procedure encapsulates data and protocol information in a HDLC-frame and executes the transmission to the network. All parameters are inserted by higher levels and are declared as follows:

D,H,T	is CR-ID (see 8.2.1.3)
S	is link-protocol header
Z	is sequence no. (0-7)
W	is bytecount
STRING(16)	is 16 databytes of which some or all may be dummy.





CHRISTIAN ROVSING A/S

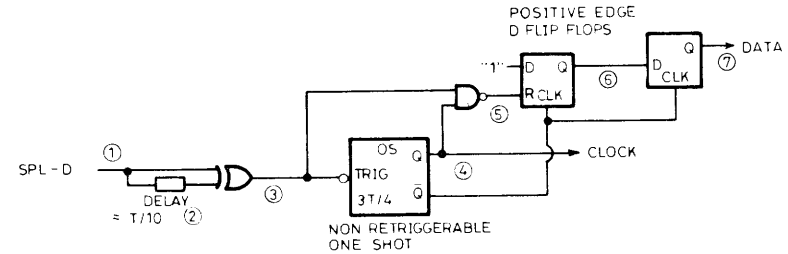
TDX-BUS Fig. 8.3.1-2
SPL-D Clock & Data Encoder
Circuit & Waveforms

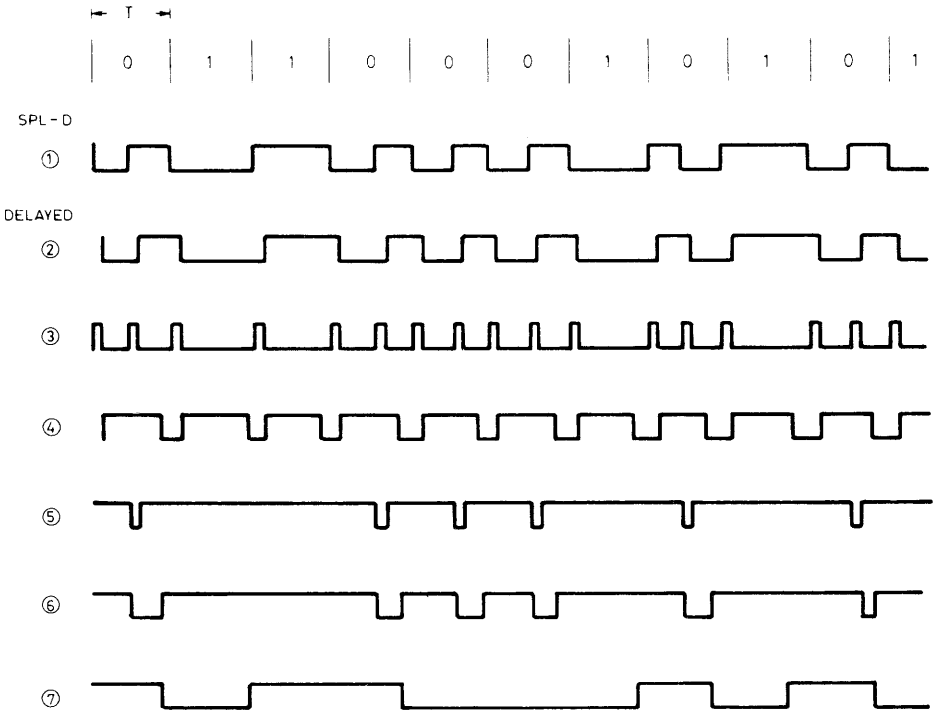
78-05-26
ASM/AMS

E 464

40-259

78-05-26





CHRISTIAN ROVSING A/S

TDX - BUS Fig. 8.3.1-3 SPL - D Clock & Data Decoder Circuit & Waveforms	78-05-26 ASM / IDA	40 - 258 1 1	1 78-05-26

- Receiver Procedure

PROCEDURE RFRAME (STRING (21))

This procedure receives and decapsulates all frames transmitted in the network. If the first character in the string is identical to the unique TDX device no. the transmitter procedure is allowed to be executed. This character is inserted by the controller and is called a MUX-No.

The CR-ID is used in the routing algorithm to select the correct datastream in the next higher level.

8.3.2 Data Link Level (level 2)

The Link Level protocol controls the data traffic on each datastream in the network. When data is delivered from link level to higher levels, it is guaranteed error-free by the CRC-16 bit check of the physical level and the Link Level Protocol. The TDX-protocol is served autonomously for as many data streams as set up by the packet level.

Network Level to Link Level interface

The two primary services provided to the network level by the data link level are transmission and reception of data-packets. The network level uses these services through a pair of routines.: Transmit packet and Receive packet. By call of the routines a reference to a data buffer is made. The data link level returns a status code in the data-bufferheader, which indicates if transmission/reception is O.K. or an unrecoverable transmission error has occurred.

The network level transmits a packet by calling:

Transmitpacket (LINE, BUFFADR.)

LINE is a number, which identifies the logical line.
BUFFADR is the address of a bufferheader, which is associated with the data-buffer containing the packet to be transmitted.

The reception of a packet is enabled by calling:

Receivepacket (LINE, BUFFADR.)

LINE identifies the logical line by a number.
BUFFADR points at the bufferheader, which is associated with the data buffer to contain the received packet.

The packet level detects the completion of reception or transmission by calling the two routines:

Return T Buffer (LINE, BUFFADR)
Return R Buffer (LINE, BUFFADR)

LINE is the identifying number of the logical line.
BUFFADR is an output parameter, which refers to the data buffer containing the packet. In the bufferheader, a status-code indicates either successful transmission/reception or an error, which was unrecoverable by the datalink level.

Link Level Protocol - Detailed

The functions of the protocol is determined by two state-event action diagrams. One for the routine that receives data from the TDX-Bus called the ingoing protocol routine and the other which determines functions of the outgoing protocol routine, which sends data to the TDX-Bus.

The smallest data unit the protocol works with, is the Frame. From a protocol point of view a frame contains one protocol byte, a 3 bit sequence number, a 5 bit byte count and between 0 and 16 data bytes.

The retransmission units of the protocol are called Packets. TDX packets may contain data from several frames. The size of a packet is determined by the application programs in the outputting device.

To transmit a packet over the TDX-Bus without error, between two ready TDX Devices, packet control and status information must be sent in both directions of the connection.

For each packet there is an "outputter" at the end of the link that originally had the packet and an "inputter" at the other end that finally gets the packet. These terms must be distinguished from "sender" and "receiver" which are used in the conventional way to distinguish the two ends of a single transmission, for a "control byte" sent in one direction may contain information about a packet output in the other direction.

Because the link is full duplex (but not necessarily the same speed in each direction) for output and input as well as for sending and receiving, two packets may be transmitted independently and simultaneously in opposite directions over the link.

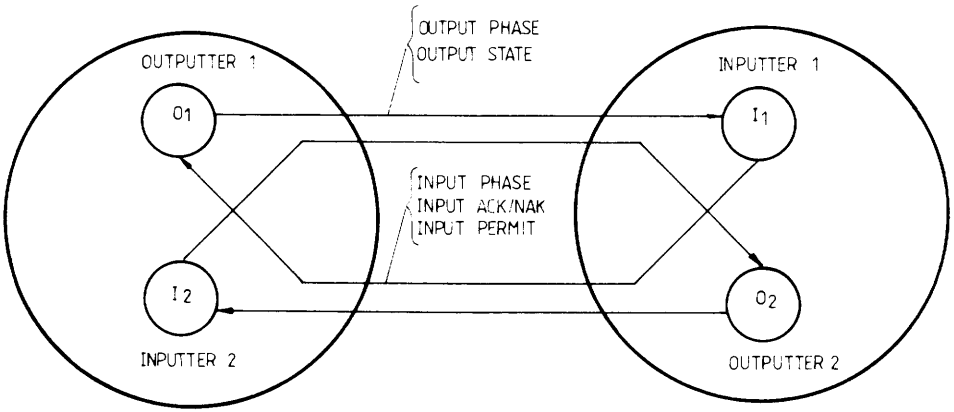
A packet consists of one or more TDX frames. The frames within a packet is contiguously numbered modulo 8, starting with zero for the first frame in the packet. The first and the last frame in the packet contains a communication control byte (shown overleaf) with "return to inputter valid bit" set which indicates the begin and end of the packet (output state) as well as the output phase (0 or 1) of the packet.

Completeness of a received packet is ensured by the inputter through contiguous numbered (modulo 8) frames between the first and the last frame in the packet. All errors will result in frame being rejected at the receiving station, errors are therefore detected by the protocol by the following frame arriving out of sequence (or time out, if last frame in packet), the complete packet is rejected immediately (without waiting for completion of packet) and retransmission requested by replying NAK. The packet phase indicated in the first and last frame of a packet is also checked, if it is n+1 (modulo 2), where n was the last packet accepted it is accepted via ACK. If it is n, it is accepted via ACK, but it is thrown away by the inputter station since it must have been a duplicate transmission caused by an error in one of the acknowledgements.

ACK/NAK is transmitted by the inputter by setting the "Return to outputter valid bit" in the communication control byte. The communication control byte is transmitted either contained in a frame in a packet going from the inputter (any frame (first, last or in the middle of packet) as this have no influence on the validity of the "return to inputter part" of comm. control byte) or initiates sending of a frame, outside packets, containing no data and which is discarded by the inputter except for the content of the "return to outputter bits" of the comm. control byte. This allows for immediate ACK/NAK response by the inputter, to a packet received from the outputter.

Fig. 8.3.2-1

PROTOCOL - CONTROL FLOW THROUGH A TDX - CONNECTION



At the transmitter each acknowledgement is also checked for errors, if erroneous it is thrown away, if error free and ACK, then packet n+1 is transmitted, and if NAK then packet n is retransmitted. A timer at the transmitter initiates retransmission of packet n in case that neither ACK nor NAK is received within a specified Ttime (e.g. lost due to error of the link, transmission is attempted 3 times before the protocol gives up on output.

Provision is also included for transmission of short packets in a single frame and for synchronizing packet phase between both ends of the link.

Packet Communication Control Byte

Three of the eight bits in the control byte (Fig. 8.3.2-2) are input status (bits 6, 5 and 4), valid content indicated by bit 7 set, and three are output status (bits 2, 1 and 0) valid content indicated bit 4 set.

Bit 6, the input phase bit, and bit 2, the output phase bit, indicate node phases. At any given time, a node is in one of two phases (0 or 1) associated with input. In each case, the phases alternate as successive packets are transmitted. Change in a node's output phase occurs whenever it sends a packet, while the input phase is always made to conform to the output phase most recently received (without error) from the other node. A phase is essentially a modulo-2 packet counter (counting 0, 1, 0, 1, ...) used to prevent confusion about which packet's status is being indicated.

The eight bits of a control byte are explained from high to low order.

Bit 7 - Return to outputter valid bit - when set indicates valid content of bit 6, 5 & 4 (Input status).

Bit 6 - input-phase bit - indicates the input phase of the sender.

Bit 5 - input-acceptance bit - is set if, and only if, the packet input by the sender during its current input phase was accepted, that is was received without error and forwarded for further processing according ot higher level protocols. A packet is received with error if it is part of an erroneous transmission (a transmission is judged to be erroneous and its contents ignored if the frame protocol detects an error) or is accompanied by an improper output state (defined in discussion of bits 1 and 0).

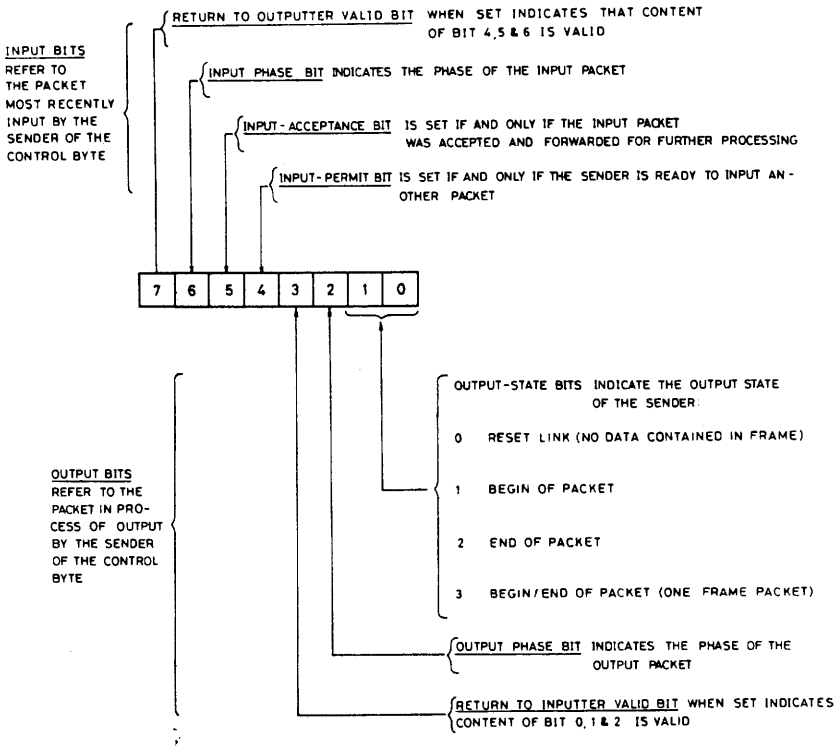
Bit 4 - input-permit bit - is set if, and only if, the sender is prepared to input a new packet.

Bit 3 - Return to inputter valid bit - when set indicates valid content of bits 2, 1 & 0 (output status).

Bit 2 - output-phase bit - indicates the output phase of the sender.

Bits 1 and 0 - the output-state bits - indicate the state of the sender with regard to output.

Output state = 00:
If the output-state bits are equal to 00, indicating the RESET LINK state, the control byte accompanies a single frame packet (frame seq. no. = 0), with or without data. The purpose being to force synchronization of the inputter phase to the phase of the outputter (indicated in bit 2, Output phase bit).



NOTE CONTROL BYTE WITH OUTPUT BITS IS INCLUDED IN FIRST AND LAST FRAME OF EVERY PACKET TRANSMISSION.
CONTROL BYTE WITH INPUT BITS CAN BE INSERTED IN ANY FRAME
PHASE IS MODULO 2 COUNTER (0,1,0,1 ..) OF TRANSMITTED PACKETS

CHRISTIAN ROVSING A/S

TDX-BUS Fig. 8.3.2-2
Packet Level Protocol
Format of Control Byte

Scale		Appoved		Issue	1
Date	79-01-23			Date	79-01-23
Drawn	ASM/IDA			Approved	
Parts no		3-1911			
Print no		Sheet 1 of 1 sheets			

The inputter will respond with ACK uppon successful reception and set its output phase to the opposite of that received. In case of NAK or time out at the Outputter (failure to receive ACK or NAK within a specified time) retransmission is initiated up to three times by the outputter.
Output state 00 is utilized either for initialization of link or to check for input permission from the inputter.

Output state = 01:
If the output-state bits are equal to 01, indicating that the control byte accompanies the first frame (frame seq. no. = 0) in a packet consisting of two or more frames.

The inputter checks that the control byte shows a change in the sender's output phase from the previous successfully received packet, which has been remembered as the receiver's input state, if this is not the case, the receiver responds with ACK (input acceptance bit 5 = 0) and discards of following frames until receiving a new start of packet, as the packet start just received must be part of a retransmission by the outputter due to failure of receiving acknowledge (retransmission initiated by "time out" at outputter) or an ACK erroneous received as NAK.
In the following data frames (not containing a valid "return to inputter" part in comm. control byte) until receiving end of packet (output state 10), the inputter check for contiguous frame sequence numbering (modulo 8), in case of out of seq. error the inputter responds with NAK and discards of previously received data in packet.

Output state 10:
If the output state bits are equal to 10, indicating that the control byte accompanies the last frames, the inputter checks that the control byte shows a change in sender's output phase from previously successfully received packet, if not response is ACK as for output state 01 (packet start). If phase is changed from previous received packet, inputter checks that the corresponding packet start (output state 01) has been received and that frame seq. no. has been incremented by one from previous frame,

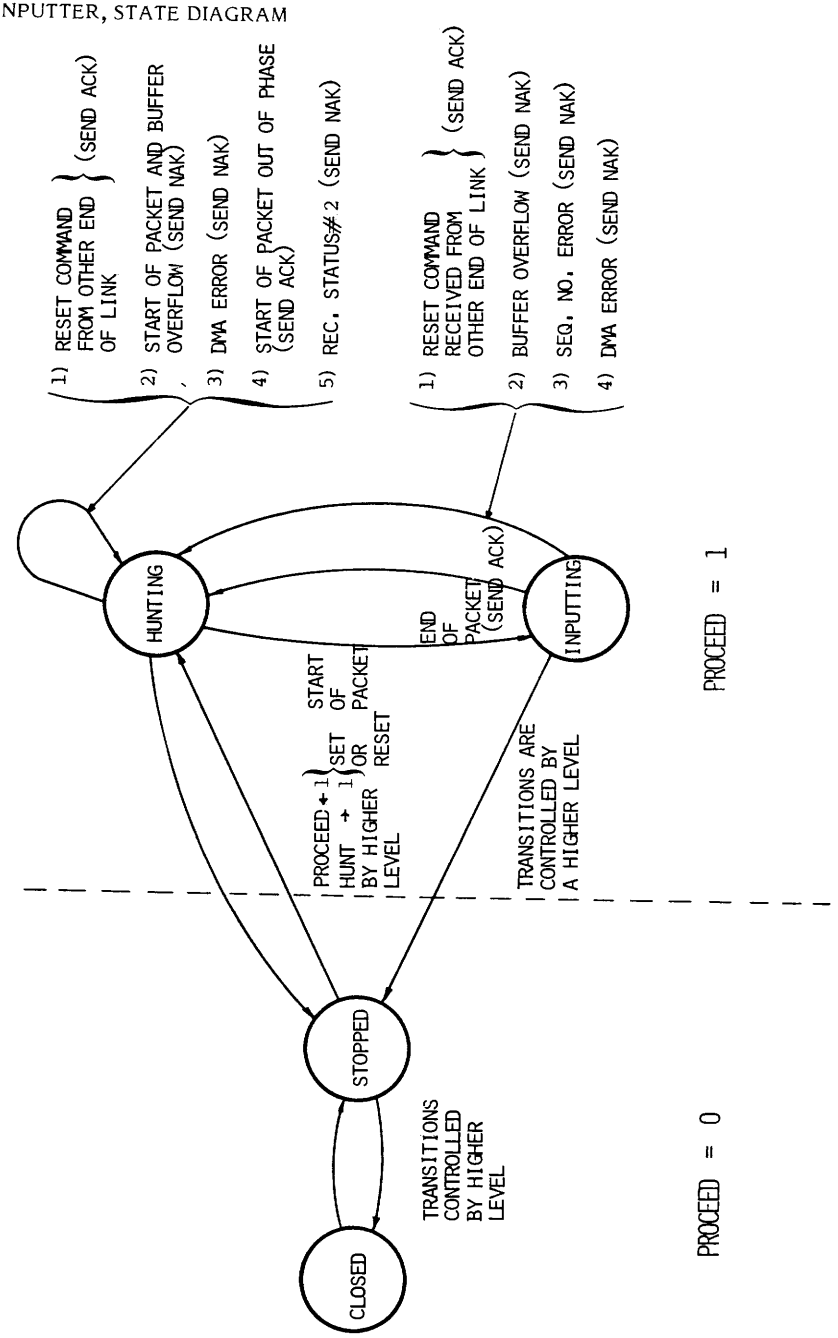
if not inputter responds with NAK, otherwise ACK is returned to outputter to indicate successful reception of packet and receiver changes its input phase.

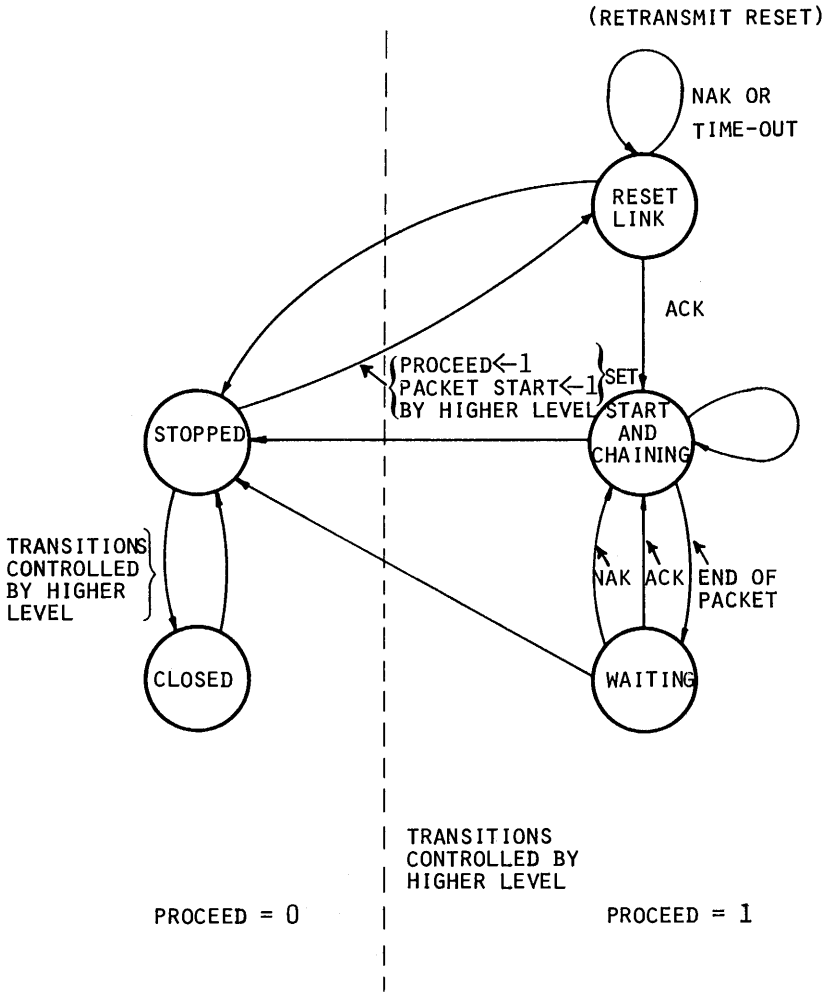
Output state 11:
If the output state bits are equal to 11, indicating that the control byte accompanies a packet consisting of a single frame, the inputter checks that the control byte shows a change in sender's output phase from the previous successfully received packet, which has been remembered as the receiver's input state, if this is not the case, the receiver responds with ACK, and discards the data contained in frame (packet), as the packet just received must be part of a retransmission due to failure of acknowledge. If phase is changed from previous received packet and frame seq. no. = 0 the receiver responds with ACK to the outputter, to indicate successful reception of packet and receiver changes its input phase.

Inputter & Outputter State Diagrams

Find overleaf (Fig. 8.3.2-3,4) the state diagrams of the inputter and outputter parts.

Fig. 8.3.2-3
INPUTTER, STATE DIAGRAM





OUTPUT STATE:

00 RESET LINK

01 START OF PACKET

10 END OF PACKET

OR 11 START/END OF PACKET

Fig. 8.3.2-4
OUTPUTTER STATE DIAGRAM

Timers:

The Link Level Protocol has three integrated timers which is adjustable from the application level:
Transmission-timer, Receive-timer and Flow-control timer.

The Transmission-timer (resident in the outputter part of the protocol) is started by transmission of the first frame in a packet and stopped by reception of an ACK of the packet. If the timer runs out, a retransmission is executed, and after three retransmissions, an error message is transferred to higher levels. The Receive-timer (resident in the inputter part of the protocol) is started by reception of the first frame in a packet. If "end of packet" has not been received before the Receive-timer runs out, an error-message is transferred to higher levels.

The Flow-Control timer common for the entire TDX-device is activated by reception of "input not permit" (received if the remote inputter has run out of empty buffers) and reset by reception of "input permit". As the inputter part of the protocol transmits "input permit" by reception of an empty packet from higher levels (provided "input not permit" has previously been transmitted) a flow control function is achieved.

Notice: An awakening of the outputter part of the protocol by reception of an "input permit" is enforced by transmitting a NAK (from the remote inputter), whereby the number of remaining retransmissions is decreased by one.

The Transmission-timer should have a value large enough to run while a packet of maximum size is transmitted from the outputter or an acknowledge is transmitted from the inputter. The Receive-timer should be large enough to run while three retransmissions are attempted by the sending device.

The timer-units are 8 msec for both LTUX-M and LTUX-S.

8.3.3 Network Level

The primary service of the network level is to setup datastreams between TDX Devices. The interface through which the application level uses the facilities of the network level consists of the routines: OPEN and CLOSE. These routines respond with a status code indicating if the operation succeeded or a failure has occurred. The operations are synchroneous, in the sense that each routine-call must be completed before a new request may be served. The next higher level sets up a line by calling:

OPEN (DEST, SOURCE, TXTIMER, RXTIMER)

DEST is the CR-ID of the remote end of the line:

SOURCE is the CR-ID of the local (subscribing) device:
TXTIMER is the Transmitter-timer (see 8.3.2 about use of timers)
RXTIMER is the Receiver-timer (see 8.3.2 about use of timers)

A logical line is removed by application level by calling:

CLOSE (LINE), where
LINE is the number of the logical line.

Another service of the network level is to request the TDX Controller to change the bandwidth assigned to the actual station on the TDX Bus. A request is made by application level by calling:

REQBW (t, LEVEL), where

t: is the Device Address (0 to 255)

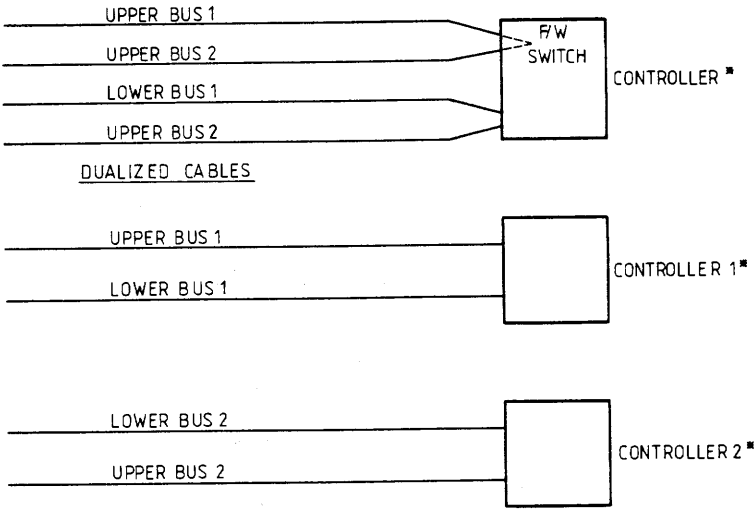
LEVEL: is a number between 0 and 14, which represents baudrates on: 0, 100 baud, 200 baud, 400 baud,...,800K baud.

The network level responds with a status code indicating if the request is accepted or rejected by the controller.

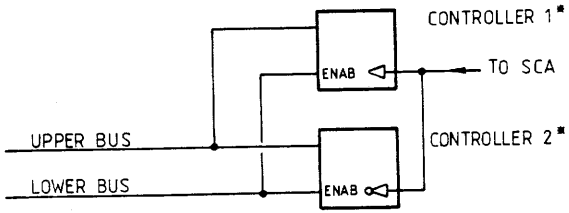
A channel is setup by sending a datagram containing the subscribing CR-ID and the remote CR-ID. The remote device reassigns the CR-ID of channel 0, opens the requested channel if it is free (closed) and responds with a datagram containing the CR-ID of the new-opened channel together with an ACK-status code. If the remote requested channel is already open or under opening a datagram containing an "Not-accepted"-status code is responded.

8.4 Fault Tolerance

The TDX Bus system highlights on a fault tolerant transmission path where all critical components can be dualized and monitored.
The TDX Bus cables can be dualized in a system including one or two TDX Controllers.



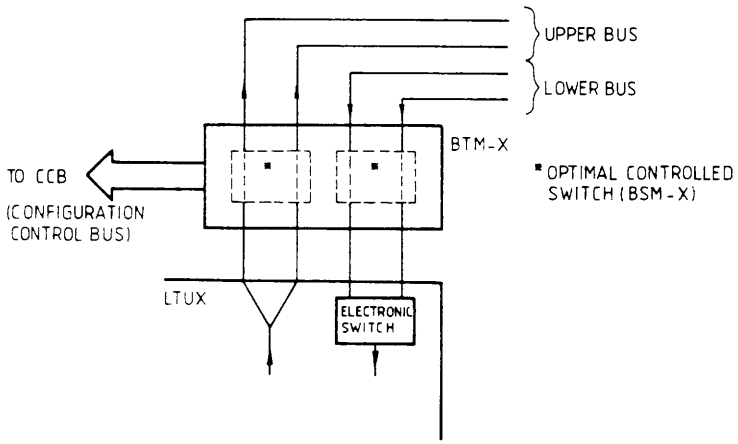
Dualized Cables and Controllers



Dualized Controllers

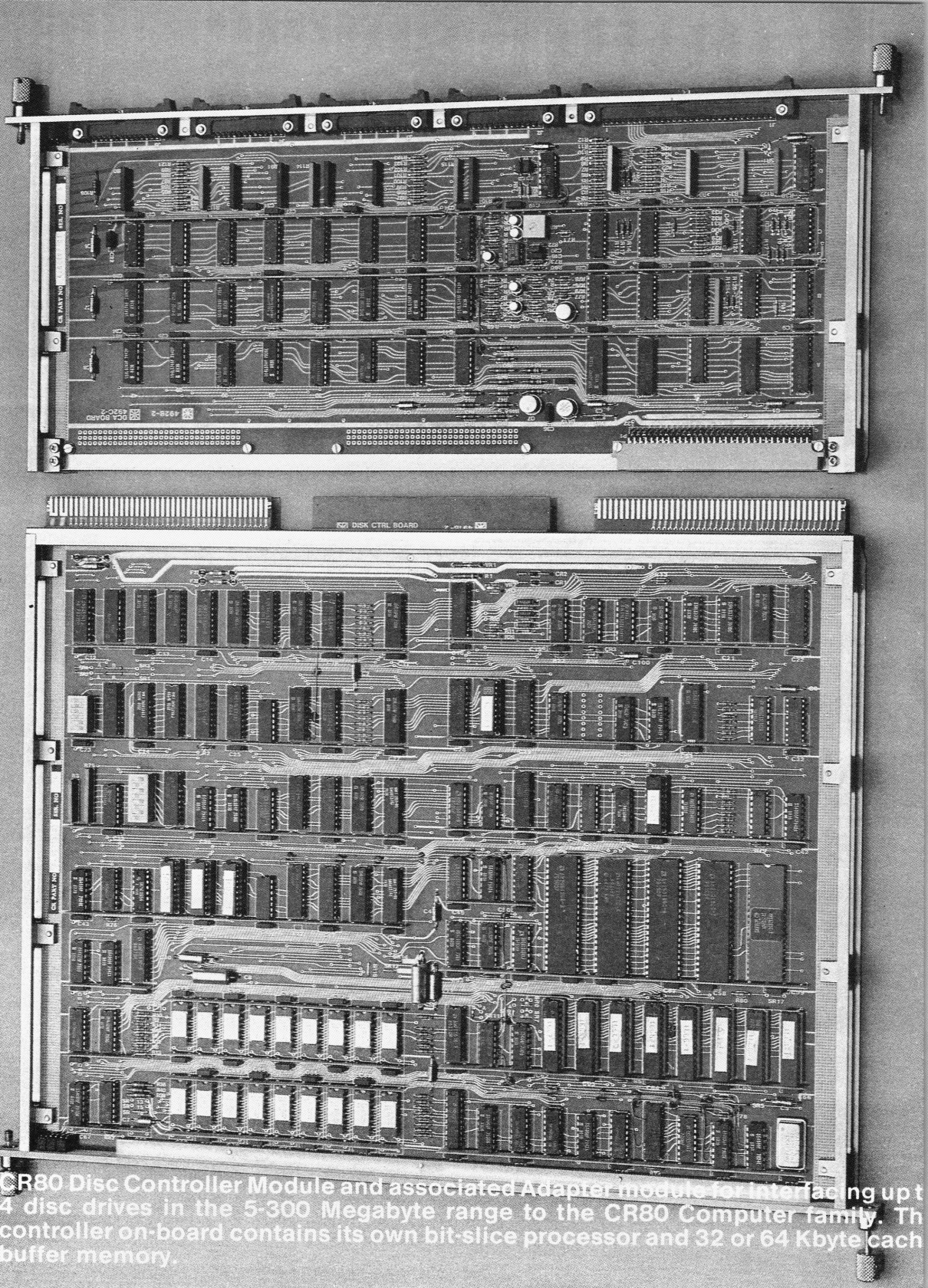
* The TDX Controller may be monitored via a SCA (Special Configuration Adapter). See chapter 7.

The selection of buses in a dualized configuration can be achieved by either electronic switch of lower buses (dependent on clock-appearance on the lower buses) and transmission on both upper buses (BTM-X concept) or switch on command from the Configuration Control Bus via the BSM-X.



BUS-Select via BTM-X / BSM-X

CR80 MINI (Model 831) Computer is particularly suited for medium size Technical or Communication applications. The snap-on front cover is easily removed during maintenance to give access to plug-in modules.



CR80 Disc Controller Module and associated Adapter module for interfacing up t 4 disc drives in the 5-300 Megabyte range to the CR80 Computer family. Th controller on-board contains its own bit-slice processor and 32 or 64 Kbyte each buffer memory.

9. THE STANDARD CR80 INSTRUCTION SET

This chapter describes the CR80 instruction set.

The standard CR80 instructions are divided into the following groups:

- Modify instructions
- Move instructions
- Jump instructions
- Arithmetic and logic instructions
- Skip instructions
- Shift instructions
- Single bit instructions
- Input/output instructions
- Privileged Read/Write instructions
- Special instructions
- Floating Point instructions
- Alternative CR80 instructions:
  - Move instructions
  - Decimal instructions

The CR80 instructions are classified into two classes:

- NON-PRIVILEGED instructions which only affect the private data of the executing process

- PRIVILEGED instructions which may affect the entire system. These instructions are prohibited while a process is executing in user state.

All CR80 instructions are one 16-bit word. The addressing capability may, however, be extended by one or more preceding MODify instructions.

The CR80 instructions are generally two-address instructions employing, where applicable, the following addressing mode:

- REGISTER
  contents of register is operand

- INDEXED
  memory addressed by register is operand

- IMMEDIATE
  instruction field is operand

- RELATIVE
  memory addressed by instruction field is operand

In the following each instruction is described. An example of an instruction description is shown overleaf.

1

```
****************    RESERVE SINGLE BIT    ********* 1 2 3 4 5 *****

INSTRUCTION:                                                    2
  RESERVE SINGLE BIT                                            3

FORMAT:
                        OP1      OP2                    NORM     MOD      BUS
  RESS 4                  X3  5  C4                    4+3*R    4+3*R  6 3+2*R
                                                       (EXECUTE NEXT:6    R=0)
                                                       (RESERVE AND SKIP: R=1)

FUNCTION:
  SET SEMAPHORE BLOCKING;
  IF OP1.[OP2:1]=1 THEN                                 7
    BEGIN
      OP1:= OP1&1[OP2:0:1];
      LOC:= LOC+2
    END
  ELSE LOC:= LOC+1;
  CLEAR SEMAPHORE BLOCKING;

DESCRIPTION:
  TESTS A BIT IN OP1. IF ONE, 8NO ACTION IS TAKEN. IF ZERO, THE
  BIT IS SET, AND THE NEXT INSTRUCTION IS SKIPPED. THE BIT NUMBER
  IS DEFINED BY OP2. THE INSTRUCTION IS PERFORMED UNDER SEMAPHORE
  PROTECTION.

NOTICE:
  MODIFICATION HAS NO INFLUENCE ON THE INSTRUCTION. MODIFICATION
  IS NOT CLEARED.

CODE:
                        OP1      OP2      BINARY                    HEXA
  RESS 9                  X3     C4                                 11
```

0	X3	C4	0	10	1	0 0 0 1
15						0

The numbered fields 1-11 are described in the following subsections 9.1-9.11.

9.1 SUBGROUP HEADING

This heading specifies the type of instruction being described in the subgroup. If only one instruction is included, the heading is identical to the instruction name.

The heading also defines the type of CPU for which this instruction is implemented:

1 CR8001S with or without micro program RAM but without subbus

2 CR8001S with or without micro program RAM but with subbus

3 CR80101S

4 CR8003M/030P-/00 unmapped CACHE CPU or the CR8002M SCM-CPU

5 CR8003M/040PC/00 mapped CACHE CPU

9.2 INSTRUCTION NAME

The name is a short form text for the instruction function. This column includes all the different instructions being described under this heading.

9.3 MNEMONIC

The instructions are identified by a 3 letter abbreviation for the instruction function, e.g.

MOD for modify
SEQ for skip if equal

In order to ease the understanding and the commenting, some of the

instructions have two names which are synonomous. For example, for the skip instructions it is possible to use one of the two terms:

skip if condition is fulfilled
execute if condition is not fulfilled,

which are identical, for example,

SEQ for skip if equal
INE for execute if not equal

Some mnemonics have a fourth character. The meaning of this character is:

B	byte
C	constant
D	double, i.e. two individual operands
I	indirect
L	long, i.e. two consecutive operands
M	multiple number of words
N	negative
P	pair, i.e. two consecutive operands
U	update with carry or borrow
4	left shifted 4
8	left shifted 8

If the character is surrounded by square brackets, it is an optional feature which may be omitted.

9.4 FORMAT

This column shows the instruction in the different formats which are provided. If the mnemonic is replaced by

<instr>

it means that all the possibilities from the mnemonic fields may be inserted instead.

A 'p' following the mnemonic instruction code means that the instruction is privileged and may only be executed in system state.

A 'pp' following the mnemonic instruction code means that the instruction is privileged and may only be executed in system state at system level 15.

9.5 OPERANDS

An instruction may have either none, one, or two operands. The OP1 and OP2 columns define operand 1 which normally is the source operand, and operand 2, which normally is the destination operand. The columns are empty if no operands are used.

An operand is designated by one or two letters and a number. The first letter indicates the addressing mode:

C	constant
B	base relative
P	program relative
L	location relative
R	register
X	index register

The optional second letter is either

B	for byte address or
N	for negative (two's complement)

The number (n) defines the bit width of the field in the binary instruction code. For example C8 means a constant contained in 8 bits, that is, a constant from 0 to 255.

Rn: The operand is the contents of a register. This number of bits to indicate which register.

Bn: The operand is the contents of the memory location, the address of which is calculated as follows:

Contents of the n-bit field of the instruction plus contents of the base register.

Pn: The operand is the contents of the memory location, the address of which is calculated as follows:

Contents of the n-bit field of the instruction plus contents of the program register (PROG).

Ln: The operand is the contents of the n-bit field of the instruction plus the contents of the location counter (PRPC).

Cn: The operand is the contents of the n-bit field of the instruction.

Xn: Means that the n-bits are used to indicate one of the registers as index. Thus:
B6.X2 means an operand which is:
The contents of the memory location, the address of which is calculated as follows:

Contents of the 6-bit field of the instruction plus the contents of the indicated (index) register plus the contents of the BASE register of the CPU.

The following list contains all operands as they are used in the instruction description:

C8: an 8-bit constant

CN8: an 8-bit constant that is negated before use (two's complement)

C4:	a 4-bit constant
CN4:	a 4-bit constant that is negated before use (two's complement)
B0:	a 0-bit process base relative constant. The notation is used to indicate that modification will be taken as a process base relative.
B6:	a 6-bit base relative constant
B8:	an 8-bit process base relative constant
BB6:	a 6-bit process base relative constant taken as a byte address
P6:	a 6-bit program base relative constant
P8:	an 8-bit program base relative constant
P16:	a 16-bit program base relative constant
L4:	a 4-bit location relative constant
LN4:	a 4-bit location relative constant that is negated (two's complement) before use
L8:	an 8-bit location relative constant that is negated (two's complement) before use
L10:	a 10-bit location relative constant
LN10:	a 10-bit location relative constant that is negated (two's complement) before use
R2:	one of the registers: R0, R1, R2, or R3 used as an operand
R3:	one of the registers: R0, R1, R2, R3, R4, R5, R6, or R7 used as an operand
X2:	one of the registers: X4, X5, X6, or X7 used as index
X3:	one of the registers: X0, X1, X2, X3, X4, X5, X6, or X7 used as index
S2:	one of the registers: S4, S5, or S6 used as return link

Note that the left column above includes the number of bits in the instruction fields used to identity the register, while the description to the right includes the register number as it is used in the assembler coding. Also note that the registers Ri, Xi, and Si are the same physical registers but the use is different.

An M in front of any oppart denotes that the instruction may be modified and that this oppart will be affected by the modifier.

E.g., M P6.X2 will set up the absolute address:

Memory address:=
- contents of program base register
- \+ a 6-bit displacement field from the instruction code
- \+ contents of the index register
- \+ contents of the modify register

The expansion of the displacement field is seen from the following table:

without modify	with 1 modify	with 2 modifies
DP0	DP8, DPN8	DP16
DP4, DPN4	DP12, DPN12	DP16
DP6	DP12, DPN12	DP16
DP8, DPN8	DP16	-

where DP stands for C, B, P, or L.

A modify indication in parentheses specifies that a modify will have a special meaning with the instruction.

9.6 TIMING SPECIFICATIONS

These three columns specify the timing consumption of each CPU for the type 1 and 2 CPUs.

All the numbers are in cycles.
NORM is the total number of cycles for the instruction.
MOD is the total number of cycles for the instruction if it is modified.
BUS is the number of cycles which are cycles.

The CPU cycles are 250ns for CR80101S, CR8003M and CR8002M, and 375 ns for the older CR8001S. The memory cycles vary with the bus and memory type. Please consult the data sheets for the memory and bus types in question.

Example:

Using a standard CR80 P-Bus and a CR8001S CPU, the CPU cycle time is 375 ns and the bus/memory cycle time is 500 ns.

The execution time for move instruction is:

	NORM	MOD	BUS
MOV M B6.X2 R3	4	5	2

The execution time for an unmodified instruction is
(4-2)*375 ns + 2*500 ns = 1750 ns

The execution time for a modified instruction is
(5-2)*375 ns + 2*500 ns = 2225 ns

9.7 FUNCTION

This field shows the function of the instruction in an Algol-like manner. Below is given a list of terms:

Assignment is denoted by :=

The successor of an operand is:

.register: the register with number 1 higher (modulo 8)

.memory: the location with address 1 higher (modulo 64*1024)

The notation: OP1. x:y means a y-bit field of OP1 with the leftmost bit being bit x.

The notation: OP1 & OP2 x:y:z means that a z-bit field from OP2 with the leftmost bit being the bit y is inserted into OP1 with the leftmost bit inserted as bit x. The 16-z remaining bits of OP1 are left unchanged.

The notation: OP1 con OP2 means OP1 concatenated with OP2, i.e. a 32-bit operand where OP1 is the 16 most significant bits (31-16) and OP2 is the 16 least significant bits (15-0).

The notation MEM x means the contents of memory location x.
P_MEM x means the contents of program memory location x.
P_MEM x means the contents of data memory location x.

Process Parameter Block Declaration:

The following Pascal-like declaration of the process parameter block is used in the following:

```
type TRM_rec =
     record
          D_TTR, P_TTR: 0..63;
     end;
var
     CSP, CST, CSB: integer;
     translation_table entry for interrupt process_parameter_page: integer
```

```
BOUNDARY: array 1..15 of integer;:
TRM: array 0..15 of TRM_rec;
```

9.8 DESCRIPTION

This field describes the function of the instruction in plain English. In addition, other special features about a particular instruction are explained.

The description field may be followed by a NOTICE and by a REMARK, which is fully instruction dependent.

9.9 INSTRUCTION CODES

In this column all possible formats of a single instruction are shown.

9.10 BINARY CODE

In this column the instruction bit lay-out is specified and the placement of the variable parameter fields is shown.

The code may contain don't-care bits (Ø) which indicate that both a zero and a one is valid.

9.11 HEXADECIMAL CODE

This column defines the fixed part of the instruction code. The variable fields specified in the BINARY code have to be added to the hexadecimal value before use.

9.12 INSTRUCTIONS, DETAILED DESCRIPTION

In the following pages the CR80 standard instructions are described in detail, with the instruction groups ordered as shown below:

MODIFY INSTRUCTIONS

THE MODIFY INSTRUCTIONS ALL HAVE ONE OPERAND, WHICH IS ADDED TO THE MODIFY REGISTER (MOD). FURTHER THE BOOLEAN, MODIFIED, IS SET TO TRUE.

A MODIFY INSTRUCTION CANNOT ITSELF BE MODIFIED. IF TWO OR MORE MODIFY INSTRUCTIONS FOLLOW EACH OTHER, THEIR FUNCTIONS ARE ACCUMULATED IN THE MODIFY REGISTER (MOD). HOWEVER WHEN TWO MODC INSTRUCTIONS FOLLOW EACH OTHER, THE RESULT IS THE SAME AS IF THE FIRST ONE MODIFIED THE SECOND ONE. THEREFORE THE ASSEMBLER ACTS AS IF THE MODC INSTRUCTION CAN BE MODIFIED. NO INTERRUPT , EXCEPT TIME-OUT AND PARITY, IS SERVED BY THE CPU AS LONG AS THE BOOLEAN MODIFIED IS TRUE.

********** MODIFY WITH CONSTANT ******* 1 2 3 4 5 *****

INSTRUCTION:
MODIFY WITH CONSTANT MODC

FORMAT:

	OP1	OP2	NORM	MOD	BUS
MODC	M C8		5	4	1
MODC	M CN8		5	4	1

FUNCTION:
MOD:= MOD+OP1;
MODIFIED:= TRUE;

DESCRIPTION:
ADDS THE OPERAND TO THE MODIFY REGISTER (MOD) AND SETS THE BOOLEAN, MODIFIED, TO TRUE.

NOTICE:
IF THE INSTRUCTION IS PRECEDED BY ANOTHER MODIFY INSTRUCTION, THIS HAS EFFECT ON THE MODIFY REGISTER (MOD), BUT NOT ON THE PRESENT INSTRUCTION. HOWEVER THIS MAKES NO DIFFERENCE BECAUSE THE INSTRUCTION OPERAND IS A CONSTANT. THEREFORE THE ASSEMBLER WILL INSERT A MODIFY INSTRUCTION IF NECESSARY FOR THE ACTUAL OPERAND VALUE. THE FINAL RESULT OF THIS IS THE SPLITTING OF ONE MODC INSTRUCTION INTO TWO, WHICH TOGETHER COVERS THE FULL ADDRESSING ROOM.

CODE:

	OP1	OP2	BINARY	HEXA
MODC	C8		1 1 1 0 0 1 0 0	00E4
MODC	CN8		1 0 1 0 0 1 0 0	00A4

********** MODIFY WITH SHIFTED CONSTANT 1 2 3 4 5 ******

INSTRUCTION:

MODIFY WITH CONSTANT, SHIFTED 4		MOD4
MODIFY WITH CONSTANT, SHIFTED 8		MOD8

FORMAT:

	OP1	OP2	NORM	MOD	BUS
MOD4	C8		5	4	1
MOD4	CN8		5	4	1
MOD8	C8		5	4	1

FUNCTION:

```
MOD:= MOD+OP1*2**SHIFTS;
MODIFIED:= TRUE;
```

DESCRIPTION:

ADDS THE LEFT SHIFTED OPERAND TO THE MODIFY REGISTER (MOD) AND SETS THE BOOLEAN, MODIFIED, TO TRUE.

CODE:

	OP1	OP2	BINARY			HEXA
MOD4	C8		C8	1 1 1 0	0 1 0 1	00E5
MOD4	CN8		CN8	1 0 1 0	0 1 0 1	00A5
MOD8	C8		C8	0 1 0 1	0 1 1 0	0056

********** MODIFY DIRECT ********** 1 2 3 4 5 *****

INSTRUCTION:
MODIFY [,NEGATIVE] MOD[N]

FORMAT:

	OP1	OP2	NORM	MOD	BUS
MOD[N]	B6.X2		4	6	2
MOD[N]	B8		4	4	2
MOD[N]	P8		4	4	2
MOD[N]	R3		6	7	1
MOD[N]	X3		6	7	2

FUNCTION:
```
OP:= OP1;
IF NEGATIVE THEN OP:= -OP;
MOD:= MOD+OP;
MODIFIED:= TRUE;
```

DESCRIPTION:
IF NEGATIVE MARKED, THE OPERAND IS NEGATED BEFORE USE. ADDS THE OPERAND TO THE MODIFY REGISTER (MOD) AND SETS THE BOOLEAN, MODIFIED, TO TRUE.

NOTICE:
IF THE INSTRUCTION IS PRECEDED BY ANOTHER MODIFY INSTRUCTION, THIS HAS EFFECT ON THE MODIFY REGISTER (MOD), BUT NOT ON THE PRESENT INSTRUCTION.

CODE:

	OP1	OP2	BINARY	HEXA
MOD	B6.X2		X2 B6 0 1 0 1 0 1 0 0	0054
MODN	B6.X2		X2 B6 0 1 0 1 0 1 0 1	0055
MOD	B8		B8 1 0 0 1 0 1 0 0	0094
MODN	B8		B8 0 0 0 1 0 1 0 0	0014
MOD	P8		P8 1 0 0 1 0 1 0 1	0095
MODN	P8		P8 0 0 0 1 0 1 0 1	0015
MOD	R3		0 1 1 1 0 R3 1 0 1 1 1 1 0 1	70BD
MODN	R3		0 0 1 1 0 R3 1 0 1 1 1 1 0 1	30BD
MOD	X3		0 1 1 1 1 X3 1 0 1 1 1 1 0 1	78BD
MODN	X3		0 0 1 1 1 X3 1 0 1 1 1 1 0 1	38BD

****
********** MOVE INSTRUCTIONS **********
****

THE MOVE INSTRUCTIONS PROVIDE THE USER WITH THE POSIBILITY OF MOVING OPERANDS SIZED FROM ONE BIT, BYTE, WORD, DOUBLE WORD, AND UP TO 16 WORDS IN ONE STEP. THE MOVE INSTRUCTION HAS TWO OPERANDS SPECIFIING THE MOVE DIRECTION:

- IMMEDIATE CONSTANT TO REGISTER
- IMMEDIATE CONSTANT TO MEMORY
- FROM REGISTER TO REGISTER
- FROM REGISTER TO MEMORY
- FROM MEMORY TO REGISTER
- FROM MEMORY TO MEMORY

********** MOVE CONSTANT ********** 1 2 3 4 5 *****

INSTRUCTION:
MOVE CONSTANT MOVC

FORMAT:

		OP1	OP2	NORM	MOD	BUS
MOVC	M	C8	R3	3	6	1
MOVC	M	CN8	R3	3	6	1
MOVC	M	C4	X3	7	8	3
MOVC	M	CN4	X3	7	8	2

FUNCTION:
OP2:= OP1;
CLEAR MODIFICATION;

DESCRIPTION:
MOVES A CONSTANT INTO OP2.

CODE:

	OP1	OP2	BINARY	HEXA
MOVC	C8	R3	C8 | 0 1 0 0 1 | R3	0048
MOVC	CN8	R3	CN8 | 0 0 1 0 1 | R3	0028
MOVC	C4	X3	0 | X3 | C4 | 0 1 0 1 | 0 0 0 0	0050
MOVC	CN4	X3	0 | X3 | CN4 | 0 0 0 1 | 0 0 0 0	0010

********** MOVE ********** 1 2 3 4 5 *****

INSTRUCTION:

MOVE MOV

FORMAT:

	OP1	OP2	NORM	MOD	BUS
MOV	M B8	R3	4	5	2
MOV	R3	M B8	3	4	2
MOV	R3	R3	3	4	1
MOV	R3	M B0.X3	3	4	2
MOV	X3	R3	3	4	2
MOV	X3	M B0.X3	3	4	3
MOV	M B6.X2	R3	4	5	2
MOV	R3	M B6.X2	4	5	2
MOV	M P8	R3	4	7	2

FUNCTION:

OP2:= OP1;
CLEAR MODIFICATION;

DESCRIPTION:

OP1 IS MOVED INTO OP2.

CODE:

	OP1	OP2	BINARY	HEXA
MOV	B8	R3	B8 | 0 1 1 0 1 | R3	0068
MOV	R3	B8	B8 | 0 1 1 1 0 | R3	0070
MOV	R3	R3	0 R3 | Φ R3 | 1 0 1 1 | 1 0 1 1	08BB
MOV	R3	B0.X3	1 R3 | Φ X3 | 1 0 1 1 | 1 0 1 1	88BB
MOV	X3	R3	0 X3 | Φ R3 | 1 1 1 1 | 1 0 1 1	08FB
MOV	X3	B0.X3	1 X3 | Φ X3 | 1 1 1 1 | 1 0 1 1	88FB
MOV	B6.X2	R3	X2 | B6 | 0 0 0 1 1 | R3	0018
MOV	R3	B6.X2	X2 | B6 | 1 0 0 0 0 | R3	0080
MOV	P8	R3	P8 | 1 0 0 1 1 | R3	0098

```
****************     MOVE BYTE     ******************** 1 2 3 4 5 *****

INSTRUCTION:
  MOVE BYTE                                                    MOVB

FORMAT:
                    OP1        OP2                  NORM      MOD       BUS
  MOVB   *)       M BB6.X2         R2               6+L       7+L       2
  MOVB   **)            R2   M BB6.X2               6+L       7+L       2
                                                    (RIGHT BYTE: L=0)
  *)  TYPE: LOAD                                    (LEFT BYTE:  L=1)
  **) TYPE: STORE

FUNCTION:
  CASE TYPE OF
    BEGIN
      LOAD:  OP2:= BYTEMEM[2*BASE+REG[X2]+BB6+2*MOD]&0[15:7:8];
      STORE: BYTEMEM[2*BASE+REG[X2]+BB6+2*MOD]:= OP1.[7:8];
    END;
  CLEAR MODIFICATION;

DESCRIPTION:
  MOVES THE FIRST OPERAND INTO THE SECOND OPERAND. THE OPERANDS
  ARE BYTE VALUES. WHEN A BYTE IS MOVED TO A REGISTER, ITS VALUE
  IS PLACED IN THE LOWER BYTE, AND THE UPPER BYTE IS CLEARED.
  WHEN A BYTE IS MOVED FROM A REGISTER, THE LOWER BYTE IS USED,
  AND THE REGISTER IS UNCHANGED. IN MEMORY THE BYTE ADDRESSES
  ARE TAKEN RELATIVE TO THE BASE. THE CONTENT OF THE MODIFY
  REGISTER IS ADDED TO THE BASE REGISTER BEFORE THE ADDRESS IS
  CONVERTED TO A BYTEADDRESS. LOWER BYTES IN MEMORY HAVE
  EVEN BYTE ADDRESSES, UPPER BYTES HAVE ODD ADDRESSES ONE
  HIGHER THAN THE LOWER BYTE IN THE SAME MEMORY WORD.

CODE:
                    OP1        OP2      BINARY                        HEXA
  MOVB              BB6.X2           R2 | X2 | B6 | 0 1 1 1 | 1 0 | R2 | 0078
  MOVB                   R2    BB6.X2 | X2 | B6 | 1 0 0 1 | 0 0 | R2 | 0090
```

INSTRUCTION:
MOVE LONG MOVL

FORMAT:

	OP1	OP2	NORM	MOD	BUS
MOVL	B6.X2	R2	6	7	3
MOVL	R2	M B6.X2	6	7	3

FUNCTION:
OP2:= OP1;
SUCCESSOR(OP2):= SUCCESSOR(OP1);
CLEAR MODIFICATION;

DESCRIPTION:
MOVES OP1 AND ITS SUCCESSOR INTO OP2 AND ITS SUCCESSOR.

CODE:

	OP1	OP2	BINARY	HEXA
MOVL	B6.X2	R2	X2 | B6 | 0 1 1 0 0 0 | R2	0060
MOVL	R2	B6.X2	X2 | B6 | 1 1 0 1 0 0 | R2	0000

********** MOVE PARAMETER ********** 1 2 3 4 5 *****

INSTRUCTION:
MOVE PARAMETER MVP

FORMAT:

		OP1	OP2	NORM	MOD	BUS
MVP	M	P6.X2	R2	4	6	2

FUNCTION:
OP2:= OP1;
CLEAR MODIFICATION;

DESCRIPTION:
MOVES OP1 INTO OP2.

CODE:

	OP1	OP2	BINARY	HEXA
MVP	P6.X2	R2	X2 | P6 | 0 0 1 0 | 0 0 | R2	0020

********** MOVE MULTIPLE ********** 1 2 3 4 5 *****

INSTRUCTION:
MOVE MULTIPLE MOVM

FORMAT:

	OP1	OP2	NORM	MOD	BUS
MOVM (M)	X3	X3	9+2*N	10+2*N	2+2*N
			(N=WORDS MOVED)		
			(UNM.: 9+2*16=41)		

FUNCTION:

```
N:= 1+(MOD-1) MODULO 16;
FOR I:= 1 STEP 1 UNTIL N DO
  BEGIN
    OP2:= OP1;
    INCR(INDEX1);
    INCR(INDEX2)
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:
MOVES A SEQUENCE OF 1 TO 16 WORDS IN MEMORY. THE TWO POINTERS, SPECIFIED AS INDEX REGISTERS, ARE INCREMENTED CORRESPONDINGLY, THUS AT THE END OF THE INSTRUCTION POINTING AT THE WORDS FOLLOWING THE LAST WORDS MOVED. THE NUMBER OF WORDS MOVED IS DEFINED BY THE MODIFY REGISTER (MOD) MODULO 16. HOWEVER A ZERO VALUE SPECIFIES MOVING OF 16 WORDS. THEREBY THE INSTRUCTION WHEN UNMODIFIED MOVES 16 WORDS.

NOTICE:
THE WORDS ARE MOVED ONE AT A TIME, SO OVERLAPPING MAY CAUSE REPETITION OF ONE OR MORE WORDS INSTEAD OF MOVING ALL WORDS.

CODE:

	OP1	OP2	BINARY	HEXA
MOVM	X3	X3	0 X3 0 X3 1 1 1 0 1 0 0 1	08E9

OP1 OP2

********** PUT MASKED ********** 1 2 3 4 5 *****

INSTRUCTION:

PUT MASKED PUT

FORMAT:

	OP1	OP2	NORM	MOD	BUS
PUT (M)	R3	X3	11	12	3
PUT (M)	X3	X3	11	12	4

FUNCTION:

OP2:= (OP2 AND (NOT MOD)) OR (OP1 AND MOD);
CLEAR MODIFICATION;

DESCRIPTION:

THE INSTRUCTION MOVES SOME BITS IN OP1 INTO THE CORRESPONDING BITS IN OP2. THE ACTIVE BITS ARE SELECTED BY ONES IN THE MODIFY REGISTER (MOD). THE REMAINING BITS IN OP2 ARE UNCHANGED.

CODE:

	OP1	OP2	BINARY	HEXA
PUT	R3	X3	1 R3 0 X3 1110 1001	83E9
PUT	X3	X3	1 X3 1 X3 1110 1000	83E8

OP1 OP2

********** EXCHANGE OPERANDS ********** 1 2 3 4 5 *****

INSTRUCTION:
EXCHANGE XCH

FORMAT:

	OP1	OP2	NORM	MOD	BUS
XCH	R3	R3	5	6	1
XCH	R3	M B0.X3	6	7	3
XCH	X3	R3	5	6	3
XCH	X3	M B0.X3	6	7	6

FUNCTION:
WORK:= OP2;
OP2:= OP1;
OP1:= WORK;
CLEAR MODIFICATION;

DESCRIPTION:
EXCHANGES THE TWO OPERANDS.

CODE:

	OP1	OP2	BINARY	HEXA
XCH	R3	R3	0 R3 0 R3 1110 1110	0EE
XCH	R3	B0.X3	0 R3 1 X3 1110 1110	8EE
XCH	X3	R3	1 X3 0 R3 1110 1111	8EF
XCH	X3	B0.X3	1 X3 1 X3 1110 1111	8EF

OP1 OP2

```
**********************     STACK INSTRUCTIONS     ********* 1 2 3 4 5 *****

INSTRUCTION:
  STACK                                                        STC

FORMAT:
                    OP1          OP2                    NORM      MOD       BUS
  STC   *)                       C4        (M)          10+N      11+N      2+N
  STC   **)              R3   M  B0.X3                  5         6         2
  STC   **)              X3   M  B0.X3                  5         6         3
                                                        (N=REGISTERS STACKED)
  *)  STACK REGISTERS
  **) STACK VALUE

FUNCTION:
  CASE INSTR OF
    BEGIN
    STACK REGISTERS:
      FOR I:= (15-C4) STEP -1 UNTIL 0 DO
        BEGIN
          REG[7]:= REG[7]-1;
          MEM[BASE+REG[7]+MOD]:= REG[I]
        END;
    STACK VALUE:
      BEGIN
        REG[X3(OP2)]:= REG[X3(OP2)]+1;
        MEM[REG[X3(OP2)]+BASE+MOD]:= OP1
      END;
    END;
  CLEAR MODIFICATION;

DESCRIPTION:
  STACKS ONE OR MORE REGISTERS IN A STACK. THE STACK POINTER ALWAYS
  POINTS AT THE STACK TOP, CONTAINING THE LAST STACKED VALUE. A
  POSSIBLE MODIFICATION WORKS AS A DISPLACEMENT TO THE STACK
  POINTER. THE FUNCTIONS ARE:
    STACK REGISTERS: STACKS THE REGISTERS WITH THE NUMBERS 15-C4
                     THROUGH 0 INTO A BACKWARD STACK, I.E. THE STACK
                     POINTER IS DECREMENTED BY THE STACK OPERATION.
                     R7 IS USED AS STACK POINTER.
    STACK VALUE:     STACKS OP1 INTO A FORWARD STACK, I.E. THE
                     STACK POINTER IS INCREMENTED BY THE STACK
                     OPERATION.

CODE:
                    OP1          OP2       BINARY                                  HEXA
  STC                            C4        1 1 1 0  C4       1 0 1 1  1 1 1 0      EOBE
  STC                   R3       B0.X3     0  R3  0  X3      1 0 1 1  1 0 0 1      0089
  STC                   X3       B0.X3     0  X3  0  X3      1 1 1 1  1 0 0 1      00F9
                                           OP1   OP2
```

```
**********************     UNSTACK INSTRUCTIONS     ******* 1 2 3 4 5 *****

INSTRUCTION:
  UNSTACK                                                    UNS

FORMAT:
                    OP1        OP2                    NORM      MOD       BUS
  UNS   *)                     C4        (M)          9+N       10+N      2+N
  UNS   **)            R3    M BO.X3                  6         7         2
  UNS   **)            X3    M BO.X3                  6         7         4
                                                      (N=REGISTERS UNSTACKED)
  *)  UNSTACK REGISTERS
  **) UNSTACK VALUE

FUNCTION:
  CASE INSTR OF
    BEGIN
    UNSTACK REGISTERS:
    BEGIN
      CNT:= 15-C4;
      PTR:= REG[7];
      REG[7]:= REG[7]+CNT;
      FOR I:=0 STEP 1 UNTIL CNT DO
        REG[I]:= MEM[BASE+PTR+MOD+I];
    END;
    UNSTACK VALUE:
      BEGIN
        OP1:= MEM[REG[X3(OP2)]+BASE+MOD];
        REG[X3(OP2)]:= REG[X3(OP2)]-1
      END;
    END;
  CLEAR MODIFICATION;

DESCRIPTION:
  UNSTACKS ONE OR MORE REGISTERS FROM A STACK. THE STACK POINTER
  ALWAYS POINTS AT THE STACK TOP, CONTAINING THE NEXT VALUE TO
  BE UNSTACKED. A POSSIBLE MODIFICATION WORKS AS A DISPLACEMENT
  TO THE STACK POINTER. THE FUNCTIONS ARE:
    UNSTACK REGISTERS: UNSTACKS THE REGISTERS WITH THE NUMBERS
                       0 THRU 15-C4 FROM A BACKWARD STACK, I.E. THE
                       STACK POINTER IS INCREMENTED BY THE UNSTACK
                       OPERATION. R7 IS USED AS STACK POINTER.
    UNSTACK VALUE:     UNSTACKS A VALUE INTO OP1 FROM A FORWARD
                       STACK, I.E. THE STACK POINTER IS DECREMENTED
                       BY THE UNSTACK OPERATION.

CODE:
                    OP1        OP2        BINARY                          HEXA
  UNS                          C4         1010 C4   1011 1110           A0BE
  UNS                  R3      BO.X3      0 R3 1 X3 1011 1001           0B89
  UNS                  X3      BO.X3      0 X3 1 X3 1111 1001           0BF9
                                          OP1  OP2
```

****
********** JUMP INSTRUCTIONS **********
****

JUMP INSTRUCTIONS ARE MAINLY UNCONDITIONAL. CONDITIONAL JUMPS
SHOULD BE PERFORMED BY COMBINATIONS OF SKIP INSTRUCTIONS AND
UNCONDITIONAL JUMPS. HOWEVER A FEW CONDITIONAL JUMP INSTRUCTIONS
ARE INCLUDED IN ORDER TO SPEED UP THE MOST COMMON CASES.

INSTRUCTION:
JUMP RELATIVE JMP

FORMAT:

	OP1	OP2	NORM	MOD	BUS
JMP		M L10	3	5	2
JMP		M LN10	3	5	2
JMP		M L8	3	6	2
JMP		M LN8	3	6	2
JMP	S2	M L8	4	7	2
JMP	S2	M LN8	4	7	2

FUNCTION:
IF RETURN THEN S2:= LOC+1-PROG;
LOC:= LOC+1+OP2+MOD;
CLEAR MODIFICATION;

DESCRIPTION:
JUMPS UNCONDITIONALLY TO THE LOCATION SPECIFIED BY OP2. IF A RETURN REGISTER IS SPECIFIED, THIS WILL POINT AT THE NEXT INSTRUCTION.

NOTICE:
THE MODIFY REGISTER IS ALWAYS ADDED TO LOC, ALSO IF THE OPERAND IS NEGATED BEFORE USE.

CODE:

	OP1	OP2	BINARY			HEXA
JMP		L10	L10	1 1 0 1	1 0	00D8
JMP		LN10	LN10	0 1 0 1	1 0	0058
JMP		L8	L8	1 1 0 1	1 1 0 0	00DC
JMP		LN8	LN8	0 1 0 1	1 1 0 0	005C
JMP	S2	L8	L8	1 1 0 1	1 1	S2	00DC
JMP	S2	LN8	LN8	0 1 0 1	1 1	S2	005C

********** JUMP INDEXED ********** 1 2 3 4 5 *****

INSTRUCTION:
JUMP INDEXED JMP

FORMAT:

	OP1	OP2	NORM	MOD	BUS
JMP	M	P6.X2	3	5	2

FUNCTION:
LOC:= REG[X2]+P6+MOD+PROG;
CLEAR MODIFICATION;

DESCRIPTION:
JUMPS UNCONDITIONALLY TO THE LOCATION SPECIFIED BY OP2.

REMARK:
THE INSTRUCTION IS MAINLY INTENDED FOR RETURN FROM A SUBROUTINE.

CODE:

	OP1	OP2	BINARY	HEXA
JMP		P6.X2	X2 | P6 | 1 0 1 1 | 1 1 1 1	00BF

INSTRUCTION:

JUMP INDIRECT JMPI

FORMAT:

	OP1	OP2	NORM	MOD	BUS
JMPI		M B6.X2	4	5	3
JMPI		M B8	5	6	3
JMPI		M P8	5	8	3
JMPI	S2	M P8	6	9	3

FUNCTION:

IF RETURN THEN S2:= LOC+1-PROG;
LOC:= OP2+PROG;
CLEAR MODIFICATION;

DESCRIPTION:

JUMPS UNCONDITIONALLY TO THE LOCATION SPECIFIED BY OP2. IF A RETURN REGISTER IS SPECIFIED, THIS WILL POINT AT THE NEXT INSTRUCTION.

CODE:

	OP1	OP2	BINARY	HEXA
JMPI		B6.X2	X2 B6 | 0 0 0 1 | 0 0 1 0	0012
JMPI		B8	B8 | 1 1 1 0 | 0 1 1 1	00E7
JMPI		P8	P8 | 1 1 1 1 | 1 1 0 0	00FC
JMPI	S2	P8	P8 | 1 1 1 1 | 1 1 | S 2	00FC

INSTRUCTION:

JUMP IF OPERAND NONZERO	JON
JUMP IF OPERAND ZERO	JOZ

FORMAT:

	OP1	OP2	NORM	MOD	BUS
<INSTR>	R3	L4	3	3	1
<INSTR>	R3	LN4	3	3	1
<INSTR>	X3	L4	4	4	3
<INSTR>	X3	LN4	4	4	3
			(NO JUMP: J=0)		
			(JUMP: J=1)		

```
FUNCTION:
  IF CASE INSTR OF
      (JON: OP1<>0,
       JOZ: OP1= 0
      ) THEN
    LOC:= LOC+1+OP2
  ELSE
    LOC:= LOC+1;
```

DESCRIPTION:
IF THE INSTRUCTION CONDITION IS FULFILLED, A JUMP IS PERFORMED TO THE LOCATION SPECIFIED BY OP2. THE CONDITIONS ARE:
JON: THE OPERAND IS NONZERO.
JOZ: THE OPERAND IS ZERO.

NOTICE:
MODIFICATION HAS NO INFLUENCE ON THE INSTRUCTIONS.
MODIFICATION IS NOT CLEARED BY THE INSTRUCTIONS.

CODE:

	OP1	OP2	BINARY	HEXA
JON	R3	L4	1 R3 L4 1111 0000	80F0
JON	R3	LN4	1 R3 LN4 1011 0000	80B0
JON	X3	L4	1 X3 L4 1111 0001	80F1
JON	X3	LN4	1 X3 LN4 1011 0001	80B1
JOZ	R3	L4	1 R3 L4 1111 0010	80F2
JOZ	R3	LN4	1 R3 LN4 1011 0010	80B2
JOZ	X3	L4	1 X3 L4 1111 0011	80F3
JOZ	X3	LN4	1 X3 LN4 1011 0011	80B3

********** JUMP ON OVERFLOW ********** 1 2 3 4 5 *****

INSTRUCTION:
  JUMP IF OVERFLOW JVN

FORMAT:

	OP1	OP2	NORM	MOD	BUS
JVN		M L4	5+2*J	6+2*J	1+J
JVN		M LN4	5+2*J	6+2*J	1+J
			(NO JUMP: J=0)		
			(JUMP: J=1)		

FUNCTION:
```
  IF PSW.[V:1]=0 THEN   LOC:= LOC+1+OP2
  ELSE                  LOC:= LOC+1;
  CLEAR MODIFICATION;
```

DESCRIPTION:
  IF THE OVERFLOW BIT IN PSW IS RESET, A JUMP IS PERFORMED TO THE LOCATION SPECIFIED BY OP2. THAT IS, THE JUMP IS PERFORMED WHEN AN OVERFLOW CONDITION EXISTS.

CODE:

	OP1	OP2	BINARY				HEXA
JVN		L4	1 Φ 1 1	L4	1 0 1 1	1 1 1 0	F0BE
JVN		LN4	0 Φ 1 1	LN4	1 0 1 1	1 1 1 0	70BE

INSTRUCTION:
SUBTRACT ONE, BRANCH IF NONZERO SOB

FORMAT:

	OP1	OP2	NORM	MOD	BUS
SOB	R3	M LN8	2+N	3+3*N	1+N
			(ZERO: N=0)		
			(NONZERO: N=1)		

FUNCTION:
```
OP1:= OP1-1;
IF OP1<>0 THEN   LOC:= LOC+1+OP2
ELSE             LOC:= LOC+1;
CLEAR MODIFICATION;
```

DESCRIPTION:
SUBTRACTS ONE FROM OP1. IF OP1 THEN IS NONZERO, A JUMP IS PERFORMED TO THE LOCATION SPECIFIED BY OP2.

CODE:

	OP1	OP2	BINARY	HEXA
SOB	R3	LN8	LN 8 | 1 1 0 0 | 0 | R3	00C0

****
********** ARITHMETIC AND LOGIC INSTRUCTIONS **********
****

THE ARITHMETIC AND LOGIC INSTRUCTIONS COVER THE FULL RANGE OF ARITHMETIC AND LOGIC OPERATIONS, INCLUDING MULTIPLE WORD OPERATIONS. IN GENERAL THE OPERATIONS MAY BE PERFORMED EITHER ON REGISTERS OR DIRECTLY IN MEMORY USING AN INDEX REGISTER.

********** ADD CONSTANT ********** 1 2 3 4 5 *****

INSTRUCTION:
ADD CONSTANT ADDC

FORMAT:

		OP1	OP2	NORM	MOD	BUS
ADDC	M	C8	R3	4	7	1
ADDC	M	CN8	R3	4	7	1

FUNCTION:
OP1:= OP1+MOD (2'S COMPLEMENT)
OP2:= OP2+OP1;
SET CONDITION CODES;
CLEAR MODIFICATION;

DESCRIPTION:
ADDS AN IMMEDIATE CONSTANT TO A REGISTER.

CODE:

	OP1	OP2	BINARY	HEXA
ADDC	C8	R3	C8 | 1 1 0 0 | 1 | R3	00C8
ADDC	CN8	R3	CN8 | 1 0 1 0 | 1 | R3	00A8

```
**********      ADD      ************************* 1 2 3 4 5 *****

INSTRUCTION:
  ADD [,UPDATE WITH CARRY]                                  ADD[U]

FORMAT:
                    OP1          OP2                NORM     MOD      BUS
  <INSTR>            R3           R3                 5        6        1
  <INSTR>            R3        M  B0.X3              5        6        3
  <INSTR>            X3           R3                 5        6        2
  <INSTR>            X3        M  B0.X3              5        6        4
  ADD             M  B6.X2        R3                 4        5        2
  ADD                R3        M  B6.X2              4        5        3

FUNCTION:
  CASE INSTR OF
    BEGIN
      ADD:  OP2:= OP2+OP1;
      ADDU: OP2:= OP2+OP1+PSW.[C:1];
    END;
  CLEAR MODIFICATION;
  SET CONDITION CODES;

DESCRIPTION:
  THE ARITHMETIC OPERATION IS PERFORMED ON THE TWO OPERANDS. THE
  RESULT IS DELIVERED IN THE SECOND OPERAND. THE OPERATIONS
  USE TWO'S COMPLEMENT ARITHMETIC. THEY ARE:
    ADD:  OP1 IS ADDED TO OP2.
    ADDU: OP1 AND THE CARRY ARE ADDED TO OP2.
  INSTRUCTIONS UPDATING WITH CARRY WORK AS REQUIRED FOR MULTI-
  LENGTH CALCULATIONS.
```

CODE:

	OP1	OP2	BINARY	HEXA
ADD	R3	R3	0 R3 0 R3 1000 1010	008A
ADD	R3	B0.X3	0 R3 1 X3 1000 1010	088A
ADD	X3	R3	1 X3 0 R3 1000 1010	908A
ADD	X3	B0.X3	1 X3 1 X3 1000 1010	988A
ADD	B6.X2	R3	X2 B6 0011 0 R3	0030
ADD	R3	B6.X2	X2 B6 0000 1 R3	0008
ADDU	R3	R3	0 R3 0 R3 1101 0100	00D4
ADDU	R3	B0.X3	0 R3 1 X3 1101 0100	08D4
ADDU	X3	R3	1 X3 0 R3 1101 0100	30D4
ADDU	X3	B0.X3	1 X3 1 X3 1101 0100	88D4

```
**********      SUBTRACT      ********** 1 2 3 4 5 *****

INSTRUCTION:
  SUBTRACT [,UPDATE WITH CARRY]                          SUB[U]

FORMAT:
                    OP1        OP2                  NORM     MOD      BUS
  <INSTR>            R3          R3                  5        6        1
  <INSTR>            R3      M 80.X3                 5        6        3
  <INSTR>            X3          R3                  5        6        2
  <INSTR>            X3      M 80.X3                 5        6        4

FUNCTION:
  CASE INSTR OF
    BEGIN
      SUB:  OP2:= OP2-OP1;
      SUBU: OP2:= OP2-OP1+PSW.[C:1]-1;
    END;
  CLEAR MODIFICATION;
  SET CONDITION CODES;

DESCRIPTION:
  THE ARITHMETIC OPERATION IS PERFORMED ON THE TWO OPERANDS. THE
  RESULT IS DELIVERED IN THE SECOND OPERAND. THE OPERATIONS
  USE TWO'S COMPLEMENT ARITHMETIC. THEY ARE:
    SUB:  OP1 IS SUBTRACTED FROM OP2.
    SUBU: OP1 AND THE BORROW ARE SUBTRACTED FROM OP2.
  INSTRUCTIONS UPDATING WITH BORROW WORK AS REQUIRED FOR MULTI-
  LENGTH CALCULATIONS.
```

CODE:

	OP1	OP2	BINARY	HEXA
SUB	R3	R3	0 R3 0 R3 1 0 0 0 1 0 1 1	0038
SUB	R3	80.X3	0 R3 1 X3 1 0 0 0 1 0 1 1	0888
SUB	X3	R3	1 X3 0 R3 1 0 0 0 1 0 1 1	8088
SUB	X3	80.X3	1 X3 1 X3 1 0 0 0 1 0 1 1	8888
SUBU	R3	R3	0 R3 0 R3 1 1 0 1 0 1 0 1	00D5
SUBU	R3	80.X3	0 R3 1 X3 1 1 0 1 0 1 0 1	08D5
SUBU	X3	R3	1 X3 0 R3 1 1 0 1 0 1 0 1	80D5
SUBU	X3	80.X3	1 X3 1 X3 1 1 0 1 0 1 0 1	88D5

********** MULTIPLY ********** 1 2 3 4 5 *****

INSTRUCTION:
MULTIPLY MUL

FORMAT:

	OP1	OP2	NORM	MOD	BUS
MUL	R3	M BO.X3	25	26	4
MUL	X3	M BO.X3	26	27	5

FUNCTION:
DEFINE OP3::= SUCCESSOR(OP2) CON OP2;
OP3:= OP1*OP2;
CLEAR MODIFICATION;

DESCRIPTION:
OP1 AND OP2 ARE MULTIPLIED. THE RESULT IS STORED AS A THIRTYTWO-BIT TWO'S COMPLEMENT NUMBER INTO THE CONCATENATION OF THE SUCCESSOR OF OP2 AND OP2 ITSELF.

CODE:

	OP1	OP2	BINARY	HEXA
MUL	R3	BO.X3	0 R3 1 X3 1 1 1 0 1 0 1 0	03EA
MUL	X3	BO.X3	1 X3 1 X3 1 1 1 0 1 0 1 0	83EA

INSTRUCTION:
MULTIPLY MUL

FORMAT:

	OP1	OP2	NORM	MOD	BUS
MUL	R3	R3	-	-	-
MUL	X3	R3	-	-	-

FUNCTION:
DEFINE OP3::= SUCCESSOR(OP2) CON OP2;
OP3:= OP1*OP2;
CLEAR MODIFICATION;

DESCRIPTION:
OP1 AND OP2 ARE MULTIPLIED. THE RESULT IS STORED AS A THIRTYTWO-BIT TWO'S COMPLEMENT NUMBER INTO THE CONCATENATION OF THE SUCCESSOR OF OP2 AND OP2 ITSELF.

CODE:

	OP1	OP2	BINARY	HEXA
MUL	R3	R3	0 R3 0 R3 1110 1010	00EA
MUL	X3	R3	1 X3 0 R3 1110 1010	80EA

********** DIVIDE ********** 1 2 3 4 5 *****

INSTRUCTION:
DIVIDE DIV

FORMAT:

	OP1	OP2	NORM	MOD	BUS
DIV	R3	R3	30	31	1
DIV	X3	R3	30	31	2

FUNCTION:
DEFINE OP3::= SUCCESSOR(OP2) CON OP2;
OP2:= OP3 DIV OP1;
SUCCESSOR(OP2):= OP3 REM OP1;
CLEAR MODIFICATION;

DESCRIPTION:
THE SUCCESSOR OF OP2 CONCATENATED WITH OP2 ITSELF IS TAKEN AS AN UNSIGNED THIRTYTWO-BIT NUMBER THAT IS DIVIDED BY OP1. THE SIXTEEN LEAST SIGNIFICANT BITS OF THE RESULT ARE STORED INTO OP2, AND THE REMAINDER OF THE DIVISION IS STORED INTO THE SUCCESSOR OF OP2.

NOTE:
NO INDICATION IS GIVEN IF AN OVERFLOW CONDITION OCCURS.

CODE:

	OP1	OP2	BINARY														HEXA
DIV	R3	R3	0	R	3	0	R	3	1	1	1	0	1	0	1	1	00EB
DIV	X3	R3	1	X	3	0	R	3	1	1	1	0	1	0	1	1	80EB

INSTRUCTION:
DIVIDE DIV

FORMAT:

	OP1	OP2	NORM	MOD	BUS
DIV	R3	M B0.X3	-	-	-
DIV	X3	M B0.X3	-	-	-

FUNCTION:
```
DEFINE OP3::= SUCCESSOR(OP2) CON OP2;
OP2:= OP3 DIV OP1;
SUCCESSOR(OP2):= OP3 REM OP1;
CLEAR MODIFICATION;
```

DESCRIPTION:
THE SUCCESSOR OF OP2 CONCATENATED WITH OP2 ITSELF IS TAKEN AS AN UNSIGNED THIRTYTWO-BIT NUMBER THAT IS DIVIDED BY OP1. THE SIXTEEN LEAST SIGNIFICANT BITS OF THE RESULT ARE STORED INTO OP2, AND THE REMAINDER OF THE DIVISION IS STORED INTO THE SUCCESSOR OF OP2.

NOTE:
NO INDICATION IS GIVEN IF AN OVERFLOW CONDITION OCCURS.

CODE:

	OP1	OP2	BINARY	HEXA
DIV	R3	B0.X3	0 R3 1 X3 1 1 1 0 1 0 1 1	08EB
DIV	X3	B0.X3	1 X3 1 X3 1 1 1 0 1 0 1 1	88EB

********** LOGICAL DYADIC INSTRUCTIONS 1 2 3 4 5 *****

INSTRUCTION:

LOGICAL AND	AND
LOGICAL INCLUSIVE OR	IOR
LOGICAL EXCLUSIVE OR	XOR

FORMAT:

	OP1	OP2	NORM	MOD	BUS
<INSTR>	R3	R3	5	6	1
<INSTR> *)	R3	M B0.X3	5	6	3
<INSTR>	X3	R3	5	6	2
<INSTR> *)	X3	M B0.X3	5	6	4

*) ONLY AND, IOR.

FUNCTION:

```
CASE INSTR OF
  BEGIN
    AND: OP2:= OP2 AND OP1;
    IOR: OP2:= OP2 OR OP1;
    XOR: OP2:= (OP2 OR OP1)-(OP2 AND OP1);
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:

THE LOGICAL DYADIC OPERATION IS PERFORMED ON THE TWO OPERANDS. THE RESULT IS DELIVERED IN THE SECOND OPERAND. THE TRUTH-TABLES ARE:

AND	IOR	XOR
0,0: 0	0,0: 0	0,0: 0
0,1: 0	0,1: 1	0,1: 1
1,0: 0	1,0: 1	1,0: 1
1,1: 1	1,1: 1	1,1: 0

CODE:

	OP1	OP2	BINARY														HEXA
AND	R3	R3	0	R3	0	R3	1	0	0	0	1	1	0	0			008C
AND	R3	B0.X3	0	R3	1	X3	1	0	0	0	1	1	0	0			088C
AND	X3	R3	1	X3	0	R3	1	0	0	0	1	1	0	0			808C
AND	X3	B0.X3	1	X3	1	X3	1	0	0	0	1	1	0	0			888C
IOR	R3	R3	0	R3	0	R3	1	0	0	0	1	1	0	1			008D
IOR	R3	B0.X3	0	R3	1	X3	1	0	0	0	1	1	0	1			088D
IOR	X3	R3	1	X3	0	R3	1	0	0	0	1	1	0	1			808D
IOR	X3	B0.X3	1	X3	1	X3	1	0	0	0	1	1	0	1			888D
XOR	R3	R3	0	R3	0	R3	1	1	1	0	1	0	0	0			00E8
XOR	X3	R3	1	X3	0	R3	1	1	1	0	1	0	0	0			80E8

********** MONADIC INSTRUCTIONS ******* 1 2 3 4 5 *****

INSTRUCTION:

CLEAR	CLR
DECREMENT	DEC
INCREMENT	INC
INVERT	INV
NEGATE	NEG
SWAP	SWP

FORMAT:

	OP1	OP2	NORM	MOD	BUS
CLR		M B6.X2	4	5	3
DEC		M B6.X2	4	5	3
INC		M B6.X2	4	5	3
INV		R3	7	8	1
INV		M B0.X3	7	8	3
NEG		R3	7	8	1
NEG		M B0.X3	7	8	3
SWP		R3	7	8	1
SWP		M B0.X3	8	9	3

FUNCTION:

```
CASE INSTR OF
  BEGIN
    CLR: OP2:= 0;
    DEC: OP2:= OP2-1;
    INC: OP2:= OP2+1;
    INV: OP2:= -1-OP2;
    NEG: OP2:= -OP2;
    SWP: OP2:= OP2.[15:8]&OP2[15:7:8];
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:

THE OPERAND IS ASSIGNED A VALUE DEPENDING ON THE INSTRUCTION:

- CLR: ZERO.
- DEC: THE OPERAND DECREMENTED BY ONE.
- INC: THE OPERAND INCREMENTED BY ONE.
- INV: ONE'S COMPLEMENT OF THE OPERAND.
- NEG: TWO'S COMPLEMENT OF THE OPERAND.
- SWP: THE TWO BYTES IN THE OPERAND EXCHANGED.

CODE:

	OP1	OP2	BINARY	HEXA
CLR		B6.X2	X2 B6 0 0 1 0 0 1 1 1	0027
DEC		B6.X2	X2 B6 0 0 0 1 0 0 1 1	0013
INC		B6.X2	X2 B6 0 0 0 1 0 1 1 0	0016
INV		R3	1 0 0 1 0 R3 1 0 1 1 1 1 0 0	90BC
INV		B0.X3	1 0 0 1 1 X3 1 0 1 1 1 1 0 0	98BC
NEG		R3	1 1 1 1 0 R3 1 0 1 1 1 1 0 0	F0BC
NEG		B0.X3	1 1 1 1 1 X3 1 0 1 1 1 1 0 0	F8BC
SWP		R3	0 1 1 0 0 R3 1 0 1 1 1 1 0 0	60BC
SWP		B0.X3	0 1 1 0 1 X3 1 0 1 1 1 1 0 0	68BC

INSTRUCTION:

DECREMENT DOUBLE	DECD
INCREMENT DOUBLE	INCD

FORMAT:

	OP1	OP2	NORM	MOD	BUS
<INSTR>	R3	R3	5	6	1
<INSTR>	X3	M 80.X3	6	7	5

FUNCTION:

```
WORK1:= OP1; WORK2:= OP2;
CASE INSTR OF
  BEGIN
    DECD: BEGIN OP1:= WORK1-1; OP2:= WORK2-1; END;
    INCD: BEGIN OP1:= WORK1+1; OP2:= WORK2+1; END;
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:

DECREMENTS OR INCREMENTS THE TWO OPERANDS OF THE INSTRUCTION.

NOTICE:

IF THE TWO OPERANDS ARE THE SAME, IT IS ONLY UPDATED ONCE.

CODE:

	OP1	OP2	BINARY					HEXA
DECD	R3	R3	0 R3	1 R3	1 0 0 0	1 1 1 0		088E
DECD	X3	BO.X3	1 X3	1 X3	1 0 0 0	1 1 1 1		888F
INCD	R3	R3	0 R3	0 R3	1 0 0 0	1 1 1 0		008E
INCD	X3	BO.X3	1 X3	0 X3	1 0 0 0	1 1 1 1		808F

INSTRUCTION:

DOUBLE DECREMENT, PAIR	DDCP
DOUBLE INCREMENT, PAIR	DICP

FORMAT:

	OP1	OP2	NORM	MOD	BUS
DDCP	R3		9	10	1
DDCP	M BO.X3		10	11	5
DICP	R3		8	9	1
DICP	M BO.X3		9	10	5

FUNCTION:

```
DEFINE OP2::= SUCCESSOR(OP1);
CASE INSTR OF
  BEGIN
    DDCP: BEGIN OP1:= OP1-2; OP2:= OP2-2; END;
    DICP: BEGIN OP1:= OP1+2; OP2:= OP2+2; END;
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:

DECREMENTS OR INCREMENTS THE OPERAND AND ITS SUCCESSOR TWICE.

CODE:

	OP1	OP2	BINARY					HEXA
DDCP	R3		0 0 1 1	0	R3	1 0 1 1	1 1 0 0	30BC
DDCP	BO.X3		0 0 1 1	1	X3	1 0 1 1	1 1 0 0	38BC
DICP	R3		0 0 1 0	0	R3	1 0 1 1	1 1 0 0	20BC
DICP	BO.X3		0 0 1 0	1	X3	1 0 1 1	1 1 0 0	28BC

INSTRUCTION:
SIGN EXTENSION SXT

FORMAT:

	OP1	OP2	NORM	MOD	BUS
SXT	X3	(M)C4	5+N	7+N	3
			(POSITIVE: N=0)		
			(NEGATIVE: N=1)		

FUNCTION:

```
BIT:= IF MODIFIED THEN   MOD MODULO 16
      ELSE                          OP2;
OP1:= OP1.[BIT:BIT+1];
IF OP1.[BIT:1]=1 THEN OP1:= OP1&(-1)[15:15:15-BIT];
CLEAR MODIFICATION;
```

DESCRIPTION:
EXTENDS A BIT IN OP1 AS A SIGN BIT. THE BIT NUMBER IS DEFINED BY OP2 OR, IF MODIFIED, BY THE MODIFY REGISTER (MOD) MODULO 16.

CODE:

	OP1	OP2	BINARY	HEXA
SXT	X3	C4	0 X3 C4 0 1 1 0 0 1 1 1	0067

********** EXTRACT BIT GROUP ********** 1 2 3 4 5 *****

INSTRUCTION:
EXTRACT XTR

FORMAT:

	OP1	OP2	NORM	MOD	BUS
XTR	R3	C4	3	4	1
XTR	X3	C4	3	4	3

FUNCTION:
IF OP2=0 THEN OP1:= 0
ELSE OP1:= OP1.[BITS-1:BITS];
CLEAR MODIFICATION;

DESCRIPTION:
EXTRACTS A GROUP OF LEAST SIGNIFICANT BITS IN OP1. THE NUMBER OF BITS IS DEFINED BY OP2.

REMARK:
IF THE SECOND OPERAND SPECIFIES ZERO BITS, THE FIRST OPERAND IS CLEARED.

CODE:

	OP1	OP2	BINARY				HEXA
XTR	R3	C4	0	R3	C4	1 1 1 0 1 1 0 0	00EC
XTR	X3	C4	1	X3	C4	1 1 1 0 1 1 0 1	80ED

****
********** SKIP INSTRUCTIONS **********
****

THE SKIP INSTRUCTIONS ARE DIVIDED INTO THE GROUPS:
  SKIP ON BIT
  SKIP ON OPERAND
  SKIP ON CONDITION

THE INSTRUCTIONS HAVE SOME COMMON VARIATIONS. GENERALLY THERE ARE ONLY 5 INSTRUCTIONS: SBN, SON, SNE, SGE, SHS. EACH OF THESE 5 INSTRUCTIONS FURTHER EXISTS IN A VERSION SKIPPING ON THE COMPLEMENTARY CONDITION. EACH OF THESE 10 INSTRUCTIONS FURTHER EXISTS IN A VERSION SKIPPING TWO LOCATIONS INSTEAD OF ONE. EACH OF THESE 20 INSTRUCTIONS HAS A SYNONYM. THIS DEFINES EXACTLY THE SAME INSTRUCTION. THE REASON FOR HAVING TWO NAMES FOR THE INSTRUCTIONS IS THE BETTER POSSIBILITY OF CORRESPONDENCE BETWEEN CODE AND COMMENTS, FOR EXAMPLE:

```
  SBNP    X4      0      ;   IF X4[0] OR
  INEP    R0      R1     ;       R0<>R1 AND
  ILT     R2         R3  ;       R2<R3 THEN
  JMP             NEXT   ;     GOTO NEXT;
```

SOME COMBINATIONS OF SKIP AND JUMP INSTRUCTIONS CAN BE REPLACED BY A CONDITIONAL JUMP INSTRUCTION.

```
****************     SKIP ON BIT     **************** 1 2 3 4 5 *****

INSTRUCTION:
  SKIP IF BIT NONZERO [,PAIR]                              SBN[P]
  SKIP IF BIT ZERO [,PAIR]                                 SBZ[P]
SYNONYME:
  EXECUTE IF BIT ZERO [,PAIR]                              IBZ[P] (SBN[P])
  EXECUTE IF BIT NONZERO [,PAIR]                           IBN[P] (SBZ[P])

FORMAT:
                  OP1       OP2                 NORM      MOD       BUS
  <INSTR>           R3 (M)C4                    4+S       6+S       1+S
  <INSTR>           X3 (M)C4                    4+S       6+S       2+S
                                                (NO SKIP: S=0)
                                                (SKIP:    S=1)

FUNCTION:
  IF MODIFIED THEN DEFINE OP2::= MOD MODULO 16;
  IF CASE INSTR OF
      (SBN[P],IBZ[P]: OP1.[OP2:1]=1,
       SBZ[P],IBN[P]: OP1.[OP2:1]=0
      ) THEN
    IF PAIR THEN LOC:= LOC+3     % SKIP 2
    ELSE         LOC:= LOC+2     % SKIP 1
  ELSE           LOC:= LOC+1;    % SKIP 0
  CLEAR MODIFICATION;

DESCRIPTION:
  INSPECTS A BIT IN OP1. THE BIT NUMBER IS SPECIFIED BY OP2
  OR, IF MODIFIED, BY THE MODIFY REGISTER (MOD) MODULO 16.
  IF THE CONDITION OF THE INSTRUCTION IS FULFILLED, A SKIP IS
  PERFORMED. IF PAIR IS FALSE, ONE MEMORY LOCATION IS SKIPPED,
  IF PAIR IS TRUE, TWO MEMORY LOCATIONS ARE SKIPPED. THE
  CONDITIONS ARE:
    SBN[P],IBZ[P]: THE BIT IS NONZERO.
    SBZ[P],IBN[P]: THE BIT IS ZERO.

CODE:
                  OP1       OP2     BINARY                                HEXA
  SBN   (IBZ)       R3        C4    | 0 | R3 | C4 | 1 1 1 1 | 0 0 0 0 |  00F0
  SBN   (IBZ)       X3        C4    | 0 | X3 | C4 | 1 1 1 1 | 0 0 0 1 |  00F1
  SBZ   (IBN)       R3        C4    | 0 | R3 | C4 | 1 1 1 1 | 0 0 1 0 |  00F2
  SBZ   (IBN)       X3        C4    | 0 | X3 | C4 | 1 1 1 1 | 0 0 1 1 |  00F3
  SBNP  (IBZP)      R3        C4    | 0 | R3 | C4 | 1 0 1 1 | 0 0 0 0 |  00B0
  SBNP  (IBZP)      X3        C4    | 0 | X3 | C4 | 1 0 1 1 | 0 0 0 1 |  00B1
  SBZP  (IBNP)      R3        C4    | 0 | R3 | C4 | 1 0 1 1 | 0 0 1 0 |  00B2
  SBZP  (IBNP)      X3        C4    | 0 | X3 | C4 | 1 0 1 1 | 0 0 1 1 |  00B3
```

```
***************     SKIP ON OPERAND       ********** 1 2 3 4 5 *****

INSTRUCTION:
  SKIP IF OPERAND NONZERO [,PAIR]                          SON[P]
  SKIP IF OPERAND ZERO [,PAIR]                             SOZ[P]
SYNONYME:
  EXECUTE IF OPERAND ZERO [,PAIR]                          IOZ[P] (SON[P])
  EXECUTE IF OPERAND NONZERO [,PAIR]                       ION[P] (SOZ[P])

FORMAT:
                  OP1       OP2                  NORM     MOD      BUS
  <INSTR>       M B6.X2                          4+S      5+S      2+S
                                                 (NO SKIP: S=0)
                                                 (SKIP:    S=1)

FUNCTION:
  IF CASE INSTR OF
      (SON[P],IOZ[P]: OP1<>0,
       SOZ[P],ION[P]: OP1=0
      ) THEN
    IF PAIR THEN LOC:= LOC+3     % SKIP 2
    ELSE         LOC:= LOC+2     % SKIP 1
  ELSE           LOC:= LOC+1;    % SKIP 0
  CLEAR MODIFICATION;

DESCRIPTION:
  IF THE CONDITION OF THE INSTRUCTION IS FULFILLED, A SKIP IS
  PERFORMED. IF PAIR IS FALSE, ONE MEMORY LOCATION IS SKIPPED,
  IF PAIR IS TRUE, TWO MEMORY LOCATIONS ARE SKIPPED.
  THE CONDITIONS ARE:
    SON[P],IOZ[P]: THE OPERAND IS NONZERO.
    SOZ[P],ION[P]: THE OPERAND IS ZERO.

CODE:
                  OP1       OP2     BINARY                          HEXA
  SON  (IOZ)      B6.X2             X2 | B6 | 0 0 1 1 | 1 0 1 1    003B
  SOZ  (ION)      B6.X2             X2 | B6 | 0 0 1 1 | 1 0 1 0    003A
  SONP (IOZP)     B6.X2             X2 | B6 | 0 0 1 1 | 1 0 0 1    0039
  SOZP (IONP)     B6.X2             X2 | B6 | 0 0 1 1 | 1 0 0 0    0038
```

```
****************    SKIP ON CONDITION    ********** 1 2 3 4 5 *****

INSTRUCTION:
  SKIP IF EQUAL [,PAIR]                                    SEQ[P]
  SKIP IF NOT EQUAL [,PAIR]                                SNE[P]
  SKIP IF GREATER OR EQUAL [,PAIR]                         SGE[P]
  SKIP IF LESS THAN [,PAIR]                                SLT[P]
  SKIP IF HIGHER OR SAME [,PAIR]                           SHS[P]
  SKIP IF LOWER [,PAIR]                                    SLO[P]
SYNONYME:
  EXECUTE IF NOT EQUAL [,PAIR]                             INE[P] (SEQ[P])
  EXECUTE IF EQUAL [,PAIR]                                 IEQ[P] (SNE[P])
  EXECUTE IF LESS THAN [,PAIR]                             ILT[P] (SGE[P])
  EXECUTE IF GREATER OR EQUAL [,PAIR]                      IGE[P] (SLT[P])
  EXECUTE IF LOWER [,PAIR]                                 ILO[P] (SHS[P])
  EXECUTE IF HIGHER OR SAME [,PAIR]                        IHS[P] (SLO[P])

FORMAT:
                    OP1       OP2              NORM     MOD      BUS
  <INSTR>             R3   M C4                4+S      7+S      1+S
  <INSTR>  *)         R3   M CN4               4+S      7+S      1+S
  <INSTR>             X3   M C4                4+S      7+S      2+S
  <INSTR>  *)         X3   M CN4               4+S      7+S      2+S
  <INSTR>             R3        R3             5+S      6+S      1+S
  <INSTR>             R3   M BO.X3             5+S      6+S      2+S
  <INSTR>             X3        R3             5+S      6+S      2+S
  <INSTR>             X3   M BO.X3             5+S      6+S      3+S
                                               (NO SKIP: S=0)
  *) ONLY SEQ[P], SNE[P].                      (SKIP:    S=1)

FUNCTION:
  IF CASE INSTR OF
      (SEQ[P],INE[P]: OP1= OP2,
       SNE[P],IEQ[P]: OP1<>OP2,
       SGE[P],ILT[P]: OP1>=OP2, % SIGNED
       SLT[P],IGE[P]: OP1< OP2, % SIGNED
       SHS[P],ILO[P]: OP1>=OP2, % UNSIGNED
       SLO[P],IHS[P]: OP1< OP2  % UNSIGNED
      ) THEN
    IF PAIR THEN LOC:= LOC+3     % SKIP 2
    ELSE         LOC:= LOC+2     % SKIP 1
  ELSE           LOC:= LOC+1;    % SKIP 0
  CLEAR MODIFICATION;

DESCRIPTION:
  CALCULATES THE CONDITION SPECIFIED BY THE INSTRUCTION. IF
  FULFILLED, A SKIP IS PERFORMED. IF PAIR IS FALSE, ONE MEMORY
  LOCATION IS SKIPPED, IF PAIR IS TRUE, TWO MEMORY LOCATIONS
  ARE SKIPPED. THE CONDITIONS ARE:
    SEQ[P],INE[P]: THE TWO OPERANDS ARE EQUAL.
    SNE[P],IEQ[P]: THE TWO OPERANDS ARE NOT EQUAL.
    SGE[P],ILT[P]: OP1 IS GREATER THAN OR EQUAL TO OP2, SIGNED.
    SLT[P],IGE[P]: OP1 IS LESS THAN OP2, SIGNED.
    SHS[P],ILO[P]: OP1 IS GREATER THAN OR EQUAL TO OP2, UNSIGNED.
    SLO[P],IHS[P]: OP1 IS LESS THAN OP2, UNSIGNED.
```

REMARK:
SOME COMBINATIONS OF SKIP AND JUMP INSTRUCTIONS CAN BE BETTER DONE BY MEANS OF THE CONDITIONAL JUMP INSTRUCTIONS.

CODE:

		OP1	OP2	BINARY	HEXA
SEQ	(INE)	R3	C4	1 R3 C4 0111 1110	807E
SEQ	(INE)	R3	CN4	0 R3 CN4 0111 1110	007E
SEQ	(INE)	X3	C4	1 X3 C4 0111 1111	807F
SEQ	(INE)	X3	CN4	0 X3 CN4 0111 1111	007F
SEQ	(INE)	R3	R3	0 R3 0 R3 1111 1010	00FA
SEQ	(INE)	R3	B0.X3	0 R3 1 X3 1111 1010	08FA
SEQ	(INE)	X3	R3	1 X3 0 R3 1111 1010	80FA
SEQ	(INE)	X3	B0.X3	1 X3 1 X3 1111 1010	88FA
SNE	(IEQ)	R3	C4	1 R3 C4 0111 1100	807C
SNE	(IEQ)	R3	CN4	0 R3 CN4 0111 1100	007C
SNE	(IEQ)	X3	C4	1 X3 C4 0111 1101	807D
SNE	(IEQ)	X3	CN4	0 X3 CN4 0111 1101	007D
SNE	(IEQ)	R3	R3	0 R3 0 R3 1111 1000	00F8
SNE	(IEQ)	R3	B0.X3	0 R3 1 X3 1111 1000	08F8
SNE	(IEQ)	X3	R3	1 X3 0 R3 1111 1000	80F8
SNE	(IEQ)	X3	B0.X3	1 X3 1 X3 1111 1000	88F8
SGE	(ILT)	R3	C4	1 R3 C4 0100 0010	8042
SGE	(ILT)	X3	C4	1 X3 C4 0100 0011	8043
SGE	(ILT)	R3	R3	0 R3 0 R3 1111 0110	00F6
SGE	(ILT)	X3	R3	1 X3 0 R3 1111 0110	80F6
SGE	(ILT)	R3	X3	0 R3 1 X3 1111 0110	08F6
SGE	(ILT)	X3	X3	1 X3 1 X3 1111 0110	88F6
SLT	(IGE)	R3	C4	1 R3 C4 0100 0000	8040
SLT	(IGE)	X3	C4	1 X3 C4 0100 0001	8041
SLT	(IGE)	R3	R3	0 R3 0 R3 1111 0100	00F4
SLT	(IGE)	X3	R3	1 X3 0 R3 1111 0100	80F4
SLT	(IGE)	R3	X3	0 R3 1 X3 1111 0100	08F4
SLT	(IGE)	X3	X3	1 X3 1 X3 1111 0100	88F4
SHS	(ILO)	R3	C4	1 R3 C4 0100 0110	8046
SHS	(ILO)	X3	C4	1 X3 C4 0100 0111	8047
SHS	(ILO)	R3	R3	0 R3 0 R3 1111 0111	00F7
SHS	(ILO)	X3	R3	1 X3 0 R3 1111 0111	80F7
SHS	(ILO)	R3	X3	0 R3 1 X3 1111 0111	08F7
SHS	(ILO)	X3	X3	1 X3 1 X3 1111 0111	88F7
SLO	(IHS)	R3	C4	1 R3 C4 0100 0100	8044
SLO	(IHS)	X3	C4	1 X3 C4 0100 0101	8045
SLO	(IHS)	R3	R3	0 R3 0 R3 1111 0101	00F5
SLO	(IHS)	X3	R3	1 X3 0 R3 1111 0101	80F5
SLO	(IHS)	R3	X3	0 R3 1 X3 1111 0101	08F5
SLO	(IHS)	X3	X3	1 X3 1 X3 1111 0101	88F5

SEQP (INEP)	R3	C4	1	R3		C4	0	0	1	1	1	1	1	0	803E
SEQP (INEP)	R3	CN4	0	R3		CN4	0	0	1	1	1	1	1	0	003E
SEQP (INEP)	X3	C4	1	X3		C4	0	0	1	1	1	1	1	1	803F
SEQP (INEP)	X3	CN4	0	X3		CN4	0	0	1	1	1	1	1	1	003F
SEQP (INEP)	R3	R3	0	R3	0	R3	1	0	1	1	1	0	1	0	00BA
SEQP (INEP)	R3	BO.X3	0	R3	1	X3	1	0	1	1	1	0	1	0	08BA
SEQP (INEP)	X3	R3	1	X3	0	R3	1	0	1	1	1	0	1	0	80BA
SEQP (INEP)	X3	BO.X3	1	X3	1	X3	1	0	1	1	1	0	1	0	88BA
SNEP (IEQP)	R3	C4	1	R3		C4	0	0	1	1	1	1	0	0	803C
SNEP (IEQP)	R3	CN4	0	R3		CN4	0	0	1	1	1	1	0	0	003C
SNEP (IEQP)	X3	C4	1	X3		C4	0	0	1	1	1	1	0	1	803D
SNEP (IEQP)	X3	CN4	0	X3		CN4	0	0	1	1	1	1	0	1	003D
SNEP (IEQP)	R3	R3	0	R3	0	R3	1	0	1	1	1	0	0	0	00B8
SNEP (IEQP)	R3	BO.X3	0	R3	1	X3	1	0	1	1	1	0	0	0	08B8
SNEP (IEQP)	X3	R3	1	X3	0	R3	1	0	1	1	1	0	0	0	80B8
SNEP (IEQP)	X3	BO.X3	1	X3	1	X3	1	0	1	1	1	0	0	0	88B8
SGEP (ILTP)	R3	C4	1	R3		C4	0	0	0	0	0	0	1	0	8002
SGEP (ILTP)	X3	C4	1	X3		C4	0	0	0	0	0	0	1	1	8003
SGEP (ILTP)	R3	R3	0	R3	0	R3	1	0	1	1	0	1	1	0	00B6
SGEP (ILTP)	X3	R3	1	X3	0	R3	1	0	1	1	0	1	1	0	80B6
SGEP (ILTP)	R3	X3	0	R3	1	X3	1	0	1	1	0	1	1	0	08B6
SGEP (ILTP)	X3	X3	1	X3	1	X3	1	0	1	1	0	1	1	0	88B6
SLTP (IGEP)	R3	C4	1	R3		C4	0	0	0	0	0	0	0	0	8000
SLTP (IGEP)	X3	C4	1	X3		C4	0	0	0	0	0	0	0	1	8001
SLTP (IGEP)	R3	R3	0	R3	0	R3	1	0	1	1	0	1	0	0	00B4
SLTP (IGEP)	X3	R3	1	X3	0	R3	1	0	1	1	0	1	0	0	80B4
SLTP (IGEP)	R3	X3	0	R3	1	X3	1	0	1	1	0	1	0	0	08B4
SLTP (IGEP)	X3	X3	1	X3	1	X3	1	0	1	1	0	1	0	0	88B4
SHSP (ILOP)	R3	C4	1	R3		C4	0	0	0	0	0	1	1	0	8006
SHSP (ILOP)	X3	C4	1	X3		C4	0	0	0	0	0	1	1	1	8007
SHSP (ILOP)	R3	R3	0	R3	0	R3	1	0	1	1	0	1	1	1	00B7
SHSP (ILOP)	X3	R3	1	X3	0	R3	1	0	1	1	0	1	1	1	80B7
SHSP (ILOP)	R3	X3	0	R3	1	X3	1	0	1	1	0	1	1	1	08B7
SHSP (ILOP)	X3	X3	1	X3	1	X3	1	0	1	1	0	1	1	1	88B7
SLOP (IHSP)	R3	C4	1	R3		C4	0	0	0	0	0	1	0	0	8004
SLOP (IHSP)	X3	C4	1	X3		C4	0	0	0	0	0	1	0	1	8005
SLOP (IHSP)	R3	R3	0	R3	0	R3	1	0	1	1	0	1	0	1	00B5
SLOP (IHSP)	X3	R3	1	X3	0	R3	1	0	1	1	0	1	0	1	80B5
SLOP (IHSP)	R3	X3	0	R3	1	X3	1	0	1	1	0	1	0	1	08B5
SLOP (IHSP)	X3	X3	1	X3	1	X3	1	0	1	1	0	1	0	1	88B5

****
********** SHIFT INSTRUCTIONS **********
****

THE SHIFT INSTRUCTIONS ARE DIVIDED INTO THE GROUPS:
  SHIFT SINGLE
  SHIFT LONG

THE TWO GROUPS ARE WORKING ON A SINGLE WORD OPERAND AND A DOUBLE WORD OPERAND RESPECTIVELY. A DOUBLE WORD OPERAND IS THE SUCCESSOR OF THE OPERAND CONCATENATED WITH THE OPERAND. THE SUCCESSOR BEING THE NEXT REGISTER OR THE NEXT MEMORY LOCATION.

LEFT SHIFTS AND RIGHT SHIFTS ARE PERFORMED BY DIFFERENT INSTRUCTIONS. THIS HAS THE CONSEQUENCE THAT A SHIFT INSTRUCTION CANNOT BE MODIFIED TO CHANGE SHIFT DIRECTION.

IF A SHIFT INSTRUCTION IS PRECEDED BY A MODIFY INSTRUCTION THE NUMBER OF BIT POSITIONS SHIFTED IS DEFINED BY THE MODIFY REGISTER (MOD) MODULO 16, THUS IGNORING THE SECOND OPERAND OF THE SHIFT INSTRUCTION. IF UNMODIFIED AND THE SECOND OPERAND SPECIFIES ZERO SHIFTS, 16 SHIFTS ARE PERFORMED.

```
****************     SHIFT SINGLE     *************** 1 2 3 4 5 *****

INSTRUCTION:
  SHIFT LEFT CYCLICALLY                                  SLC
  SHIFT RIGHT ARITHMETICALLY                             SRA
  SHIFT LEFT LOGICALLY                                   SLL
  SHIFT RIGHT LOGICALLY                                  SRL
SYNONYME:
  SHIFT LEFT ARITHMETICALLY                              SLA     (SLL)

FORMAT:
                OP1       OP2                NORM     MOD      BUS
  <INSTR>         R3 (M)C4                   5+S      9+S      1
  <INSTR>         X3 (M)C4                   5+S      9+S      3
                                             (S=SHIFTS)

FUNCTION:
  SHIFTS:= IF MODIFIED THEN   MOD MODULO 16
      ELSE IF C4=0 THEN                  16
      ELSE                               C4;
  FOR I:= 1 STEP 1 UNTIL SHIFTS DO
    CASE INSTR OF
      BEGIN
        SLC: OP1:= OP1.[15:1]&OP1[15:14:15];
        SRA: OP1:=         OP1&OP1[14:15:15];
        SLL: OP1:=           0&OP1[15:14:15];
        SRL: OP1:=           0&OP1[14:15:15];
      END;
  CLEAR MODIFICATION;

DESCRIPTION:
  OP1 IS SHIFTED AS MANY BIT POSITIONS AS SPECIFIED BY OP2 OR
  THE MODIFY REGISTER. IF UNMODIFIED AND ZERO SHIFTS ARE SPECIFIED
  16 SHIFTS ARE PERFORMED. THE SHIFTS ARE PERFORMED AS INDICATED
  IN THE INSTRUCTION:
    SLC: SHIFT LEFT CYCLICALLY. THE 16 BIT OPERAND IS ROTATED
         LEFT. NO CARRY OR OTHER EXTERNAL BITS ARE INCORPORATED.
    SRA: SHIFT RIGHT ARITHMETICALLY. THE 16 BIT OPERAND IS
         SHIFTED RIGHT WITH SIGN EXTENSION IN THE UPPER BITS.
    SLL: SHIFT LEFT LOGICALLY. THE 16 BIT OPERAND IS SHIFTED
         LEFT WITH ZERO EXTENSION IN THE LOWER BITS.
    SRL: SHIFT RIGHT LOGICALLY. THE 16 BIT OPERAND IS SHIFTED
         RIGHT WITH ZERO EXTENSION IN THE UPPER BITS.

NOTICE:
  THE CONDITION CODES ARE UNCHANGED.

CODE:
                OP1       OP2      BINARY                          HEXA
  SLC             R3        C4     1 R3 C4 1010 0010              80A2
  SLC             X3        C4     1 X3 C4 1110 0010              80E2
  SRA             R3        C4     1 R3 C4 0010 0110              8026
  SRA             X3        C4     1 X3 C4 0110 0110              8066
  SLL   (SLA)     R3        C4     1 R3 C4 1010 0000              80A0
  SLL   (SLA)     X3        C4     1 X3 C4 1110 0000              80E0
  SRL             R3        C4     1 R3 C4 0010 0100              8024
  SRL             X3        C4     1 X3 C4 0110 0100              8064
```

INSTRUCTION:

SHIFT LEFT LOGICALLY, LONG	SLLL
SHIFT RIGHT LOGICALLY, LONG	SRLL

FORMAT:

	OP1	OP2	NORM	MOD	BUS
<INSTR>	R3	(M)C4	7+S	11+S	1
<INSTR>	X3	(M)C4	8+S	12+S	5
			(S=SHIFTS)		

FUNCTION:

```
SHIFTS:= IF MODIFIED THEN   MOD MODULO 16
     ELSE IF C4=0 THEN                  16
     ELSE                               C4;
DEFINE OP1::= SUCCESSOR(OP1) CON OP1;
FOR I:= 1 STEP 1 UNTIL SHIFTS DO
  CASE INSTR OF
    BEGIN
      SLLL: OP1:= 0&OP1[31:30:31];
      SRLL: OP1:= 0&OP1[30:31:31];
    END;
CLEAR MODIFICATION;
```

DESCRIPTION:

THE SUCCESSOR OF OP1 CONCATENATED WITH OP1 IS THE OPERAND SHIFTED. THE NUMBER OF BIT POSITIONS SHIFTED IS DEFINED BY OP2 OR THE MODIFY REGISTER. IF UNMODIFIED AND ZERO SHIFTS ARE SPECIFIED 16 SHIFTS ARE PERFORMED. THE SHIFTS ARE PERFORMED AS INDICATED IN THE INSTRUCTION:

- SLLL: SHIFT LEFT LOGICALLY. THE 32 BIT OPERAND IS SHIFTED LEFT WITH ZERO EXTENTION IN THE LOWER BITS.
- SRLL: SHIFT RIGHT LOGICALLY. THE 32 BIT OPERAND IS SHIFTED RIGHT WITH ZERO EXTENSION IN THE UPPER BITS.

CODE:

	OP1	OP2	BINARY					HEXA
SLLL	R3	C4	0	R3	C4	1010	0001	00A1
SLLL	X3	C4	0	X3	C4	1110	0001	00E1
SRLL	R3	C4	0	R3	C4	0010	0101	0025
SRLL	X3	C4	0	X3	C4	0110	0101	0065

****
********** SINGLE BIT OPERATIONS **********
****

THESE INSTRUCTIONS WORK ON TWO OPERANDS. THE FIRST DEFINES A WORD, THE SECOND DEFINES A BIT NUMBER IN THE WORD. THE SPECIFIED BIT CAN BE CLEARED OR SET DIRECTLY, OR IT CAN BE CLEARED OR SET UNDER SEMAPHORE PROTECTION. THE SEMAPHORE PROTECTION GUARANTEES THAT THE ENTIRE INSTRUCTION IS EXECUTED WITHOUT INTERVENTION FROM ANY OTHER DATA TRANSFER ON THE MAIN BUS.

INSTRUCTION:
CLEAR SINGLE BIT CLRS

FORMAT:

	OP1	OP2	NORM	MOD	BUS
CLRS	R3	R3	8	9	1
CLRS	R3	M 80.X3	8	9	2
CLRS	X3	R3	8	9	3
CLRS	X3	M 80.X3	8	9	4

FUNCTION:
OP1:= OP1&0[OP2 MODULO 16:0:1];
CLEAR MODIFICATION;

DESCRIPTION:
CLEARS A BIT IN OP1. THE BIT NUMBER IS DEFINED BY OP2 MODULO 16.

CODE:

	OP1	OP2	BINARY	HEXA
CLRS	R3	R3	0 R3 0 R3 1000 1000	0088
CLRS	R3	80.X3	0 R3 1 X3 1000 1000	0888
CLRS	X3	R3	1 X3 0 R3 1000 1001	8089
CLRS	X3	80.X3	1 X3 1 X3 1000 1001	8889

********** SET SINGLE BIT ********** 1 2 3 4 5 *****

INSTRUCTION:
SET SINGLE BIT SETS

FORMAT:

	OP1	OP2	NORM	MOD	BUS
SETS	R3	R3	8	9	1
SETS	R3	M B0.X3	8	9	2
SETS	X3	R3	8	9	3
SETS	X3	M B0.X3	8	9	4
SETS	R3	C4	4	5	1
SETS	X3	C4	4	5	3

FUNCTION:
OP1:= OP1&1[OP2 MODULO 16:0:1];
CLEAR MODIFICATION;

DESCRIPTION:
SETS A BIT IN OP1. THE BIT NUMBER IS DEFINED BY OP2 MODULO 16.

CODE:

	OP1	OP2	BINARY	HEXA
SETS	R3	R3	0 R3 0 R3 0101 0010	0052
SETS	R3	B0.X3	0 R3 1 X3 0101 0010	0852
SETS	X3	R3	1 X3 0 R3 0101 0011	8053
SETS	X3	B0.X3	1 X3 1 X3 0101 0011	8853
SETS	R3	C4	0 R3 C4 1010 0011	00A3
SETS	X3	C4	0 X3 C4 1110 0011	00E3

INSTRUCTION:
RELEASE SINGLE BIT RELS

FORMAT:

	OP1	OP2	NORM	MOD	BUS
RELS	X3	C4	5	5	4

FUNCTION:
SET SEMAPHORE BLOCKING;
OP1:= OP1&0[OP2:0:1];
CLEAR SEMAPHORE BLOCKING;

DESCRIPTION:
CLEARS A BIT IN OP1. THE BIT NUMBER IS DEFINED BY OP2. THE INSTRUCTION IS PERFORMED UNDER SEMAPHORE PROTECTION.

NOTICE:
MODIFICATION HAS NO INFLUENCE ON THE INSTRUCTION. MODIFICATION IS NOT CLEARED.

CODE:

	OP1	OP2	BINARY	HEXA
RELS	X3	C4	0 X3 C4 0001 0001	0011

********** RESERVE SINGLE BIT ********* 1 2 3 4 5 *****

INSTRUCTION:
RESERVE SINGLE BIT RESS

FORMAT:

	OP1	OP2	NORM	MOD	BUS
RESS	X3	C4	4+3*R	4+3*R	3+2*R
			(EXECUTE NEXT:		R=0)
			(RESERVE AND SKIP:		R=1)

```
FUNCTION:
  SET SEMAPHORE BLOCKING;
  IF OP1.[OP2:1]=1 THEN
    BEGIN
      OP1:= OP1&1[OP2:0:1];
      LOC:= LOC+2
    END
  ELSE LOC:= LOC+1;
  CLEAR SEMAPHORE BLOCKING;
```

DESCRIPTION:
TESTS A BIT IN OP1. IF ONE, NO ACTION IS TAKEN. IF ZERO, THE BIT IS SET, AND THE NEXT INSTRUCTION IS SKIPPED. THE BIT NUMBER IS DEFINED BY OP2. THE INSTRUCTION IS PERFORMED UNDER SEMAPHORE PROTECTION.

NOTICE:
MODIFICATION HAS NO INFLUENCE ON THE INSTRUCTION. MODIFICATION IS NOT CLEARED.

CODE:

	OP1	OP2	BINARY	HEXA
RESS	X3	C4	0 X3 C4 0101 0001	0051

****
********** INPUT/OUTPUT INSTRUCTIONS **********
****

THE INPUT/OUTPUT INSTRUCTIONS ARE DIVIDED INTO THE GROUPS:
  I/O DEVICE INSTRUCTIONS
  FUNC MODULE INSTRUCTIONS

FOR CR80S AND CR80M UNMAPPED:
  AN I/O DEVICE INSTRUCTION ADDRESSES A DEVICE ON THE MAIN BUS.
  THE POSSIBLE ADDRESSES ARE 0 THROUGH 63.

FOR CR80M MAPPED:
  NO I/O DEVICE INSTRUCTIONS EXISTS.

A FUNC MODULE INSTRUCTION ADDRESSES A FUNC MODULE ON A SUB BUS.
THE POSSIBLE ADDRESSES ARE 0 THRU 7.

********** I/O DEVICE INSTRUCTIONS **** 1 2 3 4 * *****

INSTRUCTION:

SENSE I/O	SIO
CONTROL I/O	CIO
READ I/O	RIO
WRITE I/O	WIO

FORMAT:

		OP1	OP2	NORM	MOD	BUS
<INSTR>	pp	R3	(M)R3	5	5	2
<INSTR>	pp	X3	(M)R3	5	5	3

FUNCTION:

```
DEV:= OP2+MOD;
CASE INSTR OF
  BEGIN
    SIO: OP1:= DEVICE CONTROL REGISTER(DEV);
    CIO: DEVICE CONTROL REGISTER(DEV):= OP1;
    RIO: OP1:= DEVICE DATA REGISTER(DEV);
    WIO: DEVICE DATA REGISTER(DEV):= OP1
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:

THE SIX LEAST SIGNIFICANT BITS OF OP2 PLUS MOD ARE THE DEVICE ADDRESS WHILE THE REMAINING BITS [15:10] MAY BE USED AS COMMAND WITH A DEVICE DEPENDENT MEANING. THE DATA ARE DEFINED BY OP1. THE OPERATIONS ARE:

- SIO: SENSE A CONTROL WORD FROM THE DEVICE.
- CIO: TRANSFER A CONTROL WORD TO THE DEVICE.
- RIO: READ A DATA WORD FROM THE DEVICE.
- WIO: WRITE A DATA WORD TO THE DEVICE.

CODE:

		OP1	OP2	BINARY	HEXA
SIO	pp	R3	R3	0 R3 Φ R3 1001 0110	0896
SIO	pp	X3	R3	0 X3 Φ R3 1101 0110	08D6
CIO	pp	R3	R3	1 R3 Φ R3 1001 0110	8896
CIO	pp	X3	R3	1 X3 Φ R3 1101 0110	88D6
RIO	pp	R3	R3	0 R3 Φ R3 1001 0111	0897
RIO	pp	X3	R3	0 X3 Φ R3 1101 0111	08D7
WIO	pp	R3	R3	1 R3 Φ R3 1001 0111	8897
WIO	pp	X3	R3	1 X3 Φ R3 1101 0111	88D7

OP1 OP2

********** FUNC MODULE INSTRUCTIONS *** * 2 * * * *****

INSTRUCTION:

FUNC			FNC

FORMAT:

	OP1	OP2	NORM	MOD	BUS
FNC	R3		4+I	4+I	2
FNC	X3		4+I	4+I	3
FNC	R3	X3	4	4	2
FNC	X3	X3	4	4	3
			(OUTPUT: I=0)		
			(INPUT: I=1)		

```
FUNCTION:
  IF ABSENT(OP2) THEN DEFINE OP2::= REG[OP1.[2:3]];
  OUTPUT:= OP1.[15:1]=1;
  CONTR:= OP1.[14:3];
  ADDR:= OP1.[10:3];
  IF OUTPUT THEN   FUNC(ADDR,CONTR):= OP2
  ELSE             OP2:= FUNC(ADDR,CONTR);
```

DESCRIPTION:

A FUNC MODULE IS ACTIVATED. THE PARAMETERS ARE DEFINED BY OP1:
- OUTPUT: OP1.[15:1]=1 , BOOLEAN, DEFINES THE TRANSFER DIRECTION.
- CONTR: OP1.[14:3] , INTEGER, DEFINES THE CONTROL INSTRUCTION.
- ADDR: OP1.[10:3] , INTEGER, DEFINES THE FUNC MODULE ADDRESS.

A TRANSFER IS PERFORMED BETWEEN OP2 AND THE FUNC MODULE.

NOTICE:

MODIFICATION HAS NO INFLUENCE ON THE INSTRUCTION. MODIFICATION IS NOT CLEARED.

CODE:

	OP1	OP2	BINARY	HEXA
FNC	R3		1 R3 0 0 0 0 0 0 0 1	8711
FNC	X3		1 X3 0 0 0 0 0 1 0 1	8751
FNC	R3	X3	1 R3 1 X3 0 0 0 1	8811
FNC	X3	X3	1 X3 1 X3 0 1 0 1	8851

OP1 OP2

```
*************************************************************************
***********     PRIVILEGED READ/WRITE INSTRUCTIONS     ***************
*************************************************************************
```

THE PRIVILEGED READ/WRITE INSTRUCTIONS ARE USED IN MAPPED
CROM SYSTEMS FOR ACCESSING PRIVILEGED MEMORY AREAS.

****** PRIVILEGED MEMORY MAP INSTRUCTIONS **** * * * * 5 *****

INSTRUCTION:

PRIVILEGED MEMORY MAP READ	SIO
PRIVILEGED MEMORY MAP WRITE	CIO

FORMAT:

		OP1	OP2	NORM	MOD	BUS
<INSTR>	pp	R3	(M)R3	5	5	2
<INSTR>	pp	X3	(M)R3	5	5	3

FUNCTION:

```
CASE INSTR OF
  BEGIN
    SIO: OP1:= MEMORY MAP LOCATION(MOD + OP2);
    CIO: MEMORY MAP LOCATION(MOD + OP2):= OP1;
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:

THESE INSTRUCTIONS ARE USED TO ACCESS THE INTERNAL CONTROL MEMORY AND ADDRESS TRANSLATION MEMORY TABLES OF THE MAP MODULE.

CODE:

		OP1	OP2	BINARY																HEXA
SIO	pp	R3	R3	0	R3	0	R3	1	0	0	1	0	1	1	0					0896
SIO	pp	X3	R3	0	X3	0	R3	1	1	0	1	0	1	1	0					08D6
CIO	pp	R3	R3	1	R3	0	R3	1	0	0	1	0	1	1	0					8896
CIO	pp	X3	R3	1	X3	0	R3	1	1	0	1	0	1	1	0					88D6

********** PRIVILEGED MEMORY INSTRUCTIONS **** * * * * 5 *****

INSTRUCTION:

PRIVILEGED MEMORY READ	RIO
PRIVILEGED MEMORY WRITE	WIO

FORMAT:

		OP1	OP2	NORM	MOD	BUS
<INSTR>	pp	R3	(M)R3	5	5	2
<INSTR>	pp	X3	(M)R3	5	5	3

FUNCTION:

```
CASE INSTR OF
  BEGIN
    RIO: OP1:= PRIVILEGED MEMORY(MOD + OP2);
    WIO: PRIVILEGED MEMORY(MOD + OP2):= OP1
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:

THESE INSTRUCTIONS ARE USED TO ACCESS PRIVILEGED MEMORY IN CONTROLLER MODULES OTHER THAN THE MEMORY MAP MODULE. THE 4 MOST SIGNIFICANT BITS OF OP2 PLUS MOD IS THE CHANNEL UNIT ADDRESS, AND THE 12 LEAST SIGNIFICANT BITS DEFINE A PRIVILEGED MEMORY ADDRESS WITHIN THE CHANNEL UNIT.

CODE:

		OP1	OP2	BINARY	HEXA
RIO	pp	R3	R3	0 R3 0 R3 1001 0111	0897
RIO	pp	X3	R3	0 X3 0 R3 1101 0111	C8D7
WIO	pp	R3	R3	1 R3 0 R3 1001 0111	8897
WIO	pp	X3	R3	1 X3 0 R3 1101 0111	88D7

```
**************************************************************************
****************     SPECIAL INSTRUCTIONS     ***********************
**************************************************************************

THE SPECIAL INSTRUCTIONS COVERS THE FOLLOWING AREA:S
  LOAD AND SAVE OF THE PROCESS STATUS WORD (PSW)
  LOAD AND SAVE OF THE TIMER
  LOAD AND SAVE PROCESS
  LOAD AND EXECUTE MICRO-RAM
  MONITOR CALL
  NO OPERATION AND TRAP
  CPU INTERRUPT AND SIMULATED I/O INTERRUPT
  CALL, RETURN
  CHANGE STACK POINTER
  INDEX
```

INSTRUCTION:

LOAD STATUS	LDS
SAVE STATUS	SVS
LOAD TIMER	LDT
SAVE TIMER	SVT
LOAD MASK	LDM

FORMAT:

	OP1	OP2	NORM	MOD	BUS
LDS	pp	R3	6	7	1
LDS	pp	M B0.X3	6	7	2
SVS		R3	7	8	1
SVS		M B0.X3	7	8	3
LDT	pp	R3	6	7	1
LDT	pp	M B0.X3	6	7	2
SVT		R3	8	9	1
SVT		M B0.X3	8	9	3
LDM	pp	C4	10	11	1

FUNCTION:

```
CASE INSTR OF
  BEGIN
    LDS: PSW:= OP2;
    SVS: OP2:= PSW;
    LDT: TIMER:= OP2;
    SVT: OP2:= TIMER;
    LDM: PSW:= PSW&OP2[15:2:3];
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:

LDS: LOADS THE PROGRAM STATUS WORD (PSW).
SVS: SAVES THE PROGRAM STATUS WORD (PSW).
LOT: LOAOS THE TIMER REGISTER.
SVT: SAVES THE TIMER REGISTER.
LDM: LOADS THE INTERRUPT MASK BITS INTO PSW.

CODE:

	OP1	OP2	BINARY	HEXA
LDS	pp	R3	1 1 1 0 0 R3 1 0 1 1 1 1 0 0	E0BC
LDS	pp	B0.X3	1 1 1 0 1 X3 1 0 1 1 1 1 0 0	E8BC
SVS		R3	1 1 0 1 0 R3 1 0 1 1 1 1 0 0	D0BC
SVS		B0.X3	1 1 0 1 1 X3 1 0 1 1 1 1 0 0	D8BC
LDT	pp	R3	0 1 0 1 0 R3 1 0 1 1 1 1 0 0	50BC
LDT	pp	B0.X3	0 1 0 1 1 X3 1 0 1 1 1 1 0 0	58BC
SVT		R3	1 1 0 0 0 R3 1 0 1 1 1 1 0 0	C0BC
SVT		B0.X3	1 1 0 0 1 X3 1 0 1 1 1 1 0 0	C8BC
LDM	pp	C4	0 0 0 1 C4 1 0 1 1 1 1 1 0	00BE

********** EXECUTION REGISTER HANDLING * * * * * 5 *****

INSTRUCTION:
SAVE EXR SVE

FORMAT:

	OP1	OP2	NORM	MOD	BUS
SVE		R3	X	X	X
SVE		M B0.X3	X	X	X

FUNCTION:
OP2:= EXR;
CLEAR MODIFICATION;

DESCRIPTION:
SAVES THE EXECUTION REGISTER.

CODE:

	OP1	OP2	BINARY	HEXA
SVE		R3	0 1 1 0 0 R3 1 0 1 1 1 1 0 1	60B0
SVE		B0.X3	0 1 1 0 1 X3 1 0 1 1 1 1 0 1	68B0

********** SET PRIORITY ********** * * * * 5 *****

INSTRUCTION:
SET PRIORITY STP

FORMAT:

	OP1	OP2	NORM	MOD	BUS
STP p		M C4	-	-	-

FUNCTION:
EXR.[11:4]:= OP2;
PRIORITY:= OP2;

DESCRIPTION:
THE PRIORITY FIELD IN THE EXECUTION REGISTER IS UPDATED. THE MAP MODULE IS UPDATED ACCORDINGLY.

CODE:

	OP1	OP2	BINARY	HEXA
STP p		C4	0 0 0 1 C 4 1 0 1 1 1 1 0 0	10BC

```
**********************     SAVE PROCESS     ********************* 1 2 3 4 *******

INSTRUCTION:
  SAVE PROCESS                                                    SVP

FORMAT:
                    OP1         OP2                    NORM      MOD       BUS
  SVP   p                                              23        24        15
  SVP   p                       L4                     23        24        15

FUNCTION:
  IF ABSENT(OP2) THEN DEFINE OP2::= LOC+1;
  FOR I:= 0 STEP 1 UNTIL 7 DO MEM[BASE+I]:= REG[I];
  MEM[BASE+8]:=  BASE;
  MEM[BASE+9]:=  BASEMOD;  COMMENT: LATER CHANGED TO BASE;
  MEM[BASE+10]:= PROG;
  MEM[BASE+11]:= OP2;
  MEM[BASE+12]:= TIMER;
  MEM[BASE+13]:= PSW;
  PSW:= PSW&7[15:2:3];
  CLEAR MODIFICATION;

DESCRIPTION:
  SAVES THE PROCESS AT ITS BASE IN MEMORY. THE OPERAND DEFINES
  THE LOCATION IN WHICH THE PROCESS WILL CONTINUE AFTER RELOAD-
  ING. IF NO OPERAND, NEXT LOCATION IS USED. THE PSW, AFTER BEING
  SAVED, IS CHANGED TO DISABLE ALL INTERRUPTS.

REMARK:
  THE INSTRUCTION IS FOLLOWED BY THE NEXT INSTRUCTION, I.E. THE
  SAVED PROCESS CONTINUES EXECUTION.

CODE:
                    OP1         OP2    BINARY                              HEXA
  SVP   p                              0 0 1 0 | 0 0 0 0 | 1 0 1 1 | 1 1 1 0   20BE
  SVP   p                       L4     0 0 1 0 |   L4    | 1 0 1 1 | 1 1 1 0   20BE
```

INSTRUCTON:

LOAD NOLINKED	LDN
LOAD PROCESS	LDP

FORMAT:

	OP1	OP2	NORM	MOD	BUS
LDN p		R3	22	23	16
LDN p	M	80.X3	22	23	17
LDP p		R3	25	26	18
LDP p	M	80.X3	25	26	19

FUNCTION:

```
WORK:= BASE;
FOR I:= 0 STEP 1 UNTIL 7 DO REG[I]:= MEM[OP2+I];
BASE:=    MEM[OP2+8];
BASEMOD:= MEM[OP2+9];
PROG:=    MEM[OP2+10];
LOC:=     MEM[OP2+11];
TIMER:=   MEM[OP2+12];
PSW:=     MEM[OP2+13];
CASE INSTR OF
  BEGIN
    LDN: COMMENT: NO PROCESS LINK IS SAVED;
    LDP: MEM[OP2+14]:= WORK;
  END;
CLEAR MODIFICATION;
```

DESCRIPTION:

LOADS A PROCESS INTO THE CPU. THE PROCESS BASE IS DEFINED BY OP2. THE BASE FOR THE PROCESS EXECUTING THE LDP INSTRUCTION IS SAVED IN LOCATION 14 OF THE LOADED PROCESS. THE CPU IS LOADED WITH THE GENERAL REGISTERS AND THE SPECIAL REGISTERS UPTO AND INCLUDING PSW, WHICH IS LOCATION 13 IN THE PROCESS AREA. THE BASE USED BY THE LOADED PROCESS IS THE ONE LOADED BY THE INSTRUCTION. THIS MAY DIFFER FROM OP2.

REMARK:

THE MEMORY SECTION FROM WHICH THE PROCESS IS LOADED IS DEFINED BY BITS 3 AND 2 IN OP2. THE MEMORY SECTION AND THE PRIORITY USED BY THE LOADED PROCESS ARE TAKEN FROM THE PSW LOADED.

CODE:

	OP1	OP2	BINARY					HEXA
LDN p		R3	0100	0	R3	1011	1100	408C
LDN p		80.X3	0100	1	X3	1011	1100	488C
LDP p		R3	0000	0	R3	1011	1100	008C
LDP p		80.X3	0000	1	X3	1011	1100	088C

********** LOAD RAM ********** 1 2 * * * *****

INSTRUCTION:

LOAD RAM LDR

FORMAT:

	OP1	OP2	NORM	MOD	BUS
LDR	p	R3	24	25	18
LDR	p	M B0.X3	24	25	19

FUNCTION:

```
BIT:= CASE OP2.[15:2] OF (0: 47, 1: 31, 2: 15);
ADDR:= OP2.[13:2]*16;
X:= OP2.[10:3];
FOR I:= 0 STEP 1 UNTIL 15 DO
  RAM[ADDR+I]:= RAM[ADDR+I]&MEM[REG[X]+BASE+I][BIT:15:16];
CLEAR MODIFICATION;
```

DESCRIPTION:

LOADS THE RAM WITH 16 WORDS FROM MEMORY. THE RAM IS PART OF THE MICRO PROGRAM CONSISTING OF 48-BIT WORDS. OP2 SPECIFIES THE LOAD PARAMETERS:

```
BIT:  CASE OP2.[15:2] OF (0: 47, 1: 31, 2: 15)
ADDR: OP2.[13:2]*16
X:    OP2.[10:3]
```

THE ADDRESSES OF THE RAM LOCATIONS INVOLVED ARE ADDR THRU ADDR+15. THE 16-BIT MEMORY WORDS LOADED INTO THE RAM ARE TAKEN FROM THE MEMORY ADDRESSES BASE+X THRU BASE+X+15. THE WORDS ARE LOADED INTO BITS BIT THRU BIT-15.

CODE:

	OP1	OP2	BINARY	HEXA
LDR	p	R3	0 0 0 1 | 0 | R3 | 1 0 1 1 | 1 1 0 0	10BC
LDR	p	B0.X3	0 0 0 1 | 1 | X3 | 1 0 1 1 | 1 1 0 0	18BC

********** EXECUTE RAM ********** 1 2 * * * *****

INSTRUCTION:
EXECUTE RAM XRM

FORMAT:

	OP1	OP2	NORM	MOD	BUS
XRM p		C8	1+L	1+M	0+N
			(L,M,N: USER DEFINED)		

FUNCTION:
EXECUTE RAM MICROPROGRAM(63-OP2.[7:4]);

DESCRIPTION:
CAUSES A MICROPROGRAM BRANCH TO THE RAM MICRO MEMORY. THE RAM LOCATION IS ONE OF THE HIGHEST 16 ADDRESSES IN THE 64 WORDS RAM. THE FUNCTION OF THE INSTRUCTION IS TOTALLY DEFINED BY THE CONTENT OF THE RAM MEMORY.

CODE:

	OP1	OP2	BINARY		HEXA
XRM p		C8	C8	1 1 1 C 0 1 1 0	00E6

******* SWITCH TO ALTERNATE INSTRUCTION SET ****** * * 3 4 5 *****

INSTRUCTION:
SWITCH TO ALTERNATE INSTRUCTION SET ALT

FORMAT:

	OP1	OP2	NORM	MOD	BUS
ALT			-	-	-

FUNCTION:
PSW.[12:13]:= 1;
CLEAR MODIFICATION;

DESCRIPTION:
THE BIT IN THE PROCESS STATUS WORD WHICH DETERMINES THE CURRENT INSTRUCTION SET IS SET.

CODE:

	OP1	OP2	BINARY	HEXA
ALT			1011 0000 1011 1100	B0BC

********** EXECUTE INSTRUCTION ******** 1 2 3 4 5 *****

INSTRUCTION:

EXECUTE	XCU
EXECUTE INDIRECT	XCUI

FORMAT:

	OP1	OP2	NORM	MOD	BUS
XCU		X3	6	8	2
XCUI	M	B0.X3	6	8	3

FUNCTION:

```
INSTR:= OP2;
IF INDIRECT THEN INSTR:= MEM[BASE+INSTR];
CLEAR MODIFICATION;
GOTO DECODE INSTRUCTION;
```

DESCRIPTION:

IF INDIRECT MARKED, THE OPERAND POINTS IN MEMORY AT THE RESULTING OPERAND. THE OPERAND IS EXECUTED AS AN INSTRUCTION PLACED AT THE LOCATION OF THE EXECUTE INSTRUCTION.

NOTICE:

THE LOCATION COUNTER IS NOT INCREMENTED AS THIS WILL BE DONE IN THE DECODED INSTRUCTION.

CODE:

	OP1	OP2	BINARY	HEXA
XCU		X3	1000 0 X3 1011 1100	808C
XCUI		B0.X3	1000 1 X3 1011 1100	888C

```
**********      MONITOR CALL      ********** 1 2 3 4 * *****

INSTRUCTION:
  MONITOR CALL                                          MON

FORMAT:
                    OP1         OP2                 NORM      MOD       BUS
  MON                           C8                  6+N       7+N       2+M
                                                    (SPECIAL:  N=1,M=0)
                                                    (INDIRECT: N=2,M=1)

FUNCTION:
  REG[7]:= LOC+1-PROG;
  LOC:=     IF OP2<64 THEN     63          % SPECIAL
       ELSE IF OP2<256 THEN    MEM[OP2]; % INDIRECT
  CLEAR MODIFICATION;

DESCRIPTION:
  JUMPS WITH RETURN ADDRESS IN R7 TO AN ABSOLUTE MEMORY LOCATION
  DEPENDING ON OP2:
        OP2<64:  JUMPS TO LOCATION 63.
    63<OP2<256: JUMPS INDIRECT VIA THE LOCATION SPECIFIED BY OP2.
  FOR CPUS WITH THE MEMORY PROTECTION OPTION, THE CPU STATE IS
  CHANGED TO SYSTEM STATE, THE CPU BOUND REGISTER IS SET TO
  #FFFF AND THE MONITOR LEVEL (BASE RELATIVE LOCATION -2) IS
  INCREMENTED.

CODE:
                    OP1         OP2     BINARY                      HEXA
  MON                           C8      C8      1 0 1 0 0 1 1 0     00A6
```

********** DECREMENT MONITOR LEVEL ***** 1 2 3 4 * *****

INSTRUCTION:
DECREMENT MONITOR LEVEL DML

FORMAT:

	OP1	OP2	NORM	MOD	BUS
DML p			7	8	3

FUNCTION:
DECREMENT MONITOR LEVEL;
CLEAR MODIFICATION;

DESCRIPTION:
THE MONITOR LEVEL (BASE RELATIVE LOCATION -2) IS DECREMENTED. IF IT BECOMES ZERO, THE CPU BOUND REGISTER IS LOADED WITH THE CONTENTS OF BASE RELATIVE LOCATION -1, AND THE CPU STATE IS CHANGED TO USER STATE.

CODE:

	OP1	OP2	BINARY	HEXA
DML p			0 1 1 1 0 0 0 0 1 0 1 1 1 1 0 0	70BC

```
*****************     MONITOR CALL     ***************** * * * * 5 *****

INSTRUCTION:
  MONITOR CALL                                              MON

FORMAT:
                    OP1       OP2                  NORM     MOD      BUS
  MON                         C8                   -        -        -

FUNCTION:
  OLDPSW:= PSW;
  PSW.[11:1]:= 1;
  IF CSP - 15 >= CST
    THEN
      BEGIN
        CLEAR MODIFICATION;

        CSP:= CSP - 15;
        MEM[CSP + 0]:= R0;
        MEM[CSP + 1]:= R1;
        MEM[CSP + 2]:= R2;
        MEM[CSP + 3]:= R3;
        MEM[CSP + 4]:= R4;
        MEM[CSP + 5]:= R5;
        MEM[CSP + 6]:= R6;
        MEM[CSP + 7]:= R7;
        MEM[CSP + 8]:= BASE;
        MEM[CSP + 9]:= MODIFY;
        MEM[CSP +10]:= PROG;
        MEM[CSP +11]:= LOC+1;
        MEM[CSP +12]:= OLDPSW;
        MEM[CSP +13]:= EXR;

        R7:= LOC+1;
        PROG:= 0;
        BASE:= 0;

        DEFINE LEVEL= P_MEM[256 + OP2.[7:6]].[(OP2.[1:2]*4 + 3):4];
        EXR.[15:4]:= EXR.[7:4];
        EXR.[ 7:4]:= LEVEL;
        BOUND:=      D_MEM[#FFE0 + LEVEL];

        LOC:=    IF OP2<64 THEN    63        % SPECIAL
            ELSE IF OP2<256 THEN   P_MEM[OP2]; % INDIRECT

      END
    ELSE STACK_ERROR;

DESCRIPTION:
  THE CURRENT CONTEXT IS PUSHED ONTO THE CONTEXT STACK. IN CASE OF
  A STACK OVERFLOW, AN ERROR INTERRUPT IS GENERATED
  THE CPU STATE IS CHANGED TO SYSTEM STATE AND THE SYSTEM LEVEL IS
  SET AS DEFINED THROUGH OP2. THE PROG AND BASE REGISTERS ARE SET
  TO ZERO AND R7 IS LOADED WITH AN ABSOLUTE RETURN LINK. R7 MAY BE
  USED FOR FETCHING CONSTANT PARAMETERS.

  JUMPS WITH RETURN ADDRESS IN THE SAVED LOC TO AN ABSOLUTE MEMORY
```

LOCATION DEPENDING ON OP2:
OP2<64: JUMPS TO LOCATION 63.
63<OP2<256: JUMPS INDIRECT VIA THE LOCATION SPECIFIED BY OP2.

CODE:

	OP1	OP2	BINARY	HEXA
MON		C8	C 8 | 1 0 1 0 | 0 1 1 0	00A6

```
***************     RETURN FROM MONITOR CALL     ***** * * * * 5 *****

INSTRUCTION:
  RETURN FROM MONITOR CALL                                    RTM

FORMAT:
                    OP1       OP2                 NORM     MOD      BUS
  RTM   p   (M)     C4

FUNCTION:
  IF CSP + 15 <= CSB
    THEN
      BEGIN
        IF NOT MODIFY.[0:1] THEN  R0:=     MEM[CSP + 0];
        IF NOT MODIFY.[1:1] THEN  R1:=     MEM[CSP + 1];
        IF NOT MODIFY.[2:1] THEN  R2:=     MEM[CSP + 2];
        IF NOT MODIFY.[3:1] THEN  R3:=     MEM[CSP + 3];
        IF NOT MODIFY.[4:1] THEN  R4:=     MEM[CSP + 4];
        IF NOT MODIFY.[5:1] THEN  R5:=     MEM[CSP + 5];
        IF NOT MODIFY.[6:1] THEN  R6:=     MEM[CSP + 6];
        IF NOT MODIFY.[7:1] THEN  R7:=     MEM[CSP + 7];
        BASE:=          MEM[CSP + 8];
        MODIFY:=        MEM[CSP + 9];
        PRDG:=          MEM[CSP +10];
        LOC:=           MEM[CSP +11] + OP1 ;
        NEWPSW:=        MEM[CSP +12];
        EXR:=           MEM[CSP +13];
        BOUND:=         MEM[#FFE0 + EXR.[7:4]];
        PRIORITY:=      EXR.[11:4];

        CSP:= CSP + 15;

        PSW:= NEWPSW;
      END
    ELSE STACK_ERROR;
    CLEAR MODIFICATION;

DESCRIPTION:
  THE INSTRUCTION RESTORES THE PREVIOUS CONTEXT, BUT THE DISPLACEMENT
  OP1 IS ADDED TO THE STORED LOCATION COUNTER, ALLOWING FOR MULTIPLE
  EXITS FROM MONITOR PROCEDURES.
  A SET BIT IN THE MODIFY REGISTER DEFINES THAT THE CORRESPONDING
  GENERAL PURPOSE REGISTER SHOULD NOT BE RESTORED FROM THE POPPED
  CONTEXT, BUT SHOULD BE KEPT. THIS FACILITATES RETURNING PARAMETERS.
  THE PRIORITY IN THE MAP MODULE IS SET AS DEFINED BY THE POPPED EXR.

CODE:
                    OP1       OP2     BINARY                        HEXA
  RTM   p           C4                |0 0 1 0| C 4 |1 0 1 1|1 1 0 1| 20BD
```

********** NO OPERATION ********** 1 2 3 4 5 *****

INSTRUCTION:
NO OPERATION NOP

FORMAT:

	OP1	OP2	NORM	MOD	BUS
NOP			3	3	1

FUNCTION:
COMMENT: LOC:= LOC+1;
COMMENT: GOTO NEXT INSTRUCTION;

DESCRIPTION:
THE INSTRUCTION HAS NO OTHER EFFECT THAN JUMPING TO THE NEXT INSTRUCTION. AS THE MODIFICATION SITUATION IS UNCHANGED THE INSTRUCTION EVEN HAS NO EFFECT AFTER A MODIFY INSTRUCTION.

REMARK:
THE INSTRUCTION IS IN FACT THE FOLLOWING ONE:

```
  JON    R0    LOC+1
```

CODE:

	OP1	OP2	BINARY	HEXA
NOP			1000 0000 1111 0000	80F0

********** CPU INTERRUPT ********** 1 2 3 4 * *****

INSTRUCTION:
CPU INTERRUPT CPU

FORMAT:

	OP1	OP2	NORM	MOD	BUS
CPU p			5	6	1

FUNCTION:
INITIATE CPU INTERRUPTS;
CLEAR MODIFICATION;

DESCRIPTION:
INITIATES A CPU INTERRUPT IN ALL CPU-S. FOR FURTHER DETAILES SEE THE INTERRUPT DESCRIPTION.

CODE:

	OP1	OP2	BINARY	HEXA
CPU p			0 1 1 0 | 1 1 1 1 | 1 0 1 1 | 1 1 1 0	6FBE

********** TRAP INSTRUCTION ********** 1 2 3 4 5 *****

INSTRUCTION:

TRAP	TRP

FORMAT:

	OP1	OP2	NORM	MOD	BUS
TRP		C4	-	-	-

FUNCTION:
CLEAR MODIFICATION;
ILLEGAL INSTRUCTION;

DESCRIPTION:
THE TRAP INSTRUCTION IS HANDLED AS AN ILLEGAL INSTRUCTION. AN INTERNAL INTERRUPT IS GENERATED WITH CAUSE=1.

CODE:

	OP1	OP2	BINARY	HEXA
TRP		C4	0000 C4 1011 1110	00BE

********** INTERRUPT ********** 1 2 3 4 * *****

INSTRUCTION:
INTERRUPT INT

FORMAT:

	OP1	OP2	NORM	MOD	BUS
INT	p	R3	*	*	*
INT	p	X3	*	*	*

FUNCTION:
INTRPT:= OP2.[7:8];
CLEAR MODIFICATION;
GOTO INTERRUPT RESPONSE;

DESCRIPTION:
USES THE RIGHT BYTE OF OP2 AS INTERRUPT, WHICH IS THE CONCATENATION OF A 2 BIT PRIORITY AND A 6 BIT DEVICE NUMBER. THE INTERRUPT IS EXECUTED, INCLUDING A PROCESS SAVE AND A PROCESS LOAD.

NOTICE:
THE INTERRUPT IS EXECUTED INDEPENDENTLY OF THE CURRENT CPU PRIORITY AND INDEPENDENTLY OF A POSSIBLE INTERRUPT DISABLING BY A BIT IN PSW.

CODE:

	OP1	OP2	BINARY	HEXA
INT	p	R3	0 0 1 0 0 R3 1 0 1 1 1 1 0 1	E0BD
INT	p	X3	0 0 1 0 1 X3 1 0 1 1 1 1 0 1	E8BD

********** TEST CPU ********** * * * 4 5 *****

INSTRUCTION:

CACHE ENABLE	CAE
CACHE DISABLE	CAD
CACHE CLEAR	CAC
TEST CPU	TST
READ CACHE STATUS REGISTER	RCR
TERMINATE TEST MODE	TTM

FORMAT:

		OP1	OP2	NORM	MOD	BUS
CAE	p			*	*	*
CAD	p			*	*	*
CAC	p			*	*	*
TST	p			*	*	*
RCR	p			*	*	*
TTM	p			*	*	*

FUNCTION:

```
CASE INSTR OF
  CAE: ENABLE THE CACHE FUNCTION;
  CAD: DISABLE THE CACHE FUNCTION;
  TST: PERFORM A FIRMWARE TEST OF THE CPU;
  RCR: READ THE CACHE STATUS REGISTER;
  TTM: TERMINATE THE TEST MODE;
END;
CLEAR MODIFICATION;
```

DESCRIPTION:

CODE:

		OP1	OP2	BINARY	HEXA
CAE	p			1000 0000 1011 1110	80BE
CAD	p			1000 0001 1011 1110	81BE
CAC	p			1000 0010 1011 1110	82BE
TST	p			1000 0011 1011 1110	83BE
RCR	p			1000 0100 1011 1110	84BE
TTM	p			1000 0101 1011 1110	85BE

```
**********************      CALL      ********************** * * * * 5 *****

INSTRUCTION:
  CALL                                                          CLL

FORMAT:
                    OP1         OP2                   NORM      MOD       BUS
  CLL   pp           M C4          P16                 -         -         -

FUNCTION:
  IF CSP - 15 >= CST
    THEN
      BEGIN
        DEFINE VIEW= OP1 + MODIFY;
        CLEAR MODIFICATION;
        P63:= TTC DTTR[ CPU ]][63];
        CSP:= CSP - 15;
        MEM[CSP + 0]:= R0;
        MEM[CSP + 1]:= R1;
        MEM[CSP + 2]:= R2;
        MEM[CSP + 3]:= R3;
        MEM[CSP + 4]:= R4;
        MEM[CSP + 5]:= R5;
        MEM[CSP + 6]:= R6;
        MEM[CSP + 7]:= R7;
        MEM[CSP + 8]:= BASE;
        MEM[CSP + 9]:= MODIFY;
        MEM[CSP +10]:= PROG;
        MEM[CSP +11]:= LOC+2;
        MEM[CSP +12]:= PSW;
        MEM[CSP +13]:= EXR;
        MEM[CSP+14]:= P63;

        LOC:= OP2 + PROG;

        EXR.[3:4]:= VIEW.[3:4];   "CHANGE VIEW"
        PTTR[CPU]:= TRM[VIEW].P_TTR;
        DTTR[CPU]:= TRM[VIEW].D_TTR;
        TTC DTTR[CPU]][63]:= P63;
        DISABLE INTERRUPTS;

        BASE:= 0;
        PROG:= 0;

      END
    ELSE STACK_ERROR;

DESCRIPTION:
  THE CURRENT CONTEXT IS PUSHED ONTO THE CONTEXT STACK. IN CASE OF
  A STACK OVERFLOW, AN ERROR INTERRUPT IS GENERATED

  OP1 DEFINES A VIEW AND THE CORRESPONDING PHYSICAL PROGRAM AND
  DATA TRANSLATION TABLE REGISTERS ARE DETERMINED.

  THE PHYSICAL ADDRESS OF THE CURRENT LOGICAL DATA PAGE 63 IS READ.

  THE PROGRAM AND DATA TRANSLATION TABLE REGISTERS ARE LOADED BY
```

THOSE VALUES DEFINED BY THE VIEW, AND THE FORMER LOGICAL DATA PAGE 63 IS MAPPED IN AS THE NEW LOGICAL DATA PAGE 63.

INTERRUPTS ARE DISABLED AND THE BASE AND PROG ARE RESET TO ZERO. A JUMP IS MADE TO THE PROGRAM LOCATION DEFINED BY OP2.

CODE:

	OP1	OP2	BINARY	HEXA
CLL pp	C4	P16	0111 C4 1011 1110 / P16	70BC

NOTICE:
THE CALL INSTRUCTION IS A TWO WORD INSTRUCTION.

```
*****************      RETURN      ******************** * * * * 5 *****

INSTRUCTION:
  RETURN                                                          RET

FORMAT:
                    OP1       OP2                 NORM     MOD      BUS
  RET pp  (M)       C4                            -        -        -

FUNCTION:
  IF CSP + 15 <= CSB
    THEN
      BEGIN
        IF NOT MODIFY.[0:1] THEN    R0:=  MEM[CSP + 0];
        IF NOT MODIFY.[1:1] THEN    R1:=  MEM[CSP + 1];
        IF NOT MODIFY.[2:1] THEN    R2:=  MEM[CSP + 2];
        IF NOT MODIFY.[3:1] THEN    R3:=  MEM[CSP + 3];
        IF NOT MODIFY.[4:1] THEN    R4:=  MEM[CSP + 4];
        IF NOT MODIFY.[5:1] THEN    R5:=  MEM[CSP + 5];
        IF NOT MODIFY.[6:1] THEN    R6:=  MEM[CSP + 6];
        IF NOT MODIFY.[7:1] THEN    R7:=  MEM[CSP + 7];
        BASE:=          MEM[CSP + 8];
        MODIFY:=        MEM[CSP + 9];
        PROG:=          MEM[CSP +10];
        LOC:=           MEM[CSP +11] + OP1 ;
        NEWPSW:=        MEM[CSP +12];
        EXR:=           MEM[CSP +13];
        P63:=           MEM[CSP +14];
        PRIORITY:=      EXR.[11:4];

        DEFINE VIEW =   EXR.[3:4];

        CSP:= CSP + 15;

        PTTR[CPU]:= TRM[VIEW].P_TTR;
        DTTR[CPU]:= TRM[VIEW].D_TTR;
        TTE DTTR[CPU]][63]:= P63;
        BOUND:=     MEM[#FFE0 + EXR.[7:4]];
        PSW:= NEWPSW;
      END
    ELSE STACK_ERROR;


DESCRIPTION:
  THE INSTRUCTION RESTORES THE PREVIOUS CONTEXT, BUT THE
  DISPLACEMENT OP1 IS ADDED TO THE STORED LOCATION COUNTER,
  ALLOWING FOR MULTIPLE EXITS FROM CALLS.
  A SET BIT IN THE MODIFY REGISTER DEFINES THAT THE CORRESPONDING
  GENERAL PURPOSE REGISTER SHOULD NOT BE RESTORED FROM THE POPPED
  CONTEXT, BUT SHOULD BE KEPT. THIS FACILITATES RETURNING
  PARAMETERS.

  THE TOP CONTEXT IS POPPED FROM THE CONTEXT STACK. IN CASE OF A
  STACK UNDERFLOW, AN ERROR INTERRUPT IS GENERATED.

  THE PROGRAM AND DATA TRANSLATION TABLE REGISTERS TO BE RELOADED
  ARE DETERMINED FROM THE POPPED VIEW.
```

THE PHYSICAL PAGE ADDRESS OF THE CURRENT LOGICAL DATA PAGE 63 IS READ.

THE PROGRAM AND DATA TRANSLATION TABLE REGISTERS ARE RELOADED AND THE FORMER LOGICAL DATA PAGE 63 IS MAPPED IN AS THE NEW LOGICAL DATA PAGE 63.

THE PRIORITY OF THE CPU IS UPDATED IN THE MAP MODULE.

PROCESSING CONTINUES IN THE POPPED CONTEXT.

CODE:

	OP1	OP2	BINARY	HEXA
RET pp	C4		1010 C4 1011 1101	A0BD

********** NOTIFY CPU ********** * * * * 5 *****

INSTRUCTION:
  NOTIFY CPU CPU

FORMAT:

	OP1	OP2	NORM	MOD	BUS
CPU pp	R3		-	-	-
CPU pp	M B0.X3		-	-	-

FUNCTION:
```
INTERRUPT_VECTOR:= OP1;
NOTIFY( INTERRUPT_VECTOR );
CLEAR MODIFICATION;
```

DESCRIPTION:
  A NOTIFICATION USING OP1 AS THE INTERRUPT VECTOR IS ISSUED.

CODE:

	OP1	OP2	BINARY	HEXA
CPU pp	R3		0 1 1 0 0 R 3 1 0 1 1 1 1 1 0	60BE
CPU pp	B0.X3		0 1 1 0 1 X 3 1 0 1 1 1 1 1 0	68BE

********** LOAD BASE ********** * * * * 5 *****

INSTRUCTION:
LOAD BASE LDB

FORMAT:

	OP1	OP2	NORM	MOD	BUS
LDB	R3		-	-	-
LDB	M 80.X3		-	-	-

FUNCTION:
BASE:= OP1;
CLEAR MODIFICATION;

DESCRIPTION:
THE BASE REGISTER IS LOADED WITH THE CONTENTS OF OP1.
THIS INSTRUCTION WILL THUS INFLUENCE THE ADDRESSING OF ALL
SUBSEQUENT INSTRUCTIONS.

CODE:

	OP1	OP2	BINARY	HEXA
LDB	R3		0 0 0 0 0 R3 1 0 1 1 1 1 0 0	00BC
LDB	80.X3		0 0 0 0 1 X3 1 0 1 1 1 1 0 0	08BC

********** SAVE BASE ********** * * * * 5 *****

INSTRUCTION:

SAVE BASE SVB

FORMAT:

	OP1	OP2	NORM	MOD	BUS
SVB		R3	-	-	-
SVB		M 80.X3	-	-	-

FUNCTION:

OP2:= BASE;
CLEAR MODIFICATION;

DESCRIPTION:

THE CONTENTS OF THE BASE REGISTER ARE COPIED TO OP2.

CODE:

	OP1	OP2	BINARY	HEXA
SVB		R3	0 1 0 0 0 | R 3 | 1 0 1 1 | 1 1 0 0	40BC
SVB		80.X3	0 1 0 0 1 | X 3 | 1 0 1 1 | 1 1 0 0	48BC

********** LOAD TRANSLATION TABLES **** * * * * 5 *****

INSTRUCTION:
LOAD TRANSLATION TABLES LTT

FORMAT:

	OP1	OP2	NORM	MOD	BUS
LTT pp	R3		-	-	-
LTT pp	M 80.X3		-	-	-

FUNCTION:
```
FOR I:=0 STEP 1 UNTIL 63 DO
  TTC TRM[0].P_TTR][I]:= MEM[BASE+OP1+I];
FOR I:=0 STEP 1 UNTIL 63 DO
  TTC TRM[0].D_TTR][I]:= MEM[BASE+OP1+64+I];
```

DESCRIPTION:
THIS INSTRUCTION LOADS THE TWO TRANSLATION TABLES WHICH DEFINES THE MAPPING FROM LOGICAL PROGRAM AND DATA SPACE INTO PHYSICAL MAIN MEMORY PAGE FRAMES FOR THIS CPU.
THE CONTENTS OF THE PROGRAM TRANSLATION TABLE ARE TAKEN FROM THE 64 LOCATIONS OP1..OP1+63.
THE CONTENTS OF THE DATA TRANSLATION TABLE ARE TAKEN FROM THE 64 LOCATIONS OP1+64..OP1+127.
THE ADDRESS OF THE TRANSLATION TABLES IN THE MAP MODULE ARE THOSE FOR VIEW ZERO AND CAN BE DERIVED FROM THE TRANSLATION REGISTER MAP.

CODE:

	OP1	OP2	BINARY	HEXA
LTT pp	R3		1 1 1 0 | 0 | R3 | 1 0 1 1 | 1 0 1	E0B0
LTT pp	M 80.X3		1 1 1 0 | 1 | X3 | 1 0 1 1 | 1 1 0 1	E8B0

********** CHANGE STACK POINTER ******** * * * * 5 *****

INSTRUCTION:

CHANGE STACK POINTER CSP

FORMAT:

	OP1	OP2	NORM	MOD	BUS
CSP	M BO.X3	R3	-	-	-

FUNCTION:

```
;COMMENT:
;  OP1 IS THE FIRST WORD OF A STACK CONTROL BLOCK
;  WITH THE FOLLOWING LAYOUT:
;       +0 : CURRENT STACK POINTER (CUR)
;       +1 : STACK MIN POINTER     (MIN)
;       +2 : STACK MAX POINTER     (MAX)
;  IN ORDER TO BE A VALID STACK CONTROL BLOCK THE
;  FOLLOWING RELATIONS MUST HOLD - ASSUMING CUR, MIN, MAX
;  ARE UNSIGNED INTEGERS:
;            MIN <= CUR <= MAX

DEFINE  NEW:=OP1+OP2;
IF  MIN<=NEW<=MAX
  THEN
    BEGIN
        OP1:=NEW;
        OP2:=NEW;
        LOC:=LOC+2;
    END
  ELSE  LOC:=LOC+1;
CLEAR MODIFY;
```

DESCRIPTION:

OP1 IS A STACK CONTROL BLOCK. PRIOR TO EXECUTION OF CSP, OP2 DEFINES A CHANGE TO THE CURRENT STACK POINTER. IF THE STACK POINTER CAN NOT BE CHANGED AS REQUESTED EXECUTION IS CONTINUED INTO THE NEXT INSTRUCTION OTHERWISE THE STACK POINTER IS UPDATED, ITS NEW VALUE IS RETURNED IN OP2 AND A SKIP IS PERFORMED.

CODE:

	OP1	OP2	BINARY	HEXA
CSP	X3	R3	1 X3 0 R3 0 0 0 1 0 0 0 0	8010

```
****************     INDEX     ********************** * * * * 5 *****

INSTRUCTION:
  INDEX                                                    INX

FORMAT:
                    OP1       OP2                 NORM     MOD      BUS
  INX           M 80.X3         R3                  -        -        -

FUNCTION:
  ;COMMENT:
  ;  OP1 IS THE FIRST WORD OF AN ARRAY CONTROL BLOCK
  ;  WITH THE FOLLOWING LAYOUT:
  ;       +0 : BASE OF ARRAY           (BSA)
  ;       +1 : MINIMUM INDEX           (MIN)
  ;       +2 : NUMBER OF ELEMENTS      (RANGE)
  ;       +3 : SIZE OF ELEMENTS        (SIZE)

  DEFINE  INDEX:=OP2;
  IF  MIN<=INDEX<MIN+RANGE
    THEN
      BEGIN
          OP2:=OP1.BSA+OP1.SIZE*(INDEX-OP1.MIN);
          LOC:=LOC+2;
      END
    ELSE  LOC:=LOC+1;
  CLEAR MODIFY;

DESCRIPTION:
  OP1 IS AN ARRAY CONTROL BLOCK AND OP2 IS AN INDEX FOR THE
  ARRAY CONTROLLED BY OP1.
  IF OP2 FULFILS A RANGE CHECK, THEN OP2 IS CONVERTED TO THE
  ADDRESS OF THE ARRAY ELEMENT AND A SKIP IS PERFORMED,
  OTHERWISE THE EXECUTION CONTINUES WITH THE NEXT INSTRUCTION.

CODE:
                  OP1       OP2     BINARY                          HEXA
  INX               X3        R3    |1|  X3 |1|  R3 |0 0 0 1|0 0 0 0|  8810
```

****
********** FLOATING POINT INSTRUCTIONS **********
****

The floating point instructions available on a CR8C are based on the IEEE proposal for binary floating point arithmetic, Draft 8.0, of IEEE task p754.

The floating pcint instructions cover single precision and double precision:

	sign	exponent	mantissa
sirgle precision	1 bit	8 bits	23 bits
dcuble precision	1 bit	11 bits	52 bits

The floating point instructions are subdivided into the following main groups:

- arithmetic instructions
  - addition (FADD)
  - subtraction (FSUB)
  - multiplication (FMUL)
  - civision (FDIV)
  - remainder (FREM)
- mathematical instructions
  - squareroot (SQRT)
  - tangens (TAN)
  - arcus tangens (ATAN)
  - exponentiation (EXP2)
  - logarithm (LOG2)
- convert instructions
  - convert floating (single <-> double) (FCNV)
  - convert floating to integer (FINT)
  - convert integer to floating (INTF)
  - convert floating to decimal (FDEC)
  - convert decimal to floating (DECF)
- compare instructions
  - skip (pair) if less than (FSLT[P])
  - skip (pair) if acual to (FSEQ[P])
  - skip (pair) if greater or equal (FSGE[P])
  - skip (pair) if unordered (FSUO[P])

During the execution of floating point instructions, the following four classes of floating point values are distinguished. These operand classes are:

+/- 0	the operand contains a zero value.
+/- infinity	the operand represents infinity
W	the operand does represent a floating point number, which is neither zero nor infinity.
NaN	the operand does not represent any floating point number (Not a Number)

In the description of dyadic floating point operations, the following operand class matrix is used:

		OP2			
		+/- 0	W	+/-infinity	NaN
OP1	+/-0	a	b	c	OP2
	W	d	e	f	OP2
	+/-inf	g	h	i	OP2
	NaN	OP1	OP1	OP1	OP1

When performing arithmetic operations, the corresponding bits (Z, S, V and C) in the PSW are updated.

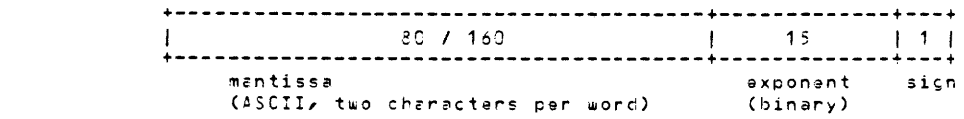
Bit V is set every time an exception occurrs, except in case of inexact result. When the V bit is set, the Z and S bits may be interpreted like this:

Z	S		Condition
0	0	=	Divide by zero
0	1	=	Invalid operation
1	0	=	Oveflow
1	1	=	Underflow

Inexact result is signalled by setting bit C.

When decimal numbers are used in conversions to/from binary floating point representation, an extended version of the standard decimal format is used. This extended decimal representation

contains a sign (1 bit), a binary exponent (15 bits including sign) and a series of bytes representing the mantissa (10 bytes for single format and 20 bytes for double format):



The mantissa holds most significant byte on lowest address, and the word, holding exponent and sign is placed after the mantissa (in the word succeeding the least significant byte of the mantissa).

********** ADD FLOATING **********

INSTRUCTION:
ADD [LONG] FLOATING FADD[L]

FORMAT:

	OP1	OP2
FADD	X3	X3
FADDL	X3	X3

FUNCTION:
OP2 := OP2 + OP1

DESCRIPTION:

Case a: result is + zero, except when both operands are - zero, then result must be - zero too.

Case c,f: result is equal to OP2.

Case g,h: result is equal to OP1.

Case b,d,e: First the binary points are aligned, this is done by shifting the significand of the operand with the smaller exponent right while incrementing its exponent. If the mantissa of the shifted operand becomes 0, the other operand is returned as result without further calculations.

The significands are added using signed magnitude addition instead of complement addition.

If overflow then the result is right-shifted one bit and the exponent is incremented.

Case i: result is + infinity if both operands are + infinity, result is - infinity if both operands are - infinity. Otherwise signal Invalid operation.

CODE:

	OP1	OP2	BINARY	HEXA
FADD	X3	X3	0 X3 0 X3 0000 0001	C001
FADDL	X3	X3	0 X3 1 X3 0000 0001	C801

********** SUBTRACT FLOATING **********

INSTRUCTION:
SUBTRACT [LONG] FLOATING FSUB[L]

FORMAT:

	OP1	OP2
FSUB	X3	X3
FSUBL	X3	X3

FUNCTION:
OP2 := OP2 - OP1

DESCRIPTION:

Subtraction is handled like addition (i.e. OP2-OP1 is treated as OP2+(-OP1)) with the following extensions:

Case b,d,e: If all the significand bits are zero:
Sign is +. The exponent is set to 0 (i.e. the result is true zero).

otherwise (if not all of the significand bits are zero) Normalize the result.

Check underflow, round and check invalid operation and overflow.

CODE:

	OP1	OP2	BINARY	HEXA
FSUB	X3	X3	0 X3 0 X3 0000 0010	CC02
FSUBL	X3	X3	0 X3 1 X3 0000 0010	0802

********** MULTIPLY FLOATING **********

INSTRUCTION:
MULTIPLY [LONG] FLOATING FMUL[L]

FORMAT:

	OP1	OP2
FMUL	X3	X3
FMULL	X3	X3

FUNCTION:
OP2 := OP2 * OP1

DESCRIPTION:

Case a,b,d: result = C with sign.

Case c,g: Signal invalid operation.

Case e: Sign of result is sign of OP1 XOR'ed with sign of OP2. Result exponent is the sum of exponent OP1 and exponent OP2.

The Significands are multiplied.

If overflow then right-shift the significand one bit and increment the exponent.

Check underflow, round and check invalid result and overflow.

Case f,h,i: If either operand is a denormalized number proceed as in case c, otherwise result = infinity with sign equal to sign of OP1 XOR'ed with sign of OP2.

CODE:

	OP1	OP2	BINARY	HEXA
FMUL	X3	X3	0 X3 0 X3 0000 0011	0003
FMULL	X3	X3	0 X3 1 X3 0000 0011	0803

********** DIVIDE FLOATING **********

INSTRUCTION:
DIVIDE [LONG] FLOATING FDIV[L]

FORMAT:

	OP1	OP2
FDIV	X3	X3
FDIVL	X3	X3

FUNCTION:
OP2 := OP2 / OP1

DESCRIPTION:

Case a,i: Signal invalid operation.

Case d,g,h: result = 0 with sign,
exception: If, in b, OP2 is a denormalized number proceed as in a.

Case b: result = infinity with sign, signal divide-by-zero,
exception: If OP1 is a denormalized number proceed as in a.

Case e: If OP1 is unnormalized proceed as in a.

Otherwise
Sign of result is sign of OP1 XOR'ed with sign of OP2.
Result exponent is exponent op2 minus exponent op1.

Divide the significands.

If N = 0, then shift the significand one bit left and decrement the exponent.

Case c,f: result = infinity with sign.

CODE:

	OP1	OP2	BINARY	HEXA
FDIV	X3	X3	0 | X3 | 0 | X3 | 00C0 | 01C0	CC04
FDIVL	X3	X3	0 | X3 | 1 | X3 | 0000 | 01C0	C804

********** REMAINDER FLOATING **********

INSTRUCTION:
REMAINDER [LONG] FLOATING FREM[L]

FORMAT:

	OP1	OP2
FREM	X3	X3
FREML	X3	X3

FUNCTION:
OP2 := OP2 REM OP1

DESCRIPTION:

Case a,b,c,f,i: Signal invalid operation.

Case d,g: If OP1 is denormalized proceed as in a otherwise result = OP2.

Case e: If OP2 is denormalized, proceed as in a, otherwise

Divide OP2 by OP1 using the normal divison routine, calculate remainder.

Normalize result.

Check underflow, round and check Invalid result and overflow.

Case h: result = OP2.

CODE:

	OP1	OP2	BINARY	HEXA
FREM	X3	X3	0 X3 0 X3 0000 01C1	0005
FREML	X3	X3	0 X3 1 X3 0000 01C1	0805

********** SQUARE ROOT **********

INSTRUCTION:
  SQUARE ROOT [LONG] FLOATING SQRT[L]

FORMAT:

	OP1	OP2
SQRT	X3	X3
SQRTL	X3	X3

FUNCTION:
  OP2 := SQRT(OP1)

DESCRIPTION:

This function calculates the squareroot of operand_1(OP1).

Case OP1 = 0: Result = OP1.
Squareroot(-0) = -0

Case OP1 < 0: Signal invalid operation.

Case 0 < OP1 < +inf: Result is squareroot(OP1).

Case OP1 = +inf: Signal invalid operation.

CODE:

	OP1	OP2	BINARY	HEXA
SQRT	X3	X3	0 X3 0 X3 0000 0110	CC06
SQRTL	X3	X3	0 X3 1 X3 0000 0110	0806

********** TANGENS **********

INSTRUCTION:

TANGENS [LONG] FLOATING TAN[L]

FORMAT:

	OP1	OP2
TAN	X3	X3
TANL	X3	X3

FUNCTION:

OP2 := TAN(OP1)

DESCRIPTION:

This function calculates the tangent to op_1 (OP1).

Case OP1 = 0: Result is OP1.

Case OP1 < (pi/4): Result is TAN(OP1).

Case OP1 >= (pi/4): Signal invalid operation.

Case OP1 < 0: Signal invalid operation.

CODE:

	OP1	OP2	BINARY	HEXA
TAN	X3	X3	0 X3 C X3 0000 0111	CC07
TANL	X3	X3	0 X3 1 X3 0000 0111	C807

ARCUS TANGENS

INSTRUCTION:

ARCUS TANGENS [LONG] FLOATING ATAN[L]

FORMAT:

	OP1	OP2
ATAN	X3	X3
ATANL	X3	X3

FUNCTION:

OP2 := ARCUS_TANGENS(OP1)

DESCRIPTION:

This function gives Arcus Tangent of OP_1 (OP1).

Case OP1 = 0: Result is OP1.

Case 0 < OP1 < 1: Result is ARCTAN(OP1).

Case OP1 >= 1: Signal invalid operation.

Case OP1 < 0: Signal invalid operation.

CODE:

	OP1	OP2	BINARY	HEXA
ATAN	X3	X3	1 X3 0 X3 0001 0001	8011
ATANL	X3	X3	1 X3 1 X3 0001 0001	8911

********** EXPONENTIATION **********

INSTRUCTION:

EXPONENTIATION [LONG] FLOATING EXP2[L]

FORMAT:

	OP1	OP2
EXP2	X3	X3
EXP2L	X3	X3

FUNCTION:

OP2 := (2**OP1)-1

DESCRIPTION:

This function calculates (2^OP1) - 1. With this function and OP2*LOG2(OP1) it is possible to raise OP1 to any power the user may wish.

Case OP1 < 0: Signal invalid operation.

Case 0 <= OP1 <= 0.5: Result is (2^OP1) - 1.

Case OP1 > 0.5: Signal invalid operation.

CODE:

	OP1	OP2	BINARY	HEXA
EXP2	X3	X3	1 X3 0 X3 0010 01C1	8C25
EXP2L	X3	X3	1 X3 1 X3 0010 01C1	8825

********** LOGARITHM **********

INSTRUCTION:
  LOGARITHM [LONG] FLOATING LOG2[L]

FORMAT:

	OP1	OP2
LOG2	X3	X3
LOG2L	X3	X3

FUNCTION:
  OP2 := OP2 * LOG2(OP1)

DESCRIPTION:

  This function calculates OP2 * LOG2(OP1), with it and (2^OP1)-1 it is possible in an efficient way to calculate OP2^OP1 and LOGr(OP1).

  Case OP2 = +-inf: Signal invalid operation.

  Case OP1 <= 0: Signal invalid operation.

  Case OP1 = +inf: Signal invalid operation.

  Case 0 < OP1 < +inf and -inf < OP2 < +inf:
    w:= LOG2(OP1)
    result:= W * OP2

CODE:

	OP1	OP2	BINARY	HEXA
LOG2	X3	X3	0 | X3 | 0 | X3 | 0010 | 01C0	CC24
LOG2L	X3	X3	0 | X3 | 1 | X3 | 0010 | 01C0	C824

```
****************     CONVERT FLOATING     ****************************

INSTRUCTION:
  CONVERT [LONG] FLOATING                                        FCNV[L]

FORMAT:
                    OP1       OP2
  FCNV              X3        X3
  FCNVL             X3        X3

FUNCTION:
  CASE INSTRUCTION OF
    CNV:  OP2 := SHORT_FLOATING(OP1)  % OP1 IS LONG FLOATING
    CNVL: OP2 := LONG_FLOATING(OP1)   % OP1 IS SHORT FLOATING
  END;

DESCRIPTION:

  Conversion from double format to single format:


  If CP1 is 0:       result = 0.

  If CP1 is a finite number:
     If CP1 is denormalized, signal invalid operation, otherwise

     If  the exponents absolute value is to large, signal overflow,
     otherwise

     Adjust exponent. (account for the different bias)

     Round fractional part to fit destination format.

  If CP1 is ± infinity:   result = infinity with sign.


  Conversion from single format to double format.



  If CP1 is 0:     result = 0.

  If CP1 is any finite number:
     If CP1 is denormalized, signal invalid operation,
     otherwise

     Adjust exponent, (account for the different biasing)

     Pad fractional part with zeroes to fit destination format.

  If CP1 is ± infinity:
```

result = infinity with sign.

CODE:

	OP1	OP2	BINARY	HEXA
FCNV	X3	X3	0 X3 0 X3 0010 0110	0026
FCNVL	X3	X3	0 X3 1 X3 0010 0110	0826

********** CONVERT FLOATING TO INTEGER **********

INSTRUCTION:
CONVERT [LONG] FLOATING TO INTEGER FINT[L]

FORMAT:

	OP1	OP2
FINT	X3	X3
FINTL	X3	X3

FUNCTION:

```
CASE INSTRUCTION OF
  FINT:  OP2 := INTEGER(OP1)  % OP1 IS SHORT FLOATING
  FINTL: OP2 := INTEGER(OP1)  % OP1 IS LONG FLOATING
ENC;
```

DESCRIPTION:

This instruction converts a fp number in any binary fp format to a long integer. The absolute value of the fp number must lie in the range $0 \leq OP1 < 2^{**}31 - 1$.

If OP1 is zero: result = zero with sign.

If OP1 is any finite number: If OP1 is denormalized signal invalid operation,
otherwise

If the exponent is greater than 31+exponent_bias signal invalid operation.
If the exponent is less than exponent_bias signal invalid operation.
If none of these two conditions are satisfied do

Right shift the significand until it represents the integer part of the fp number.

Discard the exponent and map the significand over in the destination operand, possibly after rounding.

If the fp number was negative then complement the integer.

If OP1 is ± infinity: Signal invalid operation.

CODE:

	OP1	OP2	BINARY	HEXA
FINT	X3	X3	0 X3 0 X3 0100 0000	0040
FINTL	X3	X3	0 X3 1 X3 0100 0000	C840

********** CONVERT INTEGER TO FLOATING **********

INSTRUCTION:
  CONVERT INTEGER TO [LONG] FLOATING INTF[L]

FORMAT:

	OP1	OP2
INTF	X3	X3
INTFL	X3	X3

```
FUNCTION:
  CASE INSTRUCTION OF
    INTF:  OP2 := FLOATING(OP1)       % OP1 IS INTEGER
    INTFL: OP2 := LONG_FLOATING(OP1) % OP1 IS INTEGER
  END;
```

DESCRIPTION:

This instruction converts an integer to a binary fp number in the specified fp format (single/double).

If CP1 is zero: Set result to normal zero.

If CP1 is any number except zero: Set exponent to 3C.
Save sign of integer. If negative complement the integer.

While bit 31 = 0 left shift the integer and decrement the exponent.

Move the integer to the mantissa field of the fp number, bit 30 should be at A (the most significand bit).

Round result to the specified precision.

CODE:

	OP1	OP2	BINARY	HEXA
INTF	X3	X3	0 X3 C X3 0100 0001	0041
INTFL	X3	X3	0 X3 1 X3 0100 0001	0841

********** CONVERT FLOATING TO DECIMAL **********

INSTRUCTION:
  CONVERT [LONG] FLOATING TO DECIMAL FDEC[L]

FORMAT:

	OP1	OP2
FDEC	X3	X3
FDECL	X3	X3

```
FUNCTION:
  CASE INSTRUCTION OF
    FDEC:  OP2 := DECIMAL(OP1)  % OP1 IS SHORT FLOATING
    FDECL: OP2 := DECIMAL(OP1)  % OP1 IS LONG FLOATING
  END;
```

DESCRIPTION:

This procedure converts a binary floating-point number to a decimal floating-point number (radix-2 -> radix-10). In the following K denotes the number of significand digits in the decimal representation. (K=10 for single and K=20 for double)

1. If OP1 = ± infirity or NaN padd the mantissa field with '+', '-' or '*' respectively.

   If OP1 is zero, return ± zero
   otherwise...

2. X:= |OP1|, the sign is saved.

3. If X is normalized compute:
   U:= log(X)
   otherwise
   U:= log(smallest normalized number).

4. V:= U + 1 - K rounded towards zero.

5. W:= X / (10**V) rounded normaly.

6. Adjust exponent:

   If W ≥ (10**K) + 1 then V:= V + 1; goto 5.

   If W = 10**K then V:= V + 1; W:= W / 10; goto 7.

   If W ≤ (10**(K-1))-1 and X was normalized in step 3 then V:= V-1; goto 5.

7. Exponent part for the decimal fp number is V.

8. The mantissa part of the decimal fp number is created from

W with sign of X from step 2.

CODE:

	OP1	OP2	BINARY	HEXA
FDEC	X3	X3	0 X3 0 X3 0100 0010	0042
FDECL	X3	X3	0 X3 1 X3 0100 0010	0842

********** CONVERT DECIMAL TO FLOATING **********

INSTRUCTION:
  CONVERT DECIMAL TO [LONG] FLOATING DECF[L]

FORMAT:

	OP1	OP2
DECF	X3	X3
DECFL	X3	X3

FUNCTION:
```
  CASE INSTRUCTION OF
    DECF:  OP2 := SHORT_FLOATING(OP1) % OP1 IS DECIMAL
    DECFL: OP2 := LONG_FLOATING(OP1)  % OP1 IS DECIMAL
  END
```

DESCRIPTION:

This instruction converts a decimal floating-point number.

At entry we have the following data:

E - a signed integer representing the exponent of the decimal fp number.

I - a decimal string in the form DDDDDDDD---DDD representing the mantissa of the decimal fp number.

S - a bit showing the sign of the decimal fp number.

The decimal number may be written as:

$$X = (-1)^{**}S * I*10^{**}(-K) * 10^{**}E$$

where K is the number of digits to the rigt of the decimal point in the decimal fp number. (9 for single and 19 for double fp format)

The conversion is done in this way:

1. Convert I to a binary integer.
2. If I wasn't numeric return a NaN and signal invalid operation.
3. Compute the result:

   result:= I * 10**(E-K)

All calculations done in the conversion must be made in 32/64 bit precision.

CODE:

	OP1	OP2	BINARY	HEXA
DECF	X3	X3	0 X3 C X3 0100 0011	C043
DECFL	X3	X3	0 X3 1 X3 0100 0011	C843

********** COMPARE FLOATING **********

INSTRUCTION:
COMPARE FLOATING POINT VALUES <FSKIP>[L][P]

FORMAT:

	OP1	OP2
<FSKIP>	X3	X3
<FSKIP>L	X3	X3
<FSKIP>P	X3	X3
<FSKIP>LP	X3	X3

FUNCTION:

```
IF CASE INSTR OF
  ( FSEQ[P]:  OP1 = OP2          % OP1 AND OP2 ARE SHORT FLOATING
    FSEQL[P]: OP1 = OP2          % OP1 AND OP2 ARE LONG FLOATING
    FSGE[P]:  OP1 >= OP2         % OP1 AND OP2 ARE SHORT FLOATING
    FSGEL[P]: OP1 >= OP2         % OP1 AND OP2 ARE LONG FLOATING
    FSLT[P]:  OP1 < OP2          % OP1 AND OP2 ARE SHORT FLOATING
    FSLTL[P]: OP1 < OP2          % OP1 AND OP2 ARE LONG FLOATING
    FSUO[P]:  UNORD(OP1,OP2)     % OP1 AND OP2 ARE SHORT FLOATING
    FSUOL[P]: UNORD(OP1,OP2)     % OP1 AND OP2 ARE LONG FLOATING
  )
THEN
  IF PAIR THEN LOC:= LOC+3       % SKIP 2
          ELSE LOC:= LOC+2       % SKIP 1
ELSE           LOC:= LOC+1       % NO SKIP
```

DESCRIPTION:

First of all the operands are compared and a general condition code is generated:

```
00  <=>  OP1 = OP2
01  <=>  OP1 < OP2
10  <=>  OP1 >= OP2
11  <=>  OP1 and OP2 are unordered
```

This condition code is used to update the program location at which to continue execution.

CODE:

	OP1	OP2	BINARY	HEXA
FSLT	X3	X3	0 X3 0 X3 0100 0100	0044
FSLTL	X3	X3	0 X3 1 X3 0100 0100	0844
FSEQ	X3	X3	0 X3 0 X3 0100 0101	0045
FSEQL	X3	X3	0 X3 1 X3 0100 0101	0845
FSGE	X3	X3	0 X3 0 X3 0100 0110	0046
FSGEL	X3	X3	0 X3 1 X3 0100 0110	0846
FSUO	X3	X3	0 X3 0 X3 0100 0111	0047
FSUOL	X3	X3	0 X3 1 X3 0100 0111	0847
FSLTP	X3	X3	1 X3 0 X3 0101 0000	8050

FSLTLP	X3	X3	1 X3 1 X3 0101 0000	8850
FSEQP	X3	X3	1 X3 0 X3 0101 0001	8051
FSEQLP	X3	X3	1 X3 1 X3 0101 0001	8851
FSGEP	X3	X3	1 X3 0 X3 0101 0010	8052
FSGELP	X3	X3	1 X3 1 X3 0101 0010	8852
FSLCP	X3	X3	0 X3 0 X3 0101 0011	0053
FSLCLP	X3	X3	0 X3 1 X3 0101 0011	0853

ALTERNATIVE CR80 INSTRUCTIONS

Apart from the standard instruction set, CR80 may be equipped with alternative instructions.

When executing standard instructions, the CPU may switch to an alternative instruction set by means of the standard instruction 'ALT'.

The CPU may switch back to the standard instruction set explicitly by executing an alternative 'switch back' instruction or implicitly as part of the execution of some alternative instruction.

****
********** MOVE INSTRUCTIONS **********
****

In the following is described a series of alternative move instructions, all performing an implicit switch back to standard instruction set.

********** MOVE WORDS ASCENDING **********

INSTRUCTION : MOVE WORDS ASCENDING.

FORMAT:

	OP1	OP2	OP3
MOWA	X3	X3	R3

PERFORMANCE: RESULTANT_CPU_CYCLES.

FOR OP3=0	:	5
FOR OP3<=27	:	17+4*OP3
FOR OP3>>27	:	4.2 * OP3

FUNCTION:

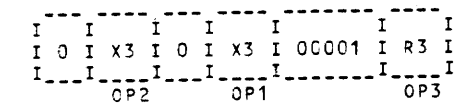
```
WHILE CP3>C DO
  BEGIN
   CP2:=CP1;
   INCR(INDX1);
   INCR(INDX2);
   DECR OP3
  END;
CLEAR MODIFICATION;
CLEAR PSW[12].
```

DESCRIPTION:

INSTRUCTION BELONGING TC THE ALTERNATIVE INSTRUCTION SET. MCVES A SEQLENCE OF 0 TO 64K WCRDS IN MEMORY WITHIN THE CURRENT PAGE, THE NUMBER OF WORDS BEING DESIGNATEC BY OP3. THE CURRENT PAGE IS REGARDED AS A CYCLIC MEMORY; CONSEQUENTLY THE SUCCESSOR OF MEMORY LOCATION #FFFF IS LOCATION 0 WITHIN THE SAME PAGE. AT EXIT OP3 WILL BE ZERO, AND INDX1,INDX2 WILL POINT AT THE MEMORY LOCATIONS SUCCEDING THE MEMORY LOCATION-PAIR LAST PROCESSED. IN THE PSW, ONLY BIT 12 IS CHANGED.

CODE:

```
MOWA    X3        X3        R3
```



********** MOVE WORDS DESCENDING **********

INSTRUCTION : MOVE WORDS DESCENDING.

FORMAT:

	OP1	OP2	OP3
MOWD	X3	X3	R3

PERFORMANCE: RESULTANT_CPU_CYCLES.

FOR OP3=0	:	5
FOR OP3<=27	:	17+4*OP3
FOR OP3>>27	:	4.2 * OP3

FUNCTION:

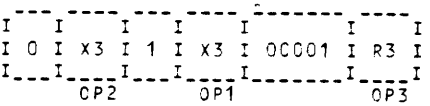
```
WHILE OP3>0 DO
  BEGIN
   OP2:=OP1;
   DECR(INDX1);
   DECR(INDX2);
   DECR OP3
  END;
CLEAR MODIFICATION;
CLEAR PSW[12];
```

DESCRIPTION:

INSTRUCTION BELONGING TO ALTERNATIVE INSTRUCTION SET. MOVES A SEQUENCE OF 0 TO 64K WORDS IN MEMORY WITHIN THE CURRENT PAGE, THE NUMBER OF WORDS BEING DESIGNATED BY OP3. THE CURRENT PAGE IS REGARDED AS A CYCLIC MEMORY; CONSEQUENTLY THE SUCCESSOR OF MEMORY LOCATION #FFFF IS LOCATION 0 WITHIN THE SAME PAGE. AT EXIT OP3 WILL BE ZERO, AND INDX1,INDX2 WILL POINT AT THE MEMORY LOCATIONS PRECEEDING THE MEMORY LOCATION-PAIR LAST PROCESSED. IN THE PSW, ONLY BIT 12 IS CHANGED.

CODE:

MOWD X3 X3 R3



********** MOVE WORDS SKEWED, ASCENDING **********

INSTRUCTION: MOVE WORDS SKEWED, ASCENDING.

FORMAT:

	OP1	OP2	OP3
MSWA	X3	X3	R33

PERFORMANCE:

	CPU CYCLES
FOR OP3=0 :	5
FOR OP3<=21 :	21 + 6*(OP3-1)
FOR OP3>>21 :	6.8 * OP3

FUNCTION:

```
W2:= LOWER BYTE OF OP3-UPPER;
WHILE OP3-LOWER > 0 DO
  BEGIN
   W1:=OP1;
   W2:= SWP( EXTR LOWER BYTE OF W2);
   W2,W1:= SLLC (W2,W1);
   OP2:=W1;
   INCR(INDX1);
   INCR(INDX2);
   DECR OP3-LOWER
  END;
OP3-UPPER:= EXTR LOWER BYTE OF W2.
CLEAR MODIFICATION;
CLEAR PSW[12];
```

DESCRIPTION:

INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE INSTRUCTION MOVES A SEQUENCE OF 0 TO 64K WORDS SKEWED WITHIN THE CURRENT PAGE, THE NUMBER OF WORDS BEING DESIGNATED BY THE REGISTER OF OP3 WITH THE LOWER NUMBER (= OP3-LOWER). THE LOWER BYTE OF THE OP3-REGISTER WITH THE HIGHER NUMBER (= OP3-UPPER) IS WRITTEN INTO THE LOWER BYTE OF THE FIRST DESTINATION LOCATION. AT EXIT INDX1,INDX2 WILL POINT AT THE MEMORY LOCATIONS SUCCEDING THE MEMORY LOCATION-PAIR LAST PROCESSED; OP3-LOWER WILL BE ZERO AND THE LOWER BYTE OF OP3-UPPER WILL CONTAIN THE UPPER BYTE OF THE SOURCE MEMORY LOCATION LAST PROCESSED. IN THE PSW, ONLY BIT 12 IS CHANGED.

CODE:

```
MSWA    X3   X3  R33
```

********** MOVE WORDS SKEWED, DESCENDING **********

INSTRUCTION: MOVE WORDS SKEWED, DESCENDING.

FORMAT:

	OP1	OP2	OP3
MSWD	X3	X3	R33

PERFORMANCE: RESULTANT_CPU_CYCLES.

FOR OP3=0	:	5
FOR OP3<=21	:	25 + 6(OP3-1)
FOR OP3>>21	:	7 * OP3

FUNCTION:

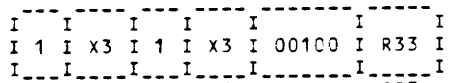
```
W2:= LOWER BYTE OF OP3-UPPER;
WHILE OP3-LOWER > 0 DO
  BEGIN
   W1:=OP1;
   W2:= EXTR LOWER BYTE OF W2;
   W2,W1:= SRLC (W2,W1);
   W2:= SWP W2;
   OP2:=W1;
   DECR(INDX1);
   DECR(INDX2);
   DECR OP3-LOWER
  END;
OP3-UPPER:= EXTR LOWER BYTE OF W2.
CLEAR MODIFICATION;
CLEAR PSW[12];
```

DESCRIPTION:

INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE INSTRUCTION MOVES A SEQUENCE OF 0 TO 64K WORDS SKEWED WITHIN THE CURRENT PAGE, THE NUMBER OF WORDS BEING DESIGNATED BY THE REGISTER OF OP3 WITH THE LOWER NUMBER (= OP3-LOWER). THE LOWER BYTE OF THE OP3-REGISTER WITH THE HIGHER NUMBER (= OP3-UPPER) IS WRITTEN INTO THE UPPER BYTE OF THE FIRST DESTINATION LOCATION. AT EXIT INDX1/INDX2 WILL POINT AT THE MEMORY LOCATIONS PRESCEDING THE MEMORY LOCATION-PAIR LAST PROCESSED; OP3-LOWER WILL BE ZERO AND THE LOWER BYTE OF OP3-UPPER WILL CONTAIN THE LOWER BYTE OF THE SOURCE MEMORY LOCATION LAST PROCESSED. IN THE PSW, ONLY BIT 12 IS CHANGED.

CODE:

```
MSWD    X3   X3   R33
```



********** MOVE BYTES ASCENDING **********

INSTRUCTION : MOVE BYTES ASCENDING.

FORMAT:	OP1	OP2	OP3
MOBA	BX3	BX3	R3

PERFORMANCE: RESULTANT_CPU_CYCLES.

FOR OP3=0	:	5
FOR OP3<=30	:	23 + 3*(OP3-1)
FOR OP3>>30	:	3.5 * OP3

FUNCTION:

```
WHILE OP3>0 DO
  BEGIN
   OP2:=OP1;
   INCR(INDX1);
   INCR(INDX2);
   DECR OP3
  END;
CLEAR MODIFICATION;
CLEAR PSW[12];
```

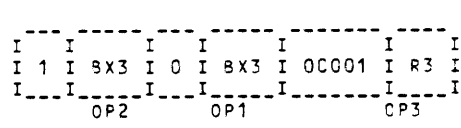
DESCRIPTION:

INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. MOVES A SEQUENCE OF 0 TO 64K BYTES IN MEMORY WITHIN THE CURRENT PAGE, THE NUMBER OF BYTES BEING DESIGNATED BY OP3. THE CURRENT PAGE IS REGARDED AS A CYCLIC MEMORY; CONSEQUENTLY THE SUCCESSOR OF MEMORY LOCATION #FFFF IS LOCATION 0 WITHIN THE SAME PAGE. AT EXIT OP3 WILL BE ZERO, AND INDX1,INDX2 WILL POINT AT THE MEMORY LOCATIONS SUCCEDING THE MEMORY LOCATION-PAIR LAST PROCESSED. IN THE PSW, ONLY BIT 12 IS CHANGED.

NOTE: BX3 MEANS A BYTE OPERAND, THE BYTE-ADDRESS OF WHICH IS CALCULATED BY ADDING 2*BASE TO THE CONTENTS OF THE REGISTER.

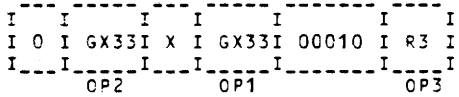
CODE:

```
MOBA    BX3   BX3   R3
```



```
****************     MOVE BYTES DESCENDING     **********************

INSTRUCTION : MOVE BYTES DESCENDING.

FORMAT:         OP1         OP2         OP3

  MOBD          BX3         BX3          R3

PERFORMANCE:                    RESULTANT_CPU_CYCLES.

  FOR OP3=0       :             5
  FOR OP3<=30     :             23 + 3*(OP3-1)
  FOR OP3>>30     :             3.5 * OP3

FUNCTION:

  WHILE OP3>0 DO
    BEGIN
     OP2:=OP1;
     DECR(INDX1);
     DECR(INDX2);
     DECR OP3
    END;
  CLEAR MODIFICATION;
  CLEAR PSW[12];

DESCRIPTION:

  INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. MOVES A
  SEQUENCE OF 0 TO 64K BYTES IN MEMORY WITHIN THE CURRENT PAGE, THE
  NUMBER OF BYTES BEING DESIGNATED BY OP3. THE CURRENT PAGE IS
  REGARDED AS A CYCLIC MEMORY; CONSEQUENTLY THE SUCCESSOR OF MEMORY
  LOCATION #FFFF IS LOCATION 0 WITHIN THE SAME PAGE. AT EXIT OP3
  WILL BE ZERO, AND INDX1,INDX2 WILL POINT AT THE MEMORY LOCATIONS
  PRECEDING THE MEMORY LOCATION-PAIR LAST PROCESSED. IN THE PSW,
  ONLY BIT 12 IS CHANGED.
  NOTE: BX3 MEANS A BYTE OPERAND, THE BYTE-ADDRESS OF WHICH IS
        CALCULATED BY ADDING 2*BASE TO THE CONTENTS OF THE
        REGISTER.

CODE:
                                  ___ _____ ___ _____ _______ ____
                                 I   I     I   I     I       I    I
  MOBD      BX3   BX3   R3       I 1 I BX3 I 1 I BX3 I 00001 I R3 I
                                 I___I_____I___I_____I_______I____I
                                       OP2       OP1           OP3
```

```
****************    GENERAL MOVE WORDS ASCENDING    ***************

INSTRUCTION : GENERAL MOVE WORDS ASCENDING.

FORMAT:         OP1         OP2          OP3

  GMCW          GX33        GX33           R3

PERFORMANCE:                  RESULTANT_CPU_CYCLES.

  FOR OP3=0       :           5
  FOR OP3<=18     :           27 + 7*(OP3-1)
  FOR OP3>>18     :           8.2 * OP3

FUNCTION:

  WHILE OP3>0 DO
    BEGIN
     OP2:=OP1;
     INCR(INDX1);
     INCR(INDX2);
     DECR OP3
    END;
  CLEAR MODIFICATION;
  CLEAR PSW[12];

DESCRIPTION:

  INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE
  INSTRUCTION MOVES A SEQUENCE OF 0 TO 64K WORDS BETWEEN ANY TWO
  PAGES IN MEMORY. THE BASE-RELATIVE SOURCE AND DESTINATION
  ADDRESSES ARE EACH CONTAINED IN REGISTER PAIRS, WITH THE
  LOW-NUMBER REGISTER CONTAINING THE PAGE, AND THE HIGH-NUMBER
  REGISTER CONTAINING THE 16 BIT ADDRESS WITHIN THE PAGE. THE
  SOURCE AND DESTINATION PAGES ARE EACH REGARDED AS CYCLIC
  MEMORIES; CONSEQUENTLY, THE SUCCESSOR OF MEMORY LOCATION #FFFF IN
  A PAGE IS LOCATION 0 WITHIN THE SAME PAGE. AT EXIT OP3 WILL BE
  ZERO, AND INDX1,INDX2 WILL POINT AT THE MEMORY LOCATIONS
  SUCCEDING THE MEMORY LOCATION-PAIR LAST PROCESSED. IN THE PSW,
  ONLY BIT 12 IS CHANGED.

CODE:

  GMCW      GX33 GX33 R3
```



```
**********************    GENERAL MOVE WORDS SKEWED    **********************

INSTRUCTION: GENERAL MOVE WORDS SKEWED.

FORMAT:          OP1          OP2          OP3

  GMSW           GX33         GX33         R33

PERFORMANCE:                      RESULTANT_CPU_CYCLES.

  FOR OP3=0        :              5
  FOR OP3<=15      :              37 + 8*(OP3-1)
  FOR OP3>>15      :              10 * OP3

FUNCTION:

  W2:= LOWER BYTE OF OP3-UPPER;
  WHILE OP3-LOWER > 0 DO
    BEGIN
     W1:=OP1;
     W2:= SWP( EXTR LOWER BYTE OF W2);
     W2,W1:= SLLC (W2,W1);
     OP2:=W1;
     INCR(INDX1);
     INCR(INDX2);
     DECR OP3-LOWER
    END;
  OP3-UPPER:= EXTR LOWER BYTE OF W2.
  CLEAR MODIFICATION;
  CLEAR PSW[12];

DESCRIPTION:

  INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE
  INSTRUCTION MOVES A SEQUENCE OF 0 TO 64K WORDS SKEWED BETWEEN ANY
  TWO PAGES IN MEMORY. THE BASE-RELATIVE SOURCE AND DESTINATION
  ADDRESSES ARE EACH CONTAINED IN REGISTER PAIRS, WITH THE
  LOW-NUMBER REGISTER CONTAINING THE PAGE, AND THE HIGH-NUMBER
  REGISTER CONTAINING THE 16 BIT ADDRESS WITHIN THE PAGE. THE
  SOURCE AND DESTINATION PAGES ARE EACH REGARDED AS CYCLIC
  MEMORIES; CONSEQUENTLY, THE SUCCESSOR OF MEMORY LOCATION #FFFF IN
  A PAGE IS LOCATION 0 WITHIN THE SAME PAGE. IN OTHER RESPECTS,
  OPERATION IS SIMILAR TO THE INSTRUCTION "MOVE WORDS SKEWED
  ASCENDING".

CODE:

  GMSW  GX33 GX33 R33
```

********** GENERAL MOVE BYTES ASCENDING **********

INSTRUCTION : GENERAL MOVE BYTES ASCENDING.

FORMAT:

	OP1	OP2	OP3
GMOB	GBX33	GBX33	R3

PERFORMANCE: RESULTANT_CPU_CYCLES.

FOR OP3=0	:	5
FOR OP3<=30	:	18 + 3*(OP3-1)
FOR OP3>>30	:	3.3 * OP3

FUNCTION:

```
WHILE OP3>0 DO
  BEGIN
   OP2:=OP1;
   INCR(INDX1);
   INCR(INDX2);
   DECR OP3
  END;
CLEAR MODIFICATION;
CLEAR PSW[12];
```

DESCRIPTION:

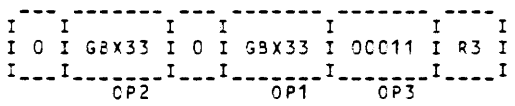
INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE INSTRUCTION MOVES A SEQUENCE OF 0-64K BYTES BETWEEN ANY TWO PAGES IN MEMORY, THE NUMBER OF BYTES BEING DESIGNATED BY OP3. THE SOURCE AND DESTINATION PAGES ARE REGARDED AS CYCLIC MEMORIES; CONSEQUENTLY THE SUCCESSOR OF MEMORY LOCATION #FFFF IN ONE OF THE PAGES IS LOCATION #0000 WITHIN THE SAME PAGE. AT EXIT OP3 WILL BE ZERO, AND INDX1,INDX2 WILL POINT AT THE MEMORY LOCATIONS SUCCEDING THE MEMORY_LOCATION_PAIR LAST PROCESSED. IN PSW, ONLY BIT 12 IS CHANGED .

NOTE: GBX33 MEANS A BYTE OPERAND, THE ADDRESS OF WHICH IS CALCULATED AS FOLLOWS:

PAGE = BIT (3:2) OF THE LOW_NUMBER REGISTER OF THE REGISTER PAIR.
BYTE ADDRESS WITHIN PAGE= 2*(BASE + CONTENTS OF THE HIGH_NUMBER REGISTER OF THE REGISTER PAIR) + BIT(0) OF THE LOW_NUMBER REGISTER OF THE REGISTER PAIR.

CODE:

```
GMOB  GBX33 GBX33  R3
```



********** GENERAL LOAD BYTE **********

INSTRUCTION: GENERAL LOAD BYTE.

FORMAT: OP1 OP2 OP3

  GLDB GBX33 R3

PERFORMANCE: RESULTANT_CPU_CYCLES.

  15

FUNCTION::

```
  OP2:=OP1;
  CLEAR MODIFICATION;
  CLEAR PSW[12];
```

DESCRIPTION:

  INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE INSTRUCTION PERFORMS A BYTE LOAD OPERATION FROM A GENERAL (19 BIT) PROCESS RELATIVE BYTE ADDRESS. THE UPPER BYTE OF THE LOADED REGISTER IS CLEARED.
  NOTE: GBX33 MEANS A BYTE OPERAND, THE ADDRESS OF WHICH IS CALCULATED AS FOLLOWS:

  SOURCE PAGE = BIT (3:2) OF THE LOW-NUMBER REGISTER OF THE REGISTER PAIR,
  BYTE ADDRESS WITHIN THE PAGE = 2*(BASE + CONTENTS OF THE HIGH NUMBER REGISTER OF THE REGISTER PAIR) + BIT (0) OF THE LOW-NUMBER

CODE:

```
                  ___  _______ ___ ____ _________________
                 I   I        I   I    I                 I
GLDB   GBX33  R3 I 0 I GBX33 I 0 I R3 I    00110000     I
                 I___I_______I___I____I_________________I
                       OP1       OP2
```

********** GENERAL STORE BYTE **********

INSTRUCTION: GENERAL STORE BYTE.

FORMAT: OP1 OP2

  GSTB R3 GBX33

PERFORMANCE: RESULTANT_CPU_CYCLES.

  13

FUNCTION::

```
OP2:=OP1;
CLEAR MODIFICATION;
CLEAR PSW[12];
```

DESCRIPTION:

INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE INSTRUCTION PERFORMS A BYTE STORE OPERATION TO A GENERAL (19 BIT) PROCESS RELATIVE BYTE ADDRESS.

NOTE: GBX33 MEANS A BYTE OPERAND, THE ADDRESS OF WHICH IS CALCULATED AS FOLLOWS:

SOURCE PAGE = BIT (3:2) OF THE LOW-NUMBER REGISTER OF THE REGISTER PAIR,
BYTE ADDRESS WITHIN THE PAGE = 2*(BASE + CONTENTS OF THE HIGH NUMBER REGISTER OF THE REGISTER PAIR) + BIT (0) OF THE LOW-NUMBER REGISTER OF THE REGISTER PAIR.

CODE:

```
                          ___  _______ ___ ____ _________________
                         I   I       I   I    I                 I
GSTB     R3   GBX33      I 0 I GBX33 I 1 I R3 I    C0110000     I
                         I___I_______I___I____I_________________I
                               OP2         OP1
```

********** GENERAL LOAD WORD **********

INSTRUCTION: GENERAL LOAD WORD.

FORMAT: OP1 OP2 OP3

  GLDW GX33 R3

PERFORMANCE: RESULTANT_CPU_CYCLES.

  13

FUNCTION::

```
  OP2:=OP1;
  CLEAR MODIFICATION;
  CLEAR PSW[12];
```

DESCRIPTION:

  INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE INSTRUCTION PERFORMS A WORD LOAD OPERATION FROM A GENERAL (18 BIT) PROCESS RELATIVE WORD ADDRESS.

  NOTE: GX33 MEANS A WORD OPERAND, THE ADDRESS OF WHICH IS CALCULATED AS FOLLOWS:

  SOURCE PAGE = BIT (3:2) OF THE LOW-NUMBER REGISTER OF THE REGISTER PAIR,
  WORD ADDRESS WITHIN THE PAGE = BASE + CONTENTS OF THE HIGH-NUMBER REGISTER OF THE REGISTER PAIR

CODE:

```
                        ___  _______ ___ ____ _________________
                       I   I        I   I    I                 I
  GLDW   GX33    R3    I 1 I GX33   I C I R3 I    C0110000     I
                       I___I_______I___I____I_________________I
                             OP1         OP2
```

********** GENERAL STORE WORD **********

INSTRUCTION: GENERAL STORE WORD.

FORMAT:

	OP1	OP2
GSTW	R3	GX33

PERFORMANCE:

RESULTANT_CPU_CYCLES.

13

FUNCTION::

```
OP2:=OP1;
CLEAR MODIFICATION;
CLEAR PSW[12];
```

DESCRIPTION:

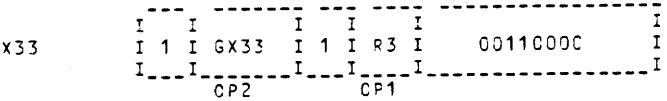
INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE INSTRUCTION PERFORMS A WORD STORE OPERATION TO A GENERAL (18 BIT) PROCESS RELATIVE WORD ADDRESS.

NOTE: GX33 MEANS A WORD OPERAND, THE ADDRESS OF WHICH IS CALCULATED AS FOLLOWS:

SOURCE PAGE = BIT (3:2) OF THE LOW-NUMBER REGISTER OF THE REGISTER PAIR,
WORD ADDRESS WITHIN THE PAGE = BASE + CONTENTS OF THE HIGH-NUMBER REGISTER OF THE REGISTER PAIR.

CODE:

GSTW R3 GX33



********** GET NEXT WORD **********

INSTRUCTION: GET NEXT WORD.

FORMAT:

	OP1	OP2
GNW	88	R3

PERFORMANCE: RESULTANT_CPU_CYCLES.

3

FUNCTION:

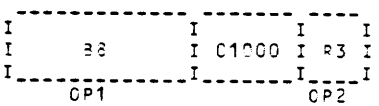
```
OP2:=MEM(BASE + OP1);
OP1:=OP1 +1;
CLEAR MODIFICATION;
CLEAR PSW[12];
```

DESCRIPTION:

INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE INSTRUCTION IS INTENDED EXCLUSIVELY FOR USE BY THE COBOL RUN-TIME INTERPRETER FOR FETCH OF THE NEXT PSEUDO-INSTRUCTION. IN THE PSW, ONLY BIT 12 IS CHANGED.

CODE:

```
GNW     88   R3
```



****
********** DECIMAL INSTRUCTIONS **********
****

In the following is described a series of alternative instructions, handling variable length decimal numbers, holding the ASCII value of one decimal digit in each byte. All instructions perform an implicit switch back to standard instruction set.

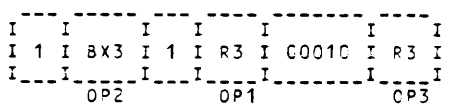
```
****************     COMPARE BYTES     ******************************

INSTRUCTION: COMPARE BYTES.

FORMAT:          OP1     OP2      OP3

  CMPB           BX3     BX3      R3

PERFORMANCE:                  RESULTANT_CPU_CYCLES.

  FOR OP3=0       :           5
  FOR OP3<=22     :           20 + 7*(OP3-1)
  FOR OP3>>22     :           7.6 * OP3

FUNCTION:

  RESULT:=OP3
  WHILE OP3>0 DO
    BEGIN
      W1:=OP1-OP2
      CASE W1 OF
        BEGIN
          W1 POSITIVE:  RESULT:=2; JUMP TO EXIT;
          W1 NEGATIVE:  RESULT:=0; JUMP TO EXIT;
          W1 ZERO     :  RESULT:=1;
        END
      INCR (INDX1);
      INCR (INDX2);
      DECR OP3
    END;
  EXIT: OP3:=RESULT;
        CLEAR MODIFICATION;
        CLEAR PSW[12];

DESCRIPTION:

  INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE
  INSTRUCTION PERFORMES A COMPARISON OF TWO BYTE-STRINGS WITHIN THE
  CURRENT PAGE, OF EQUAL BUT VARIABLE SIZE, AND DELIVERS THE RESULT
  EQUAL, LESS THAN OR GREATER THAN. EQUAL POSITIONED BYTES ARE
  COMPARED UNTIL ALL POSITIONS HAVE BEEN PROCESSED OR UNTIL THE
  FIRST UNMATCH OCCURS. THE VALUES OF THE NON-MATCH BYTES ARE USED
  FOR RESULT DETERMINATION. THE FOLLOW- ING RESULT CODES ARE USED:

  OP3=0 : NONMATCH. OP1 < OP2. INDX1 AND INDX2 POINTS AT THE
          NONMATCH BYTES.
  OP3=1 : THE TWO BYTE-STRINGS ARE IDENTICAL. INDX1 AND INDX2
          POINTS AT THE BYTE-PAIR SUCCEDING THE LAST PROCESSED
          PAIR.
  OP3=2 : NONMATCH. OP1 >OP2. INDX1 AND INDX2 POINTS AT THE
          NONMATCH BYTES.
```

NOTE: BX3 MEANS A BYTE OPERAND, THE BYTE ADDRESS OF WHICH IS CALCULA- TED BY ADDING 2*BASE TO THE CONTENTS OF THE REGISTER. THE CURRENT PAGE IS REGARDED AS A CYCLIC MEMORY.

CODE:

				I	I		I	I		I		I		I
CMPB	BX3	BX3	R3	I	0	I	BX3	I	X	I	BX3	I	00011	I	R3	I
				I		I		I		I		I		I		I
							OP1				OP2				OP3	

```
*****************     FILL BYTES     **********************************

INSTRUCTION: FILL BYTES.

FORMAT:          OP1       OP2      OP3

  FILB           R3        BX3      R3

PERFORMANCE:                      RESULTANT_CPU_CYCLES.

  FOR OP3=0        :              5
  FOR OP3<=50      :              16 + 3*(OP3-1)
  FOR OP3>>50      :              3.3 * OP3

FUNCTION:

  WHILE OP3 > 0 DO
    BEGIN
      OP2:=LOWER BYTE OF OP1;
      INCR (INDX2);
      DECR OP3
    END;
  CLEAR MODIFICATION;
  CLEAR PSW[12];

DESCRIPTION:

  INSTRUCTION BELONGING TO THE ALTERNATIVE INSTRUCTION SET. THE
  INSTRUCTION PERFORMS A COPY OPERATION OF A CONSTANT BYTE VALUE
  INTO ALL POSITIONS OF A BYTE STRING OF VARIABLE SIZE IN THE
  CURRENT PAGE. NOTE: BX3 MEANS A BYTE OPERAND, THE ADDRESS OF
  WHICH IS CALCULATED BY ADDING 2*BASE TO THE CONTENTS OF THE
  REGISTER. THE CURRENT PAGE IS REGARDED AS A CYCLIC MEMORY. AT
  EXIT OP3 WILL BE ZERO, OP1 WILL BE UNCHANGED, AND INDX2 WILL
  POINT AT THE BYTELOCATION SUCCEDING THE LAST BYTE PROCESSED. ONLY
  BIT 12 OF THE PSW IS CHANGED.

CODE:
```




```
  FILB      R3  BX3  R3
```

```
****************     ADD DECIMAL     **********************************

INSTRUCTION: ADD DECIMAL.

FORMAT:         OP1      OP2      OP3

  ADDD          R3       R3       R3

PERFORMANCE.                    CPU CYCLES.

  FOR OP3=0       :             8
  FOR OP3<=16     :             46 + 20*(OP3-1)
  FOR OP3>>16     :             21.7 * OP3

FUNCTION:

  WHILE OP3 > 0 DO
    BEGIN
      W1:=BYTEMEM(OP1+2*BASE+OP3-1);
      W2:=BYTEMEM(OP2+2*BASE+OP3-1);
      IF NON_NUMERIC_ASCII_CHARACTER THEN
        BEGIN
          RESULTCODE:=0;
          GOTO EXIT
        END
      ELSE
      W3:=W1+W2+PSW[4];   % ASCII/BCD ARITHMETIC OPERATION,LOADING
                            PSW[4] WITH CARRY AT OVERFLOW
      BYTEMEM(OP2+2*BASE+OP3-1):=W3;
      DECR OP3
    END;
  CASE PSW[4] OF
    BEGIN
      0 : RESULTCODE:=2;
      1 : RESULTCODE:=1;
    END
  EXIT:  CASE RESULTCODE OF
           BEGIN
             0 : LOC:=LOC+1  % SKIP 0
             1 : LOC:=LOC+2  % SKIP 1
             2 : LOC:=LOC+3  % SKIP 2
           END
  CLEAR MODIFICATION;
  CLEAR PSW[12];

DESCRIPTION:

  INSTRUCTION BELONGING TO THE  ALTERNATIVE  INSTRUCTION  SET.  THE
  INSTRUCTION  PERFORMS  THE  OPERATION A:=A+B ON TWO DECIMAL ASCII
  STRINGS OF EQUAL BUT VARIABLE SIZE WITHIN THE CURRENT  PAGE.  THE
  INSTRUCTION  ADDS EQUAL POSITIONED BYTES, ACCOUNTING FOR OVERFLOW
  CONDITIONS, AND DETECTING ILLEGAL (NON-NUMERIC) ASCII CHARACTERS.
```

THE CURRENT PAGE IS REGARDED AS A CYCLIC MEMORY. AT EXIT, PSW[4] IS SET IF THE LAST ADDITION CAUSED OVERFLOW.

PSW[7:5] IS UNDEFINED.

PSW[12] IS CLEARED.

CODE:

```
                                 ___ ____ ___ ____ _______ ____
                                I   I    I   I    I       I    I
  ADDC     R3  R3  R3          I 0 I R3 I C I R3 I 00101 I R3 I
                                I___I____I___I____I_______I____I
                                    OP2       OP1         OP3
```

```
****************     SUBTRACT DECIMAL     ***************************

INSTRUCTION: SUBTRACT DECIMAL.

FORMAT:         OP1       OP2       OP3

  SUBD          R3        R3        R3

FUNCTION:

PERFORMANCE:                  RESULTANT_CPU_CYCLES.

  FOR OP3=0       :           8
  FOR OP3<=16     :           46 + 19*(OP3-1)
  FOR OP3>>16     :           20.7 * OP3

  WHILE OP3 > 0 DO
    BEGIN
      W1:=BYTEMEM(OP1+2*BASE+OP3-1);
      W2:=BYTEMEM(OP2+2*BASE+OP3-1);
      IF NON_NUMERIC_ASCII_CHARACTER THEN
        BEGIN
          RESULTCODE:=0;
          GOTO EXIT
        END
      ELSE
      W3:=W1-W2-PSW[4];   % ASCII/BCD ARITHMETIC OPERATION, ADDING
      10
                            AND SETTING PSW[4] AT UNDERFLOW.
      BYTEMEM(OP2+2*BASE+OP3-1):=W3;
      DECR OP3
    END;
  CASE PSW[4] OF
    BEGIN
      0 : RESULTCODE:=2;
      1 : RESULTCODE:=1;
    END
  EXIT:  CASE RESULTCODE OF
           BEGIN
             0 : LOC:=LOC+1  % SKIP 0
             1 : LOC:=LOC+2  % SKIP 1
             2 : LOC:=LOC+3  % SKIP 2
           END
  CLEAR MODIFICATION;
  CLEAR PSW[12];

DESCRIPTION:

  THE  INSTRUCTION  BELONGS TO THE ALTERNATIVE INSTRUCTION SET. THE
  INSTRUCTION PERFORMS THE OPERATION A:=A-B ON  TWO  DECIMAL  ASCII
  STRINGS  OF  EQUAL BUT VARIABLE SIZE WITHIN THE CURRENT PAGE. THE
  INSTRUCTION SUBTRACTS  EQUAL  POSITIONED  BYTES,  ACCOUNTING  FOR
```

UNDERFLOW CONDITIONS, AND DETECTING ILLEGAL (NON-NUMERIC) ASCII CHARACTERS. THE CURRENT PAGE IS REGARDED AS A CYCLIC MEMORY.

AT EXIT PSW[4] IS SET IF THE LAST SUBTRACTION CAUSED UNDERFLOW.

PSW[7:5] ARE UNDEFINED. PSW[12] IS CLEARED.

CODE:

```
                                  ___ ____ ___ ____ _______ ____
                                 I   I    I   I    I       I    I
SUBC     R3  R3  R3              I 0 I R3 I 1 I R3 I 00101 I R3 I
                                 I___I____I___I____I_______I____I
                                     OP2      OP1         OP3
```

DATA TYPES AND ORGANIZATION

The CR80 is capable of handling the following main data types:

- single bit
- byte (8 bits)
- integer (16 bits)
- long integer (32 bits)
- floating point number (32 bits)
- long floating point number (64 bits)
- decimal numbers (variable length)

The main memory is organized as sequences of 16 bits words. The general purpose registers each contain a 16 bit word.

In the following is shown the mapping of each individual data type onto 16 bits words.

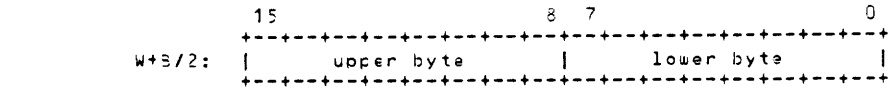
single bit

A single bit is selected within a word by indexing in the range 0..15 as illustrated below:

Bit 0 is the least significant bit.

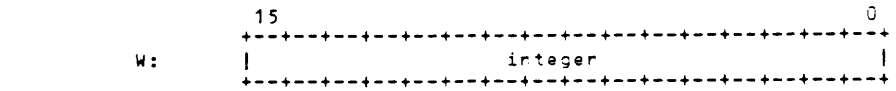
byte (8 bits)

A byte is selected via a word address (W) and a byte offset (B). If the byte offset is even, the lower byte is selected (bits 0..7), otherwise the upper byte is selected (bits 8..15).



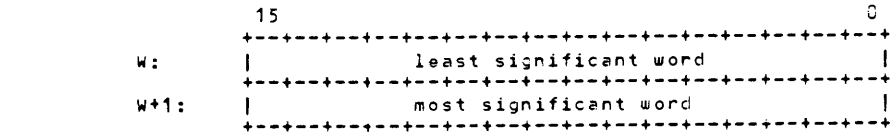
integer (16 bits)

An integer is selected via a word address (W):



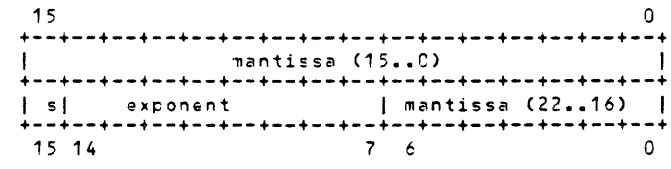
long integer (32 bits)

A long integer consists of two consecutive words (memory words or general purpose registers), and it is selected via the word address of the least significant word (W).



floating point number (32 bits)

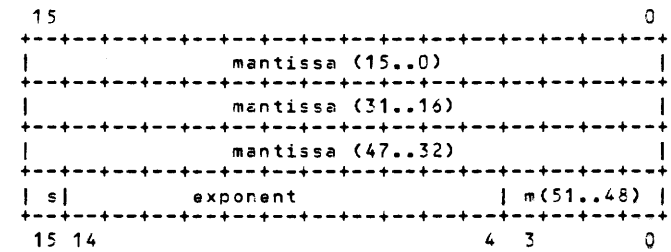
A floating point number consists of a sign, an 8 bits exponent and a 32 bits mantissa. A floating point number is stored in two consecutive words (in memory) and it is selected via the word address (W) of the least significant part of the mantissa.



(s = sign)

long floating point number (64 bits)

A long floating point number consists of a sign, an 11 bits exponent and a 52 bits mantissa. A long floating point number is stored in four consecutive words (in memory), and it is selected via the word address (W) of the least significant part of the mantissa.

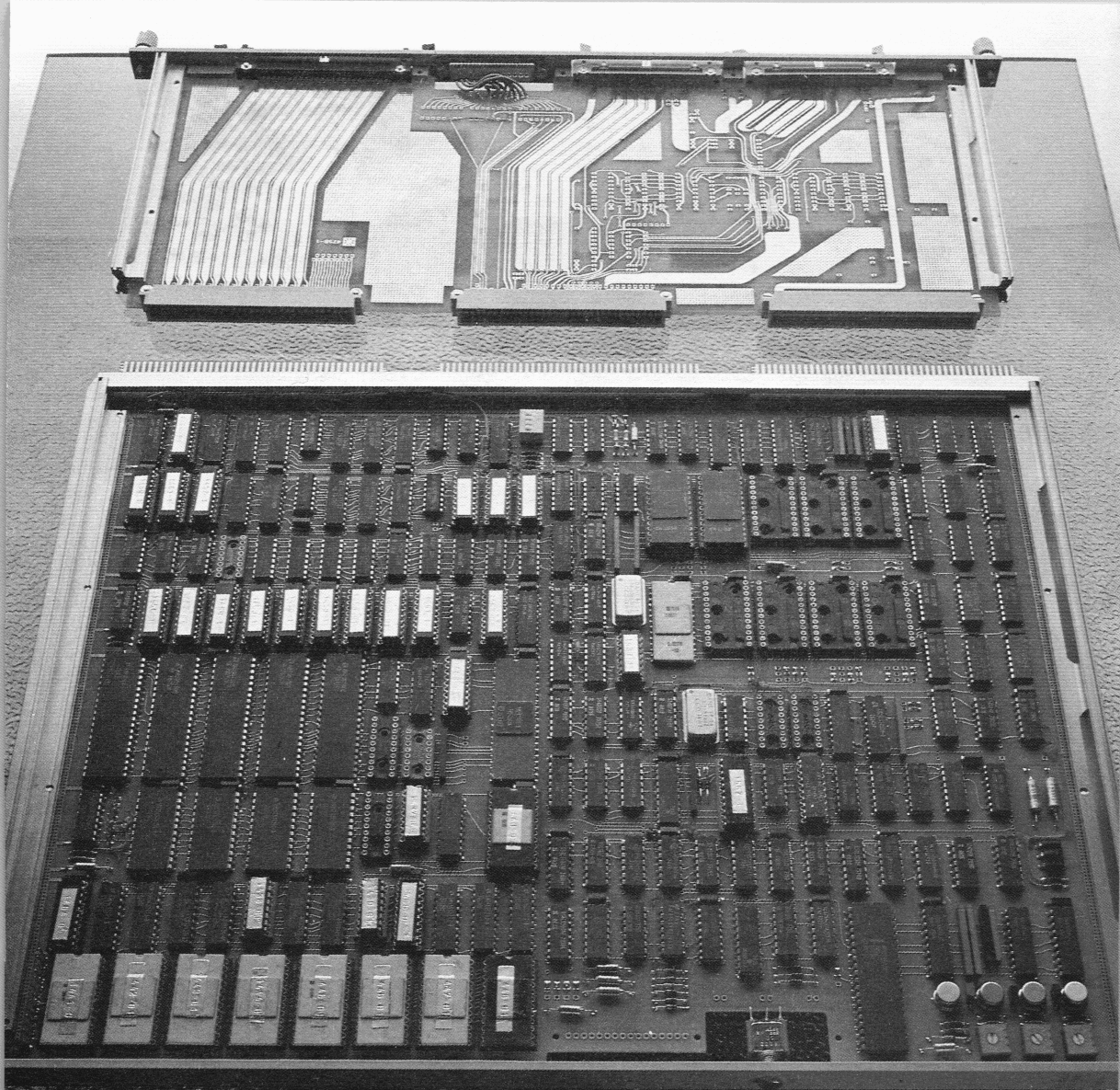


(s = sign)

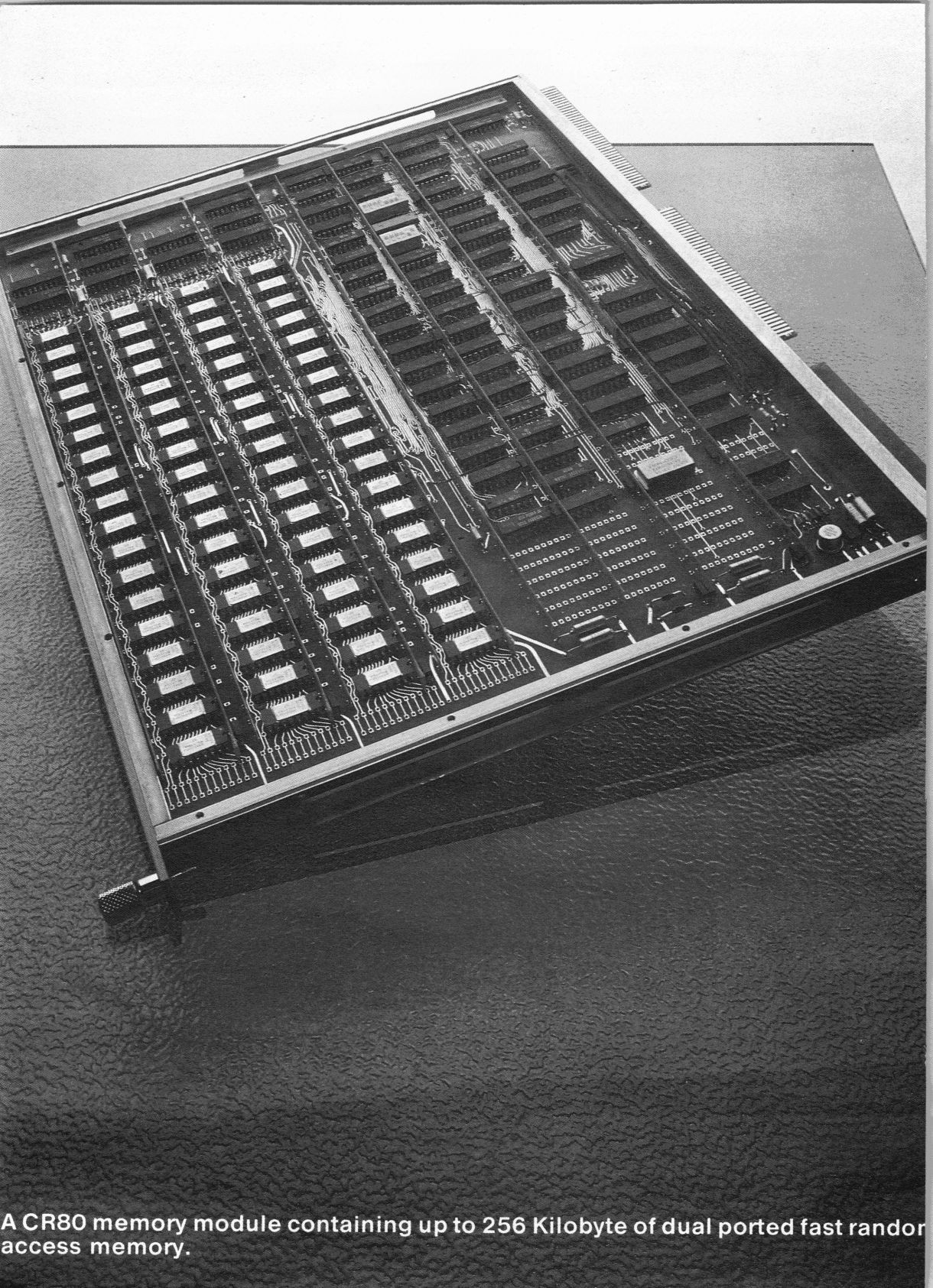
decimal number (variable length)

A decimal number consists of a variable number of consecutive bytes (in memory), each of which contains the ASCII value of a decimal digit. A decimal number is selected via the byte address (B) of the most significant byte, and the lower bytes are located at increasing byte addresses (B+1, B+2, ...).

Decimal numbers may start with lower or upper bytes (of the corresponding words) and their size may be an odd or an even number of bytes.



e CR80 CPU/SCM module and associated Adapter module provide the basic
ocessing power of CR80 non-mapped MINI and TWIN Computers. The module
a single 8 layer printed circuit board contains a complete bit-slice CPU, all
stem Control functions, a serial port and two interfaces to fast line printers, as
ll as 14 kilobyte RAM/EPROM.



A CR80 memory module containing up to 256 Kilobyte of dual ported fast randor access memory.

10. CR80 Standard Modules

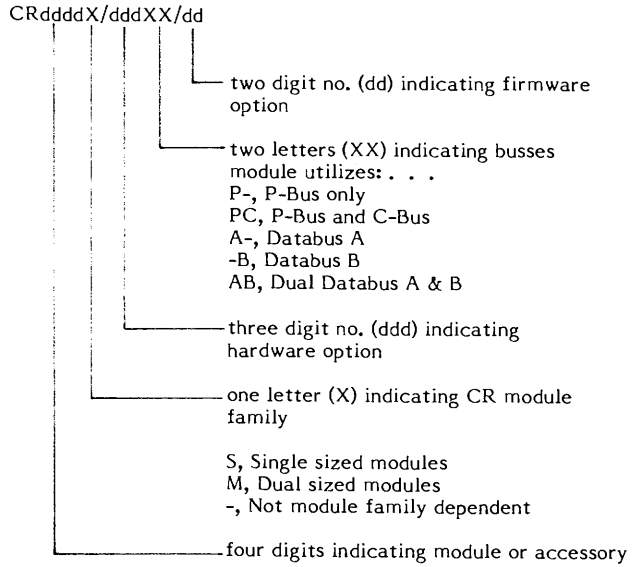
10.1 Introduction

In the following the nomenclature and breakdown of the module and accessory type numbers are described. The index and the following data sheets covers the following CR80 standard products:

CR80 Standard Modules
CR80 Peripheral Modules
CR80 Power Supply and Miscellaneous Modules
CR80 Communication Modules
CR80 Adapter Modules
CR80 Watchdog Subsystem
CR80 Rack, Crates and Accessories
CR80 Cables & Cable Accessories
CR80 Peripheral Equipment
TDX and X-Net Modules
TDX and X-Net Crates & Accessories
MP^2 Board and Accessories

10.2 Nomenclature of Module Type Numbers in the Index

CR80M, TDX modules and accessories follow the below breakdown of the type number:



CR80M Modules (8000-8099):

8000-8009 P-Bus Devices
8010-8019 Memories
8020-8029 C-Bus Devices
8030-8049 Peripheral Interfaces
8050-8059 Power Supply & Miscellaneous
8060-8069 Communication Interfaces
8070-8099 Adapters

CR80M Rack, Crates & Accessories (8100 8199):

8100-8109 Racks & Associated
8110-8119 CR80M Crates
8120-8129 CR80M Crates
8130-8139 Adapter Crates
8140-8159 Miscellaneous

CR80M Cables (8200-8299):

8200-8299 Cables

CR80 Peripherals (8300-8399):

8300-8309 Disc Drives
8310-8319 Consumerables: Disc Packs, Tapes, Paper,etc.
8320-8329 Tape Drives
8330-8339 Line Printer
8340-8349 TBS
8350-8359 Terminals VDU
8360-8369 TBS
8370-8379 TBS
8380-8389 TBS
8390-8399 Consoles, VDU's, etc.

TDX & X-Net (1000-3000):

1000-1099 TDX Modules & Accessories
2500-2599 X-Net Modules & Accessories

EXAMPLES:

CR8002M/010P-/00	CPU+SCM, single bus
CR8003M/040PC/00	CPU+CACHE, dual bus
CR8045M/040AB/00	TAPE CTRL, dual bus
CR1070S/000--/00	TDX & X-Net Controller

10.3 CR80 Module, Accessory and Peripheral Index

CR80 Module, Accessory and Peripheral Index

	Type	Description	Ref	Page
10.3.3	CR80M Power Supply & Miscellaneous Modules:			
	CR8050M/010--/00	Power Supply, PU Crate or CU Crate	50.0	10-80
	CR8055M/020--/00	Bus Termination, Processor Bus, Channel Bus, A Bus or B Bus	55.1	10-81
10.3.4	CR80M Communication Interface Modules:			
	CR8066M/030A-/00	4 Ch., ASYNC LTU, Single Bus (intern RAM: 2 + 2K)	66.0	10-82
	CR8066M/010A-/00	4 Ch., SYNC LTU, Single Bus (HDLC, SDLC, BSC Protocols etc.) (intern RAM: 16 + 16K)	66.1	10-82
	CR8066M/030AB/00	4 Ch., ASYNC LTU, Dual Bus (intern RAM: 2 + 2K)	66.3	10-84
	CR8066M/010AB/00	4 Ch., SYNC LTU, Dual Bus (HDLC, SDLC, BSC Protocols etc.) (intern RAM: 16 + 16K)	66.4	10-84

10.3.5 CR80M Adapters Modules:

Type	Description	Ref	Page

10.3.6 CR80M Watchdog Subsystem

WPU: Watchdog CPU, see:	CR8066M/010AB/00 (sync. LTU)
WCA: Watchdog CPU Adapter, see:	CR8076M/010--/00 (WCA)
WPC: Watchdog Panel Controller,see:	CR8040M/010AB/00 (WPC)
CCA: Crate Configuration Contrl., see:	CR8089M/010--/00 (CCA)
CCA: Minicrate, Watchdog, see:	CR8115M/005P-/00 (Minicr. WD)
CBC: Configuration Bus Cable, see:	CR8209M/XXX--/00 (cable, config. bus)

10.3.7 CR80M Rack, Crates & Accessories

CR8101-/042--/00	19" Rack M:42", H:7' (42U)	101.0	10-114
CR8101-/036--/00	19" Rack D:29", H:6' (36U) (up to 2 M-Crates)	101.1	10-115
CR8102-/000--/00	19" Rack D:29", H:3' 6" (18U) (1 Minicrate)	102.0	10-116
CR8105M/020--/00	Fan unit for M-Crates Redundant (1 needed each crate)	105.0	10-117
CR8105S/010--/00	Fan unit for LTUX and S-Crates (1 needed each crate)	105.1	10-119
CR8106-/220--/00	Mains Filter, 220V (1 needed each rack)	106.0	10-120
CR8106-/110--/00	Mains Filter, 110V, 60Hz (1 needed each rack)	106.1	10-120
CR8106-/240--/00	Mains Filter, 240V (1 needed each rack)	106.2	10-120
CR8107-/010--/00	Power Distribution Panel (1 or 2 needed each rack)	107.0	10-121
CR8112M/212PC/00	PU-Crate for 2 PU's, 12+12 pos. incl. dual busses, adaptor crate, module to adapter cables. Needs 4xCR8055M/020--/00, bus termination modules not included.	112.0	10-122

CR80 Module, Accessory and Peripheral Index

CR80 Module, Accessory and Peripheral Index

Type	Description	Ref	Page
CR8147M/001--/00	Blank front panel, 1M wide	147.0	10-146
CR8147M/002--/00	Blank front panel, 2M wide	147.1	10-146
CR8147M/004--/00	Blank front panel, 4M wide	147.2	10-146
CR8147M/008--/00	Blank front panel, 8M wide	147.3	10-146
CR8148M/001--/00	Blank rear panel, 1M wide	148.0	10-147
CR8148M/002--/00	Blank rear panel, 2M wide	148.1	10-147
CR8148M/004--/00	Blank rear panel, 4M wide	148.2	10-147
CR8148M/008--/00	Blank rear panel, 8M wide	148.3	10-147
CR8149M/001--/00	Blank front panel, rack, 1U	149.0	10-148
CR8149M/002--/00	Blank front panel, rack, 2U	149.1	10-148
CR8149M/003--/00	Blank front panel, rack, 3U	149.2	10-148
CR8149M/004--/00	Blank front panel, rack, 4U	149.3	10-148
CR8149M/005--/00	Blank front panel, rack, 5U	149.4	10-148
CR8149M/006--/00	Blank front panel, rack, 6U	149.5	10-148

*NOTE: CR8125M (PU or CU) Crates are exclusive external cables, Power, Data channel, Supra or TDX (see CR80M cables for separate ordering).
*NOTE: Minicrate & Fan units (CR80M, LTUX and CR80S) are exclusive power cables (see CR80M cables for separate ordering).

Type	Description	Ref	Page
10.3.8	**CR80M Cables & Cable Accessories**		
CR8201M/015--/00	Power cable, rack power distribution panel to M-Crate, minicrate or M-fan unit (150cm) (up to 2 needed for each M-Crate (1 or 2 power supplies), 1 needed for each minicrate, 2 needed for each M-Crate fan unit)	201.0	10-149
CR8201S/015--/00	Power cable, rack power distribution panel to floppy station, LTUX crate or LTUX crate fan unit, 1 needed for each.	201.1	10-150
CR8209M/100--/00	Cable, configuration bus, 10m, delivered with female connectors mounted at 1m intervals	209.0	10-151
CR8209M/200--/00	Cable configuration bus, 20m, delivered with female connectors mounted at 1m intervals	209.1	10-151
CR8209M/300--/00	Cable, configuration bus, 30m, delivered with female connectors mounted at 1m intervals	209.2	10-151
CR8211M/005--/00	Cable, Data channel, 50cm	211.0	10-152
CR8211M/015--/00	Cable, Data channel, 150cm	211.1	10-152
CR8211M/030--/00	Cable, Data channel, 300cm	211.2	10-152
CR8211M/060--/00	Cable, Data channel, 600cm	211.3	10-152
CR8211M/120--/00	Cable, Data channel, 1200cm	211.4	10-152
CR8211M/738--/00	Terminator for Data channel, 1 needed to terminate one data channel chain at end of chain	211.5	10-153
CR8212M/015--/00	Cable, extension bus, EPA to EIA 150cm	212.0	10-154
CR8212M/030--/00	Cable, extension bus, EPA to EIA 300 cm	212.1	10-154

Type	Description	Ref	Page
10.3.9	CR80 Peripheral Equipment (Peripherals non-discountable)		
CR8300-/040--/00	Disc Drive, SMD, Removable Disckpack, 40MB, incl. Cabinet & Standard Options	300.0	10-165
CR8300-/080--/00	Disc Drive, SMD, Removable Discpack, 80MB, incl. Cabinet & Standard Options	300.1	10-165
CR8300-/150--/00	Disc Drive, SMD, Removable Disckpack, 150MB, incl. Cabinet & Standard Options	300.2	10-165
CR8300-/300--/00	Disc Drive, SMD, Removable Discpack, 300MB, incl. Cabinet & Standard Options	300.3	10-165
CR8300-/001--/00	SMD OPTION, Acoustic Cabinet	300.4	10-165
CR8300-/002--/00	SMD OPTION, Dual Channel	300.5	10-165
CR8300-/003--/00	SMD OPTION, Cable A, see CR80M cables, CR8221 M	300.6	
CR8300-/004--/00	SMD OPTION, Cable B, 20' see CR80M cables, CR8221 M	300.7	
CR8301-/016--/00	Disc Drive, CMD, One Fixed and One Removable Discpack (16MB), 16+16MB	301.0	10-166
CR8301-/048--/00	Disc Drive, CMD, One Fixed and One Removable Discpack (16MB), 48+16MB	301.1	10-166
CR8301-/080--/00	Disc Drive, CMD, One Fixed and One Removable Discpack (16MB), 80+16MB	301.2	10-166
CR8302-/012--/00	Disc Drive, MMD, Fixed Pack, 12MB	302.0	10-167
CR8302-/024--/00	Disc Drive, MMD, Fixed Pack, 24MB	302.1	10-167
CR8302-/080--/00	Disc Drive, MMD, Fixed Pack, 80MB	302.2	10-167
CR8302-/160--/00	Disc Drive, MMD, Fixed Pack, 160MB	302.6	10-167
CR8302-/001--/00	MMD Option, 1MB Fixed Head for CR8302-/024--/00	302.3	10-167
CR8302-/002--/00	MMD Option, 1MB Fixed Head for CR8302-/080--/00	302.4	10-167
CR8302-/003--/00	MMD Option, 2MB Fixed Head for CR8302-/080--/00	302.5	10-167
CR8308-/216--/00	Floppy disk station, dual drive, single side incl. 19" rack mountable enclosure and power supply.	308.0	10-168
CR8308-/108--/00	Floppy disk station, single drive, single side incl. 19" rack mountable enclosure and power supply.	308.1	10-168

CR80 Module, Accessory and Peripheral Index

Type	Description	Ref	Page
CR8308-/232--/00	Floppy disk station, dual drive, dual side incl. 19" rack mountable enclosure and power supply.	308.2	10-168
CR8308-/116--/00	Floppy disk station, dual drive, single side incl. 19" rack mountable enclosure and power supply.	308.3	10-168
CR8319-/016--/00	Disc Cartridge 16MB for CMD Drives	319.0	10-169
CR8319-/040--/00	Disc Pack 40MB for SMD Drives	319.1	10-170
CR8319-/080--/00	Disc Pack 80MB for SMD Drives	319.2	10-170
CR8319-/150--/00	Disc Pack 150MB for SMD Drives	319.3	10-170
CR8320-/001--/00	Pertec FT8000 Tape Station (1600Bpi, PE, IBM Compatible, 25ips)	320.0	10-171
CR8320-/002--/00	Pertec FT9000 Tape Station (1600Bpi, PE, 75ips)	320.1	10-171
CR8330-/200--/00	Matrix Printer, 200Lpm. (DP: M200)	330.0	10-172
CR8331-/300--/00	Band Printer, 300Lpm. (DP: B300)	331.0	10-173
CR8331-/600--/00	Band Printer, 600Lpm. (DP: B600)	331.1	10-173
CR8332-/300--/00	Drum Printer, 300Lpm. (DP: 2230)	332.0	10-174
CR8332-/600--/00	Drum Printer, 600Lpm. (DP: 2260)	332.1	10-174
CR8332-/900--/00	Drum Printer, 900Lpm. (DP: 2290)	332.2	10-174
CR8333-/064--/00	Matrix Printer, 64-440 Lpm. (TI810), full ASCII, with keyb.	333.0	10-175
CR8350-/004--/00	VDU, V24 Async., 8 Functional Keys (Infoton 400/4), incl. polling option	350.0	10-176
CR8350-/005--/00	VDU, V24 Async., 20 Functional Keys (Infoton 400/5), incl. polling option	350.1	10-176
CR8390-/034--/00	Matrix Printer w. Keyboard, V24, 30 Cps (Decwr.LA34) tractorfeed	390.0	10-177
CR8391-/820--/00	Matrix Printer w. Keyboard, V24, 300 Cpm (TI820) full ASCII	391.0	10-178
CR8392-/743--/00	Thermal Printer w. Keyboard, V24 (TI743), full ASCII	392.0	10-179

*NOTE: Peripherals are exclusive of external cables (for connecting to CR80M controllers), see CR80M cables for separate ordering.

10.3.10 TDX and X-net

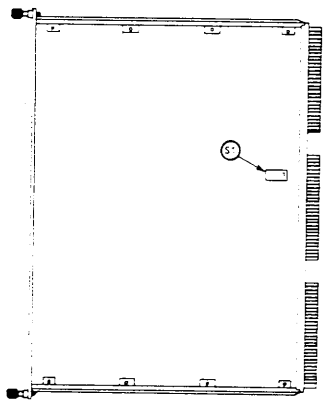
TDX & X-net Modules

Type	Description	Ref	Page
CR1070S/000--/00	TDX & X-net controller	470.0	10-180
CR8021M/010-C/00	STI, TDX & X-net CR80M Host I/F, (unmapped systems),see CR80M standard modules.		
CR8021M/010-C/00	STI, TDX, X-net & Supra Host I/F, (mapped systems), see CR80M standard modules		
CR8073M/000--/00	TIA, TDX bus/CR80M I/F adapter, see CR80M adapters.		
CR1060S/000--/00	LTUX-S, single board LTUX	460.0	10-182
CR1061S/000--/00	LTUX-M-X21	460.1	10-187
CR1090S/000--/00	LTUX-M-FE front end for multiple board LTUX	490.0	10-190
CR1091S/000--/00	LTUX-M-CPU processor for multiple board LTUX	491.0	10-192
CR1092S/000--/00	PIO-X TTL input	492.0	10-196
CR1092S/010--/00	PIO-X 16 relay/16 opto	492.1	10-196
CR1092S/011--/00	PIO-X 16 relay/16 opto	492.2	10-196
CR1092S/020--/00	PIO-X 32 opto	492.3	10-196
CR1092S/021--/00	PIO-X 32 opto	492.4	10-196
CR1092S/030--/00	PIO-X 32 relay	492.5	10-196
CR1093S/000--/00	ADC-X AD converter	493.0	10-198
CR1094S/000--/00	DAC-X D/A converter, 2ch.	494.0	10-198
CR1094S/010--/00	DAC-X D/A converter, 8 ch.	494.1	10-200
CR1067S/000--/00	DL-X dataline timecode I/F	467.0	10-200
CR2505-/000--/00	Chassis and Power Supply for X-net devices (used with X-net controller CR1070 if space not available in X-net port chassis's)	405.0	10-204
CR2510-/000--/00	TDX or X-Net wall outlet,XWO	400.0	10-206
CR2550-/000--/00	X-net terminal adapter, XTA incl. CR2505 chassis and power supply	404.0	10-208
CR2560-/004--/00	X-net port, XP, 4xV24 async., incl. CR2505 chassis and power supply	406.0	10-211
CR2560-/008--/00	X-net port, XTP, 8xV24 async., incl. CR2505 chassis and power supply	406.1	10-211
CR2560-/012--/00	X-net port, XTP, 12xV24 async., incl. CR2505 chassis and power supply	406.2	10-211
CR2500-/000--/00	X-net Cable (100m)	402.0	10-214

DATASHEET FOR:

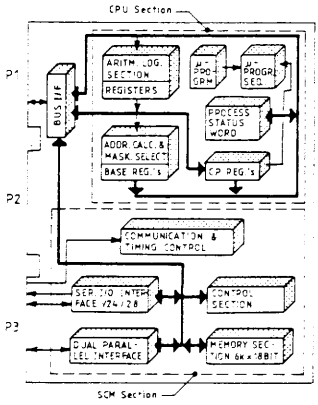
CPU + SCM CR 8002M/010P-/00

SET UP REFERENCE



FRONT PANEL

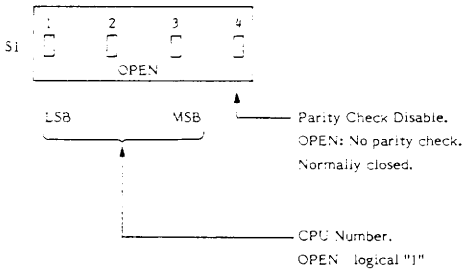
BLOCK DIAGRAM



CHARACTERISTICS:

- The module is the basic module in a non-mapped CR80 system.
- The module contains a 16 bit general purpose Central Processing Unit (CPU) and a System Control Module (SCM).
- Two instruction sets are available, one set contains the CR80 standard instruction set.
- 8 general purpose registers and 6 special purpose registers.
- Micro instruction cycle time 250 ns.
- Detection and Receiving I/O interrupts.
- P-Bus access control.
- Real time clock 10 ms.
- Dual Parallel I/F.
- Serial I/F.
- Up to 6K of memory.

PROGRAMMERS REFERENCE:



- Power Consumption:

+5V:	11A
+12V:	20mA (typical)
-12V:	54mA (typical)

- Mechanical Dimensions:

Height:	411.0 mm (10 U)
Width:	17.1 mm (1M)
Length:	305.0 mm

- MTBF: 36,500 hours
- Weight: 1.2 kg

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

CR80 STANDARD INSTRUCTION SET

Instructions are 16 bit words, and provide the user with a large selection of functions. Each function is implemented with several addressing schemes.

However, all instructions have the possibility of address modifications in which case an arbitrary number of words may precede the instructions to build up any wanted addressing scheme.

The instructions are implemented in a 2K word PROM micro memory.

The instruction set includes instructions for operation on single bits and provides semaphore protection of memory areas.

REGISTERS

The CPU contains 8 general purpose registers R0 - R7, a data base register B, a program register P, a program counter PC, and a time register T. The general registers R0 - R7 may be used as accumulators or index registers of choice.

Data is always addressed relatively to the contents of the base register B. Instructions are addressed relatively to P. The time register T is decremented by timer interrupts. When T becomes negative, an internal interrupt is generated. T may be used for exact timing of events. The timer interrupts can be masked independently of other interrupts.

INTERRUPT

The CR8002 has 256 I/O interrupt levels which are grouped in 4 main levels IL. Interrupts are handled either by a user defined microprogram or by switching of software processes.
The CPU is only interrupted when I/O interrupts occur which have priorities higher than or same as the priority currently defined in the CPU Process Status word. When an I/O interrupt is received by the CPU, the CPU switches to a user defined process or optionally branches to a user defined micro routine.
Communication between CPU's is accomplished by means of a special type of interrupts: CPU interrupts. CPU interrupts have higher priority than ordinary I/O interrupts.
I/O and CPU interrupts can be masked independently.

ADDRESSING

Instructions are addressed in 16 bit words, and a maximum of 64K instruction words can be addressed.

Data may be addressed in several formats:

single bit, 8 bit bytes, 16 bit words,
32 bit words, and n x 16 bit words where
n = 1, —— 16.

The CPU is able to address 256K 16 bit words. Addresses consist of a 2 bit page address which is contained in the Process Status Word, and a 16 bit data address.

For detailed specification of the CPU including instruction set please refer to CR80 Handbook.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

DESCRIPTION (SCM PART)

The SCM act as supervisor for the CPU's .
The SCM generates the timing signals necessary for synchronization of the modules belonging to the CR80 system.

Detecting and receiving of serially transmitted I/O interrupt code are carried out in the SCM. The two interrupt priority bits are fed through to the CPU for control of process switching and thereby fetching of the interrupt code in parallel form. the SCM senses on the power lines, and generates a power failure interrupt if a drop occurs. the authority control for the Main Bus is designed so that up to seven CPU's can be connected to the same CR80 System, and so that no time will be lost when the authority is changed from one CPU to another.

The Memory part of the SCM communicates with the CR80 System via the Main bus. It consists of three areas, each 2K x 16 bits. Each of the three areas can be independently addressed. The SCM can contain the following form of memory:

4K RAM and 2K PROM
2K RAM and 4K PROM
6K PROM

Where the PROM's can be bipolar or UV erasable. Optionally every memory area can consist of 4K UV erasable PROM's allowing the SCU to contain an area of up to 12K UV erasable PROM.

SERIAL I/O INTERFACE

Asynchroneous/synchroneous serial RECEIVE/TRANSMIT in full or half duplex.

Reception/Transmission speed is determined by the software. The following Baud rates are available: 50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, 19200 Baud.

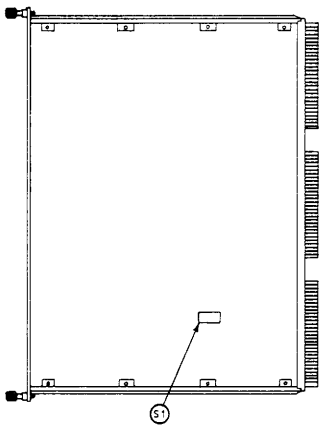
Parity is software selectable.

1, 1½ or 2 stop bits are software selectable.
Electrical, mechanical, and logical the SIOI's I/O slot conform to CITT V24/V28 and EIA RS232-C recommendation.

DUAL PARALLEL I/F

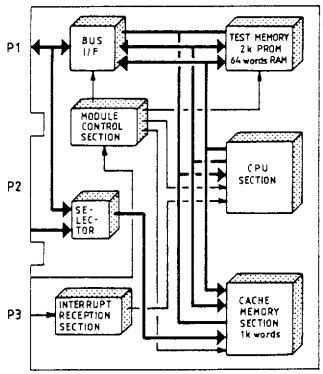
The SCM contains an 8 bit input and 8 bit output parallel port, with separate handshaking signals. The electrical level is TTL.

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

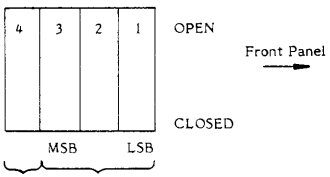
- 16 bit general purpose CPU with interface 1K word cache memory for use in the CR80 system.
- Internal test facilities, providing extensive on-line and off-line test capabilities.
- Can be used in multiprocessor system.
- Operates with and without MAP module.
- Cache memory transparent to software.
- Mechanical Dimensions:

Height:	411 mm
Width:	17.1 mm (1M)
Length:	305 mm
Weight:	1.5 kg

- MTBF: 26,100 hours
- Power Consumption:
  +5V: 18A

PROGRAMMERS REFERENCE:

SWITCH SETTING (S1)



ENABLING/ DISABLING OF CACHE PARITY ERROR SIGNALLING TO CPU.
OPEN ~ ENABLED

CPU NUMBER.
OPEN ~0
CLOSED ~1

For further information, please refer to CR80 HANDBOOK, Cpt. 2.

CPU SECTION:

- The CPU accesses Memory and Peripheral modules via the P-Bus.
- Alternative instruction interpretation is possible. Current mode of interpretation is indicated in the Process Status Word (PSW).

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- One mode of instruction is used to implement the CR80 standard instruction set.

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 1 sec. if it has been carried out successfully.

- A second mode interpretation is used to implement an instruction set well suited for high level languages like Algol, PASCAL, PL/I, etc. This mode also allows customer defined instructions to be implemented.

- 8 general purpose registers

- SPECIAL purpose registers:

PSW:	Process status word
EXR:	Execution register
BASE:	Base register for data addressing
MOD:	Modify register
PROG:	Program base register
PRPC:	Program counter
TIMER:	Timer register
BOUND:	Bound register
CER:	Cache error register

- Fast register set switching possible activated by program or interrupts.

- HARDWARE SUPPORT:

  - In the CPU the P-Bus interface fetches and stores data while the processing part is running. Access to buffers in the Bus I/F is controlled by HW, eliminating processor overhead.
  - Three busses provided internally (Input BUS, Output BUS, and Address BUS).
  - Two ALU's are included, allowing address and data manipulations to be performed simultaneously when STACK machines are emulated or allowing address calculations involving three elements to be performed in one cycle for machines employing base register addressing.

- Micro instruction cycle times 250nS.

CACHE MEMORY SECTION:

- Increases processing speed.

- Uses a write through algorithm to avoid cache handling software.

- CPU read operations from the cache memory are carried out without loading the processor bus.

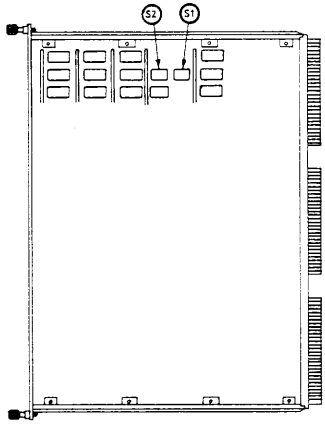
- Write operations to main memory location also present in the cache are detected on both the P-Bus and the C-Bus and the cache memory is updated accordingly.

- Automatic cache memory initialization at master clear included.

DATASHEET FOR:

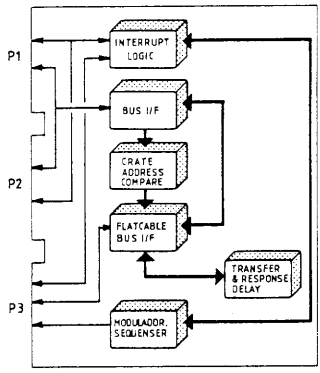
EPM CR8009M/010P-/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Active Main Bus Extension for the CR80 system together with EPA and EIA-A&B modules in unmapped systems.
- All CR80 Bus signals are interfaced.
- Interface between the two Main Busses:
  P-Bus and/or C-Bus & Flatcable Extension Bus in Processor Crate.
- Up to eight EPA modules can be connected to the Flatcable I/O Bus.
- Up to 15 EPM modules can be used in a Processor Crate when they are assigned to a different I/O Crate address each.
- Interrupt priority can be set by switch S1.
- The address of the used I/O crate can be set by switch S2 (can be disconnected by S1, contact CC).

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

S1

4	3	2	1
A15	A14	A13	A12

S2

4	3	2	1
P1	P0	MP	CC

Front Panel ←

S1 - DIL Crate Address Switch

Address determined by A12-A15; A15: MSB.
(0=switch ON; 1=switch OFF)

S2 - DIL Priority Switch

CC: crate address comperator ON/OFF
MP: manual priority setting * ON/OFF
P0: priority setting
P1: priority setting

(0=switch OFF, 1=switch ON)

* Note:

When the manual priority setting is OFF, the P1 and P0 settings must be OFF

CHARACTERISTICS: (CONTINUED)

- Typical delay of transfer is 300nsecs and of response is 200 nsecs during transfer from/to the extended Main Bus through the whole Extension System.

- Mechanical Dimensions:

Height:	412.6 mm
Width:	17.1 mm (1M)
Length:	305.0 mm

- Power Consumption:

+5V:	1.2A typ.

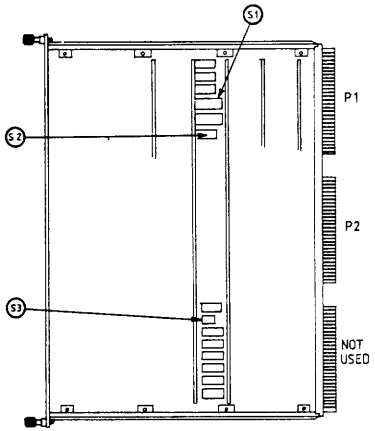
- MTBF: 172000 (H/F)

- Weight: 1000 g

DATASHEET FOR:

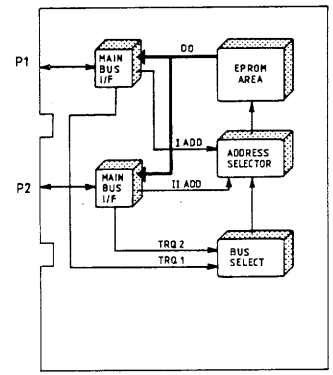
EPROM CR8013M/XXXPX/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- ERASABLE PROM for CR80 System.
- Dual or single ported for Channel Bus and/or Processor Bus Interface.
- Address within CR80 memory space set by switch S2.
- Memory area can be set by S1, depending of prom type.
- Read response time:
  125ns <tr< 562, depending of prom type and switch setting of S3.
- Equipped with 32K EPROM chips as standard.
- Standard module is 16K word (16 bit + 2) EPROM.
- Power consumption:
  +5V: max. 6A
- Mechanical dimensions:
  Height: 412.6 mm
  Width: 17.1 mm
  Length: 305 mm
  Weight: 1.1 kg
- MTBF: 10,873 h

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

S1

8	7	6	5	4	3	2	1
C0	C1	C2	C3	B0	B1	B2	B3

Front Panel ←

S2

6	5	4	3	2	1
A0	A1	A2	A3		

S3

4	3	2	1
D	C	B	A

Module Address Setting (S2)

Address determined by A0-3; A3:MSB
0∼switch ON. 1∼switch OFF

Memory Area Size (S1)

Upper limit B3-B0; B3:MSB
Lower limit C3-C0; C3:MSB
The space setting depends of Prom type (size) (16K, 32K and 64K can be used).

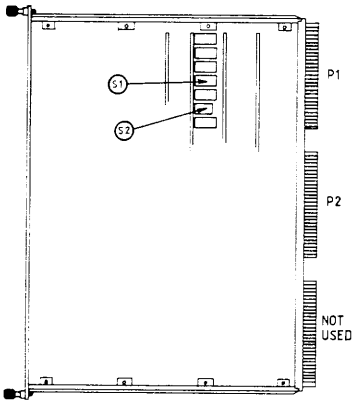
Response Time tr Setting (S3)

Presetting is binary. D:MSB

DATASHEET FOR:

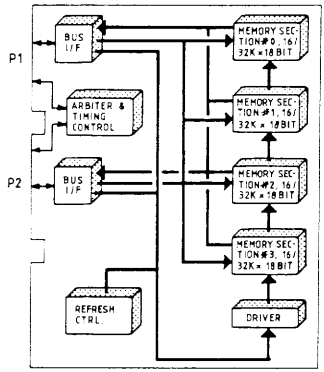
RAM CR8016M/XXXPX/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Random Access Memory for CR80 System.
- Dual or single ported, for Channel Bus and/or Processor Bus Interface.
- Address within CR80 memory space set by switch S2.
- Part of the memory can be disabled by switch S1.
- 2.6 mega access per second.
- Write response time TRW:
  190 ns<TRWmin<280ns, module not occupied by refresh or the other bus.
  TRWmax: TRWmin + 750 ns, module occupied by refresh and the other bus.
- Read response time T_{RR}:

  T_{RR}: T_{RW} + 125ns.
- 2.54% occupation due to refresh.
- Equipped with either 1x32K or 1x16K Dynamic RAM chips.

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

S1

8	7	6	5	4	3	2	1
C0	C1	C2	C3	B0	B1	B2	B3

S2

6	5	4	3	2	1
A0	A1	A2	A3	D	

Front Panel ←

Module Address Setting:

Address determined by A3-A0; A3: MSB

0~switch ON, 1~switch OFF

If module is equipped with 32K chips, A0 is don't care.

Module allocation within the one mega word memory space.

: P* 64K to (P+1)* 64K-1; P:A3-A0 (16 K RAM chips)

P*128K to (P+1)*128K-1; P:A3-A1 (32K RAM chips)

Disable Function:

32K chips { D:ON-> Part of upper 64K can be disabled
D:OFF-> Part of lower 64K can be disabled

16K chips { A0:OFF ∧ D:ON
A0:ON ∧ D:OFF } No disable can be performed
REMAINING Part of 64K can be disabled

Upper limit: B3-B0; B3:MSB

Addresses ≥ (N+1)*16K-> module is disabled

N:B3-B0

Lower limit: C3-C0; C3:MSB

Addresses < N*16K-> module is disabled

N:C3-C0

CHARACTERISTICS: (CONTINUED)

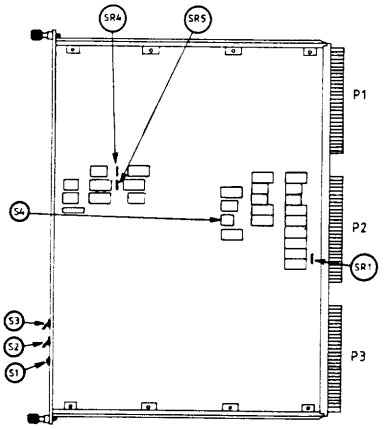
- Power Consumption:
  - +5V: 3.5A
  - +12V: 1A
  - -12V: 10mA

- MTBF: 128K: 17000 h
  - 64K: 29600 h

- Weight: 1.1 kg

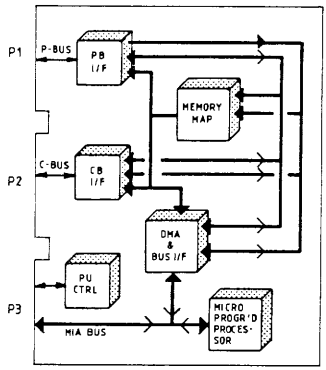
- Dimensions:
  - Height: 412.6 mm
  - Width: 17.1 mm
  - Length: 305.0 mm

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Extend the CR80 CPU address space to 16 mega words by logical-to-physical address translation.

  address space: Absent indication, Read only access, No access, Full access.

- Microprocessor included for interrupt processing, DMA, and AV24 handling.
- Intra Memory DMA controller performs move of data buffers within the total address space -including compartmentalized RAM of peripheral modules.
- Max. DMA rate: 0.6 M words/sec.
- Data Channel and AV24 communications port connected via CR8071 MIA.
- System control functions included: C-bus and P-Bus Arbitration, Clock generation, 250 us Fast Timer, Real time clock, Power failure detection, Master Clear, PU Disable, Maintenance Mode.
- MTBF: 19400 hours

(CONTINUED)

PROGRAMMERS REFERENCE:

SWITCH AND STRAP DEFINITIONS:

S1: push: Master Clear. Inhibited by SR5 while PU is enabled.

S2: up (MAIN): Maintenance Mode. Inhibited by SR4 while PU is enabled.

down (NRM): Normal Mode

S3: up (DIS): PU Disable

down (EN): PU Enable. Inhibited while "INH" LED is on if SR1 is not mounted.

S4: PU number (0 - 15):

Front Panel ←

4	3	2	1	OPEN = 1
N_3	N_2	N_1	N_0	ON = 0

N_0 = LSB

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- AV24 Baud Rate: max. 9600 baud.
  Buffer size: max. 512 bytes.

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.

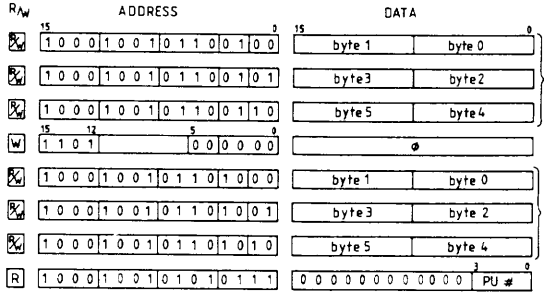
- Mechanical Dimensions:
  H: 412 mm
  W: 17.1 mm
  L: 305 mm

- Weight: 1.4 kg

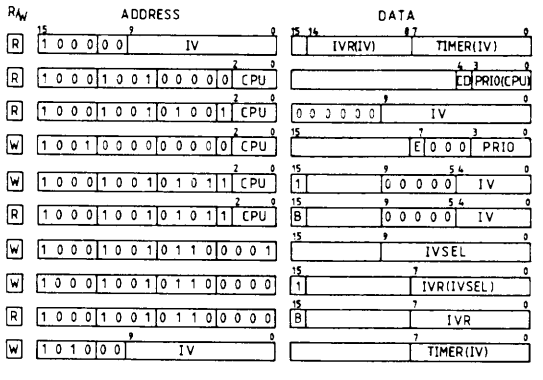
- Power Consumption:
  +5V: 6A from P1 + 7A from P2
  +12V: 1mA from P1
  -12V: 4mA from P1

PROGRAMMERS REFERENCE: (CONTINUED)

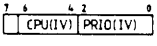
System Control Functions:



Interrupt Processing Control:



IVR layout:



CD = 0: CPU enabled for notification
CD = 1: CPU disabled

E = 1: CD(CPU) is cleared
E = 0: CD is unchanged

B = 1: MAP is processing instruction
B = 0: MAP is ready for a new instruction

TIMER(IV) is in units of 1024 ms.

(CONTINUED)

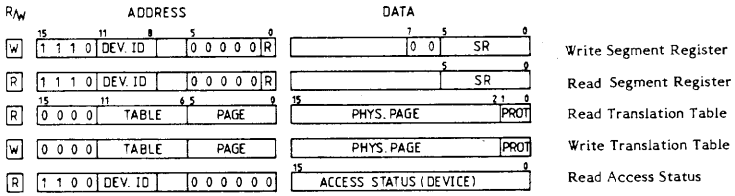
PROGRAMMERS REFERENCE: (CONTINUED)

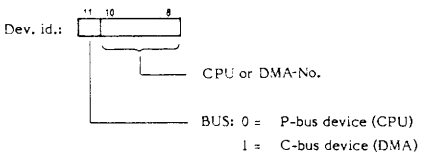
Interrupt Vector (IV) Definitions:

	IV			
	CA	MA	IV #HEX	Description of IV
bit #	9 8 7 6	5 4 3 2 1 0		
	n	1 1 1 1 1 1		Power failure in Crate #n, (PU or CU)
	0 0 0 0	1 1 1 1 1 0	03E	Real time interrupt
	0 0 0 0	0 X X X X X	000-01F	CPU interrupt
	0 0 0 0	1 1 1 1 0 1	03D	Intra memory DMA in MAP
	0 0 0 0	1 1 1 1 0 0	03C	AV24 Communication port in MIA
	REMAINING			Available for other modules

Memory Address Translation:

Instruction:





R (Register Select): 0 = 'Program' (P-Bus) or 'Read' (C-Bus)
1 = 'Data' (P-Bus) or 'Write' (C-Bus)

Prot.:

Bit 1	Bit 0	P-Bus Syst. State	P-Bus User State, C-Bus
0	0	Page absent	Page absent
0	1	Full access	Full access
1	0	Full access	Read only
1	1	Full access	No access

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Access Status Register:

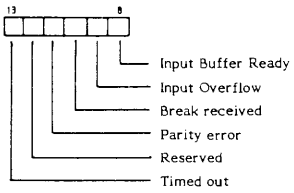
Bit 1 - 0:	Access protection bits (invalid for privileged read/write)	
Bit 2:	Parity error in lower TT-byte	
Bit 3:	Parity error in upper TT-byte	
Bit 4:	Reserved	
Bit 5:	Reserved	
Bit 6:	Reserved	
Bit 7:	Time out detected by MAP	
Bit 8:	Reserved	Error Status from Channel Unit
Bit 9:	Address Register error	
Bit 10:	Parity error during write	
Bit 11:	Parity error during read	
Bit 12:	CU-error (bit 11-8 valid)	
Bit 13:	Parity error in upper byte from Data Channel	Set by MIA
Bit 14:	Parity error in lower byte from Data Channel	
Bit 15:	Time out on Data Channel	

AV24 Port Control:

R/W	ADRESS	DATA	
R	1000 1001 1101 0000	B STATUS	Read input status
W	1000 1001 1101 0000	1 0000000000000 COM	Write input command
R	1000 1001 1101 0001	ISIZE	Read ISIZE (bytes)
R	1000 1001 1101 0010	ISTART	Read ISTART (bytes)
W	1000 1001 1101 0011	BLOCKSIZE	Write BLOCKSIZE (bytes)
W	1000 1001 1101 0100	TIME	Write TIME (ms)
W	1000 1001 1101 0101	DIS 0 CHAR	Write STOPCHAR
W	1000 1001 1101 0110	MODE	Write MODE
R	1000 110 LOC	byte(LOC×2+1) byte(LOC×2)	Read input data
R	1000 1001 1101 1000	B E	Read output status
W	1000 1001 1101 1000	1 0000000000000 COM	Write output command
W	1000 1001 1101 1001	OSIZE	Write OSIZE (bytes)
W	1000 1001 1101 1010	OSTART	Write OSTART (bytes)
W	1000 111 LOC	byte(LOC×2+1) byte(LOC×2)	Write output data

B (Busy): 1 = MAP is processing command
0 = MAP is ready for a new command

Input Status:



(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Input commands: 0: Get Data
1: Clear Buffer
2 - 7: Illegal

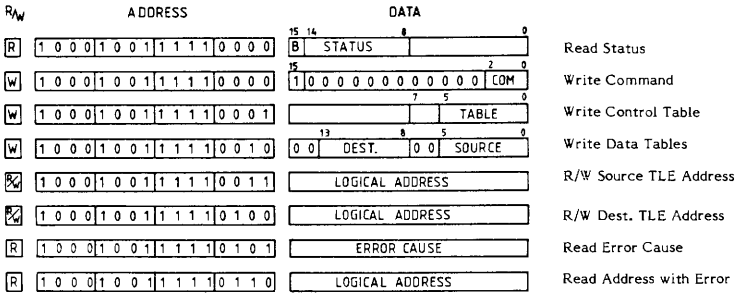
DIS = 0: STOPCHAR enabled
DIS = 1: STOPCHAR not used

Modes: 0: Normal
1: Auto echo

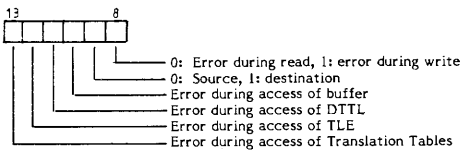
Output Status: E = Output buffer empty

Output Commands: 0: Xmit data
1: send BREAK
2 - 7: Illegal

DMA instructions:



Status:



Commands: 0: Start Transfer
1: Cancel Transfer

Error Cause: As 'Access Status Register',
but bit 6 = Parity error detected by MAP

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Maintenance Mode Commands:

R

Run - Bus authority enabled

H

Halt - Bus authority disabled

S

Single - Bus authority enabled for one cycle

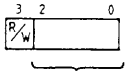
Nc

Notify - c is one digit: 0, 1, 2, 3, 4 or 7.

M $a_5a_4a_3a_2a_1a_0$ $n_3n_2n_1n_0$
$d_3d_2d_1d_0$ $d_3d_2d_1d_0$ .....

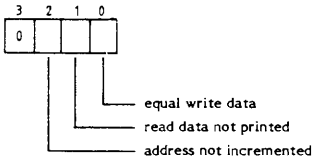
Move - Data are read to or written from the console.
a, n, d are hex-digits.

a_5:



0: I/O
1: MAP
2: Memory, lower byte
4: Memory, upper byte
6: Memory, word

a_4:



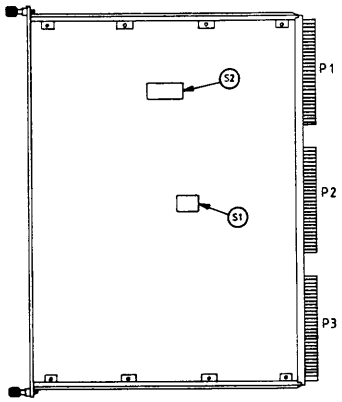
$a_3a_2a_1a_0$: start address. For memory: logical address.
$n_3n_2n_1n_0$: number of locations
$d_3d_2d_1d_0$: write data. Only one word, if 'equal write data'.

Output: * - Ready
? - Error
$d_3d_2d_1d_0$ - four hex-digits: read data from M-command
_ - request for parameters.

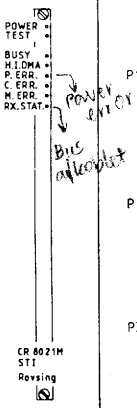
DATASHEET FOR:

STI CR8021M/010-C/00

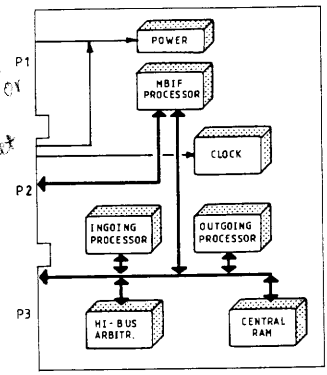
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Interface from the CR80 Computers to the TDX-system (via a TIA) and to the Supra Bus (via a SBA).
- Interface control RAM accessible by privileged instructions.
- Module address and interrupt priority set by S2.
- HOST address set by S1.
- MTBF = 32,800 H/F.
- The STI has a standard C-Bus connector towards the CR80, and a I/O connector for a 64 pin flatcable bus, allowing a maximum number of 8 TIA's or SBA's to be connected to the STI. Data for communication is moved by DMA.
- The Ingoing processor moves data from the TIA or SBA to the CR80 memory. The Outgoing processor moves data from CR80 memory to the TIA or SBA. The data-traffic is controlled by the SUPRA/TDX-packet protocol.

PROGRAMMERS REFERENCE:

S2

1	2	3	4	5	6	7	8
A0	A1	A2	A3	A4	P0	P1	P2

Front Panel

S1

4	3	2	1
D3	D2	D1	D0

Module Address Setting:

Address determined by A0-A4; A4:MSB
0~Switch ON, 1 ~ Switch OFF.

(CONTINUED)

- Power Consumption:

  +5V : 6.5A +/- 0.6A
  +12V : 40mA +/- 4mA
  -12V : 3mA +/- 0.3mA

- Mechanical Dimensions:

  Height: 375,5mm
  Width: 17.2mm
  Length: 304mm
  Weight: 1.2 kg

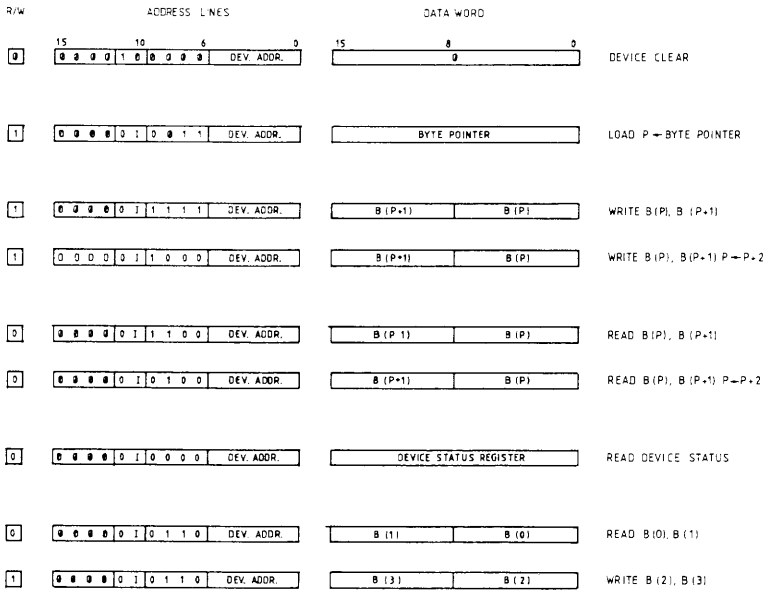
PROGRAMMERS REFERENCE: (CONTINUED)

Interrupt Priority:

The priority determined by P0-P2; P2:MSB
0~Switch ON, 1 ~ Switch OFF.

Memory Address:

0000H-0FFFH:	Program memory
1000H-1FFFH:	Program memory
2000H:	Interrupt to MBIF processor
2400H:	HOST address in
2800H:	Host address out
3000H-37FFH:	RAM memory
6000H-7FFFH:	Adapter memory
8000H-FFFFH:	Central RAM



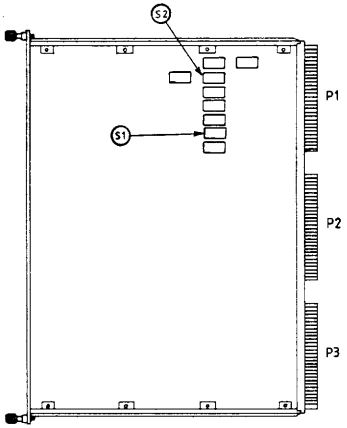
NOTES

1 BIT 10 (I-FIELD) INDICATES THAT THE μ-PROCESSORS IN HOST I/F ARE INTERRUPTED

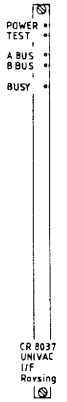
2 THE BYTE POINTER POINTS TO A BYTE IN CENTRAL RAM

3 P IS THE CONTENTS OF THE BYTE POINTER REGISTER
B(P) IS THE CONTENTS OF THE LOCATION POINTED OUT BY P

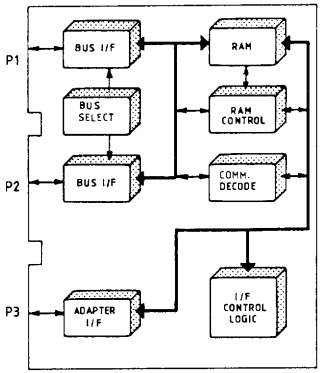
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Interface from CR80 to a UNIVAC series 1100 ISI input/output channel pair, combined with 16K memory. Used in connection with the CR8079M UNIVAC I/F Adapter.
- Control/Status memory section accessible by privileged instructions only.
- Compartmentalized RAM is part of normal CR80 memory address space of A and B bus, and shared with the peripheral module internal controller processor.
- Dual bus ports.
  (Single bus version CR8037M/010A-/00)
- Data transfer rate to/from UNIVAC: 1 Mchar/sec. (max).
- Module address set by S1.
- RAM base address set by S2.
- Power Consumption:
  +5V: 7A
  +12V: 350mA
  -12V: 10mA

(CONTINUED)

PROGRAMMERS REFERENCE:

S1

8	7	6	5	4	3	2	1
P1	P0	M5	M4	M3	M2	M1	M0

Front Panel ←

S2

6	5	4	3	2	1
A5	A4	A3	A2	A1	A0

Module Address Setting:

Address determined by M5-M0; M5:MSB
Ø~Switch ON, 1~Switch OFF.

Interrupt Priority Setting:

Priority determined by P1-P0; P1:MSB.
Ø~Switch ON, 1~Switch OFF.

RAM Base Address Setting:

Base address determined by A5-AØ; A5:MSB.
RAM occupies addresses.
B*16K to (B+1)*16K-1, B:A5-AØ.
Ø~Switch ON, 1~Switch OFF.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Size:
  Length: 305.0 mm
  Height: 412.0 mm
  Width: 17.1 mm

- Weight: 1.1 kg

- MTBF: 33,200 H/F

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 1 sec. if it has been carried out successfully.

PROGRAMMER'S REFERENCE: (CONTINUED)

I/O Commands:

(AD5-0 is module address LS1-0 is 00).

R/W		AD11-6		
0		00X	XXX	SENSE BUSY STATUS
1	or	01X	XX1	SET RECEIVE BUSY
1	or	11X	X1X	SET SEND BUSY
X		100	XXX	DISABLE I/F, CLEAR
X		101	XXX	ENABLE I/F, CLEAR

X = DON'T CARE

On SENSE BUSY STATUS, data lines 3-Ø are:

DA3:	UNIVAC I/O CLEAR
DA2:	CLEAR REQUEST FLAG
DA1:	SEND BUSY FLAG
DA0:	RECEIVE BUSY FLAG

Other data lines are Ø.

Command Buffer:

The command buffer occupies RAM words ØØ to $ØE_x$ (relative to RAM base address). The layout is:

15	8	7	0
0	SEND STATUS	SEND ERROR FLAGS	
1	reserved	SEND CONTROL FLAGS	
2	SEND BUFFER ADDRESS		
3	SEND BUFFER LENGTH		
4	SEND EF-Word (upper half)		
5	SEND EF-Word (lower half)		
6	RECEIVE STATUS	REC. ERROR FLAGS	
7	reserved	REC. CONTROL FLAGS	
8	RECEIVE BUFFER ADDRESS		
9	ADDRESS OF FIRST FREE WORD IN REC. BUFFER		
A	REC. EF-Word (upper half)		
B	REC. EF-Word (lower half)		
C	REC. PARITY ERROR POINTER		
D	REC. CHECKSUM FLAGS		
E	REC. BUFFERSIZE		

Receive process statusword

15	status inf.	8	7	error flags	0

STATUS:

(a) Receive Busy flag set:
00_x: BUSY
(b) Receive Busy flag cleared:
00_x: IDLE
01_x: READY
02_x: CLEARED

ERROR FLAGS: (normally zero)

(a) READY status:
bit 7 not used
bit 6 bad parameters
bit 5 buffer overflow
bit 4 checksums bad
bit 3 EF-word parity error (S/W)
bit 2 EF-word parity error (H/W)
bit 1 Data parity error
bit 0 Memory parity error

(b) CLEARED status:
bit 7 adapter error (upper byte)
bit 6 adapter error (lower byte)
bit 5 not used
bit 4 memory parity error
bit 3 memory check 1
bit 2 memory check 2
bit 1 memory check 3
bit 0 memory check 4

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

SEND process statusword:

15 status inf. 8	7 error flags 0

STATUS:

(a) SEND BUSY flag set:
00_X: BUSY
(b) SEND BUSY flag cleared:
00_X: IDLE
01_X: READY
02_X: CLEARED

ERROR FLAGS: (normally zero)

(a) READY status:
bit 7 not used
bit 6 not used
bit 5 not used
bit 4 not used
bit 3 not used
bit 2 bad parameters
bit 1 memory parity error - no transmission
bit 0 memory parity error during transmission

(b) CLEARED status as for RECEIVE.

RECEIVE CONTROL FLAGS:

bit 0: 0: Calculate checksums
1: Calculate, modify and check checksums

bit 1: 0: No check of EF-Word S/W parity
1: Check EF-Word S/W parity

SEND CONTROL FLAGS:

bit 0: 0: No checksum insertion
1: Insert checksums

bit 1: 0: No modification of EF-Word
1: Insert S/W parity bit in EF-Word

To Output One Buffer:

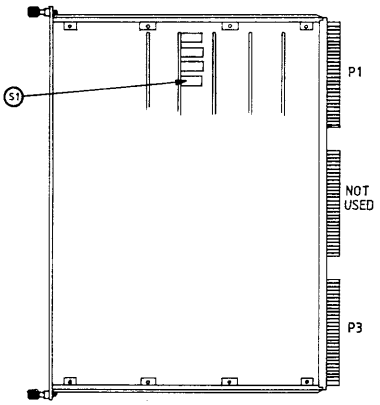
(a) SEND must have IDLE status.
(b) Initialize: CONTROL FLAGS
BUFFER ADDRESS
BUFFER LENGTH
EF-Word
(c) Set SEND BUSY flag.
(d) SEND interrupts with READY status after completion. Error flags should be zero. Set to IDLE.

To Input One Buffer:

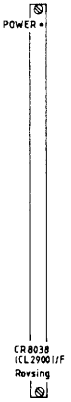
(a) RECEIVE must have IDLE status.
(b) Initialize: CONTROL FLAGS
BUFFER ADDRESS
BUFFER SIZE
(c) Set RECEIVE BUSY flag.
(d) RECEIVE interrupts with READY status after completion. Error flags should be zero. Set to IDLE.

ICL 2900 I/F CR 8038M/010A-/00

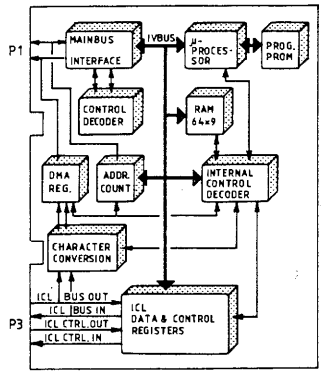
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Designed to interface between the CR80 computer and an ICL 2900 NRPI, using the ICL I/F ADAPTER CR8077
- 16 bits DMA to/from CR80 memory
- Parity checked and generated.
- 4 sets of character conversion tables defined by firmware. Selected by software.
- Mechanical dimensions:

Height:	412 mm
Width:	17.2 mm (1M)
Length:	305 mm
Weight:	1.2 kg

- Power consumption:

+5V 3A Typ	Incl. power to
-12V 200mA Typ	ICL I/F ADAPTER

- MTBF: 59 500 hours

PROGRAMMERS REFERENCE:

Switch Orientation and Definitions

	8	7	6	5	4	3	2	1	
S1	P1	P0	A5	A4	A3	A2	A1	A0	Front Panel ←

Module Address Setting

Address determined by A5-A0 (A5 = MSB) OPEN = "1".

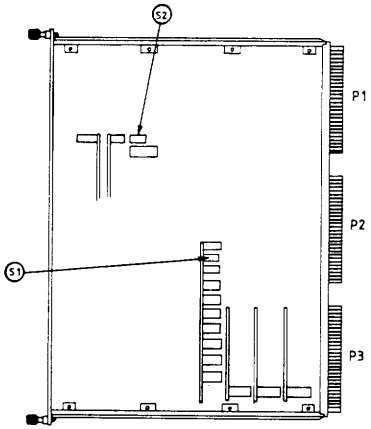
Interrupt Priority

The interrupt priority is determined by P0, P1 (P1 = MSB) OPEN = 1.

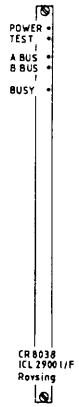
DATASHEET FOR:

ICL 2900 I/F Dual Bus CR 8038M/040AB/00

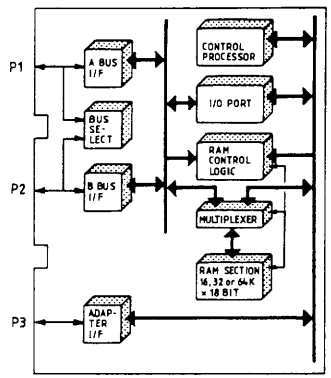
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- ICL 2900 I/F combined with 32K memory.
- Dual Bus port
- Control/Status memory section accessible by privileged instructions only.
- Compartmentalized RAM is part of normal CR80 memory address space of A and B bus, and shared with the peripheral module internal controller processor.
- Connection of up to 8 ICL 2900 via ICL 2900 I/F adapters.
- Data transfer rate:
  To/from ICL M/F: 400K bytes/sec.
- Conversion of data received from/transmitted to ICL 2900 from any 8 bit code to any other specified code.
- Power Consumption:

+5V:	6.5A
+12V:	0.3A
-12V:	0.1A

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definitions

S1

6	5	4	3	2	1
A5	A4	A3	A2	A1	A0

Front Panel

S2

8	7	6	5	4	3	2	1
P1	P0	M5	M4	M3	M2	M1	M0

Memory Address Space determined by
A1-A5; A1 LSB, A5 MSB
A0 don't care

Module Address and Priority determined by
M0-M5, P0-P1; M0 LSB, P1 MSB.

The module consists of a Controller part and a memory part. These are connected internally for data transfer between ICL M/F and memory concurrent with main bus access to the memory.

The controller is built around a high speed programmable controller. It receives commands and delivers status via its own bus control logic, whereas instruction and status concerning ICL M/F operations are fetched/stored in a command buffer consisting of seven 16 bits registers. The controller interprets instructions stored in the command buffer, controls and monitors the link to the ICL M/F and performs the byte oriented data transfer.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Mechanical Dimensions:
  Length: 305.0 mm
  Height: 412.0 mm
  Width: 17.1 mm (1M)

- Weight: 1200 g

- Mean Time between Failure:
  $36 * 10^3$ hrs.

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.

PROGRAMMERS REFERENCE: (CONTINUED)

The ICL 2900 I/F accepts the following I/O commands:

R/W	AD11-6 Command Code	Command
0	000 b2b1b0	Read Command Buffer, location b2b1b0
1	000 b2b1b0	Write Command Buffer, location b2b1b0
X	100 XXX	Disable ICL 2900 I/F, Reset
X	101 XXX	Enable ICL 2900 I/F, Reset

Command Buffer Layout

FUNCTION	Loc. b2b1b0	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
	0	Not Used
INSTRUCTION	1	0 0 0 AA2 AA1 AA0 CC1 CC0 PPOS CDOI CDRW BSC 1 BCRI BRC
MEMORY START ADDRESS	2	MA0-15
BYTE COUNT	3	BCT0-15
PRIMARY STATUS	4	Not Used — O O ATTN TRMB SHB O O O O
COMMAND WORD	5	Not Used — OEC MPE DPE ICL Command
DATA BYTE/ INFORMATION BYTE	6	DB0-7 — Always "0" — INCB8
SELF TEST STATUS	7	Always "0" — MER AEB CBE

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Instruction Word

BRC, Busy/Ready CR80 :
"0": 'End of Action' serivced by CR80
"1": 'End of Action' waiting to be serviced.

BRI, Busy/Ready I/F:
"0": I/F ready to receive new command
"1" I/F busy executing last received command

IC, Interrupt Control:
"0": Interrupt is generated as 'End of Action'
"1": No interrupt is generated as 'End of Action'

BS, Byte Swap:
"0": Lower byte of data to/from memory is used first
"1": Upper byte of data to/from memoryis used first

DRW, Diagnostic Read/Write:
On board memory and registers are checked for proper operation

CDI, CR80 Data In:
Data is transferred from memory to ICL M/F

CDO, CR80 Data Out:
Data is transferred from ICL M/F to memory

PPS, Present Primary Status:
The Primary Status byte is transferred to the ICL 2900. If the operation involves data transfer, Primary Status is sent when the data transfer is finished.

CC0-1, Conversion Code:

CC1	CC0	Function
0	0	No Conversion
0	1	Conversion Table 1
1	0	Conversion Table 2
1	1	Conversion Table 3

AA0-3, Adapter Address:
Used to address 1 of 8 possible adapters

Primary Status

SHB: Short Block
TRM: Terminated
ATT: Attention

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Command Word:

ICL Command:

Code (Hex)	Command
0	Initialize Deck No. 0
1	Initialize Deck No. 1
2	Initialize Deck No. 2
3	Initialize Deck No. 3
4	Initialize Deck No. 4
5	Initialize Deck No. 5
6	Initialize Deck No. 6
7	Initialize Deck No. 7
8	Read
9	Write
A	Sense
B	Identify
C	Limit
D	Send Property Code
E	Rewind
F	Error
10	End of Block
11	End of Block with Error
12	Data Error

DPE, Data Parity Error:

A parity error has been detected in data transferred from ICL 2900.

MPE, Memory Parity Error:

A parity error has been detected in data fetched from the memory.

OEC, Other Error Conditions:

Other errors than those covered by DPE and MPE have been detected (Command buffer parity error, invalid instruction word, power failure on adapter).

Information Byte:

INC, Invalid Command:

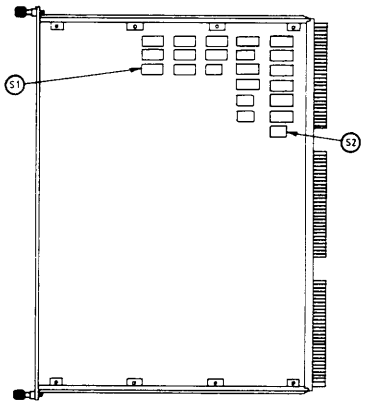
Used when receiving an erroneous command from the ICL M/F.
DB0-7 and CB8 are loaded with the state of the signals DFX0-8 received from the ICL M/F in case of an erroneous command.

Self Test Status:

CBE: Controller Board Error
ABE: Adapter Board Error
MER: Memory Error (only used in connection with a Diagnostic Read/Write Command).

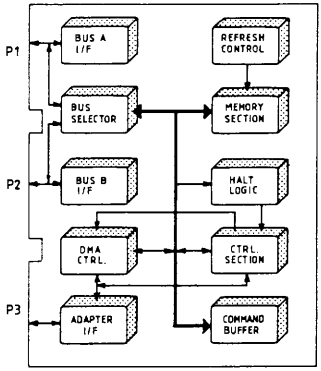
IBM CH. I/F CR8039M/040AB/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Operates on an IBM 360/370 Block Multiplexer Channel.
- Control Unit simulation under software control.
- Compartmentalized RAM is part of normal CR80 memory address space of A and B bus, and shared with the peripheral module internal controller processor.
- Odd parity checked and generated.
- Dual bus ports.
- Module address and interrupt priority set by switch S1.
- Memory address space set by switch S2.
- Memory transfer rates:
  To/from IBM Channel: 200K bytes/sec.
- Size:

Length:	305.0 mm
Height:	412.0 mm
Width:	17.1 mm

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definitions

S1

8	7	6	5	4	3	2	1
P1	P0	A5	A4	A3	A2	A1	A0

Front Panel ←

S2

6	5	4	3	2	1
B5	B4	B3	B2	B1	B0

Module Address Setting:

Address determined by A5-A0, A5: MSB, 0=switch ON, 1=switch OFF. Correspond to the Main Bus address bits AD5 - AD0.

Interrupt Priority Setting:

Priority determined by P1-P0, P1: MSB.

Memory Address Space Setting:

Address space determined by B5-B0, B5: MSB. Correspond to the Main Bus address bits AD19 - AD14. Memory allocation within the one Mega word memory space: P*16K to (P+1)*16K-1, P: B5-B0.

Memory Access

Memory access from the Main Bus is possible when LS1-0 ≠ 00, and AD19-14 match the setting of S2. RAM address is set by AD13-0, and read/write by R/W(L/H).

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Weight: 1.1 kg
- Power Consumption:
  +5V: 7A (typic)
- MTBF: 32400 (H/F)
- Available in single bus version
  CR8039M/010A-/00
- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has carried out successfully.

PROGRAMMERS REFERENCE: (CONTINUED)

Commands

Commands are decoded when LS1-0 = 00, and AD5-0 match the setting of S1. Commands are decoded from AD11-6, R/W(L/H):

R/W(L/H)	AD11-6	Command
1	100 000	Disable IBM CH. I/F, Reset
1	101 000	Enable IBM CH. I/F, Reset
	else:	
0	b5b4...b0	Read Command Buffer loc. b5b4...b0
1	b5b4...b0	Write Command Buffer loc. b5b4...b0

Data Format of Commands:

DA0-7: Read/Write word in Command Buffer
DA8: Read position of ON LINE/OFF LINE REQUEST (L/H) switch (on the IBM Adapter).
DA9: Read value of Operational Out (L) from the IBM Channel
DA10: Read value of Metering Out (L) from the IBM Channel.
DA11: Read value of Parity Error (L), set at previous I/O write command. If Parity Error is set, the previous write command has been ignored (contents of Command Buffer is unchanged).

Command Buffer Layout

HEX Addr	Mnemonic	Name	Type	R/W
0	INSTR	Instruction Word	IN	R/W
1-2	DMADL, DMADM	DMA Address LSNs, MSBs	IN	R/W
3-4	BCNTL, BCNTM	Byte Count LSBs, MSBs	IN	R/W
5	STAAD	Status Address MSBs	IN	R/W
6	ENDSTA	End Status	IN	R/W
8	REQIN	Req. Instruction Word	RQ	R/W
9	REQAD	Req. Device Address	RQ	R/W
A	REQSTA	Req. Status	RQ	R/W
10	IBM COM	IBM Command	IN	R
11	IBM AD	IBM Address	IN	R
12-13	CONT 1,2	Control Word one, two	IN	R
15-16	UCNTL, UCNTM	Updated Byte Count LSBs, MSBs	IN	R
23-1c	COM0-5	Command 0-5	IN	R/W
21-26	STAIN0-5	Status Index 0-5	IN	R/W
29	STACK	Stack	TS	R
2A	BUSY	Short Busy Status	TS	R
2B	CMASK	Command Mask	IN	R/W
2D	ILLSTA	Illegal Command Status	IN	R/W
39	HWER	H/W Error Word	TS	R
3A	BYCNT	Short Busy Seq. Counter	TS	R/W
3B	AUTSTA	Auto Status	TS	R

IN: Registers used at Initial Selection
RQ: Registers used at Service Request
TS: Registers for test purposes

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Special Bit Definition:

Instruction Word	DA7:	Instruction Loaded
	DA6:	OFF LINE
	DA5:	Reset
	DA2:	Data Out
	DA1:	Data In
	DA0:	Present End Status
Control Word One	DA6:	Available
	DA5:	Command Chaining
	DA3:	Stop
	DA2:	Finish
	DA1:	Disconnect
	DA0:	Non Zero Status
Control Word Two	DA7:	Bus Out Check
	DA6:	Data Check
	DA4:	System Reset
	DA3:	Selective Reset
	DA1:	Reset Done
	DA0:	Instruction Error
Req. Instruction Word	DA6:	(Busy)
	DA3:	Set Busy
	DA2:	Present Req. Status
	DA1:	(Suppressible)
	DA0:	Service Request
Stack	DA2:	END STA (/INITSTA)
	DA0:	Auto Request
H/W Error Word	DA2:	Busin Error
	DA1:	RAM Error 1
	DA0:	RAM Error 2

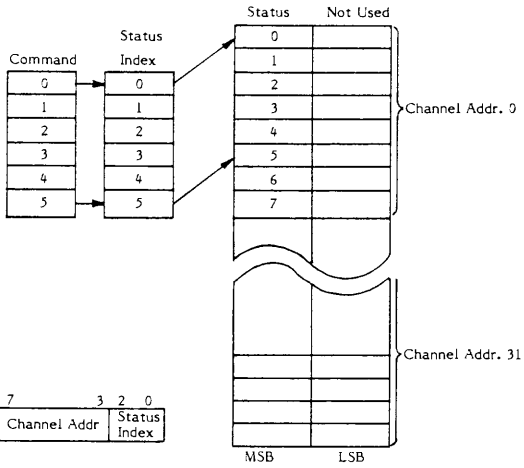
(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Status Table

The Status Table (256*16 bit words) is placed in the 16K memory and contains status information for all device addresses.

Format:



Status Address:

Note:

1) Before a Channel Command is compared with COM0-5, unnecessary bits are removed by masking with CMASK.

2) Command compare starts with COM0.

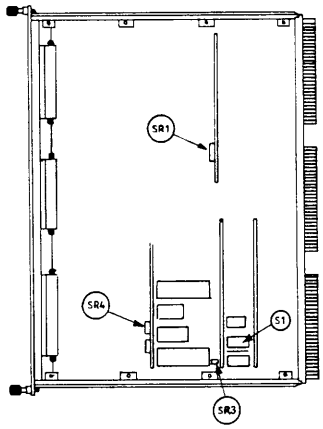
3) ILLSTA will be presented for the channel if the command is illegal.

4) CR80 is free to update the Status Table, while the IBM I/F performs a data transfer or while the I/F has a CR80 interrupt pending. If an Initial Selection can occur during updating, the I/F must be set Busy by CR80 (Req. Instruction Word).

DATASHEET FOR:

WPC CR8040M/0X0AB/00

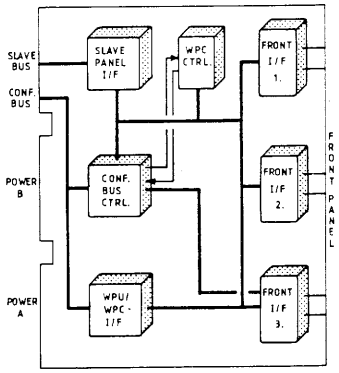
SET UP REFERENCE



FRONT PANEL

See next page

BLOCK DIAGRAM



CHARACTERISTICS:

Watchdog Panel Controller

CR8040M/010AB/00
WPC including standard front panel
CR8040M/020AB/00
WPC exclusive front panel

- The Watchdog Panel CTRL. is an optional part of the total Watchdog System.
- It has 2 main modes of operation: Manual or Auto.
- In Automode the WPC is able, on WPU request, to display on its front panel WPU status.
- In manualmode where the WPC has configuration CTRL. it performs the WPU task of switching LIA's by means of front panel switches. Feedback of switching, done by the CCA's, is checked and displayed.
- Serial interface to Configuration Bus.
- Display of system status on front panel.

(CONTINUED)

PROGRAMMERS REFERENCE:

	LSB							MSB
S1	1	2	3	4	5	6	7	8
	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7

FRONT PANEL →

Switch S1 has the single purpose, of informing the WPC-firmware of the number of CCA's (SCA's) connected to the Configuration Bus. The number is set in binary code.

The following jumpers are included:

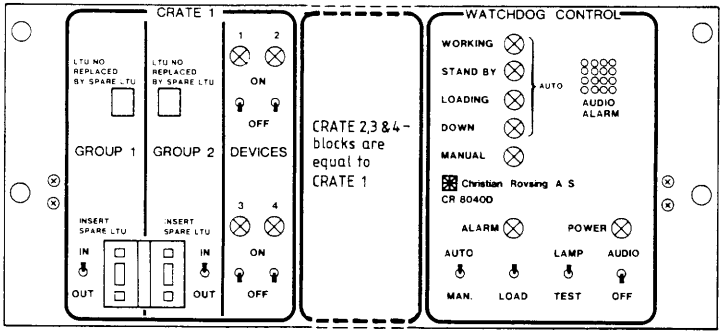
SR1: Short circuiting these two pins resets WPC.

SR3: Inserted - 8031 processor and external EPROM is used.
: Omitted - 8751 processor and no EPROM is used.

SR4: Pos.A - 8K EPROM
: Pos.A - 4K EPROM.

CHARACTERISTICS: (CONTINUED)

- Failsafe switchover to WPC manual mode and alarm indication in case of WPU break-down.
- Power supply for the WPC is dualized, i.e. both A- and B-Bus applies the power.
- Either standard WPC-front panel or customized front panel can be used. Layout and functions of the standard front panel is shown below:



- Power Consumption:

+5V	2A	(without front panel)
+5V	3.5A	(with standard front panel)
+12V	0.35A	

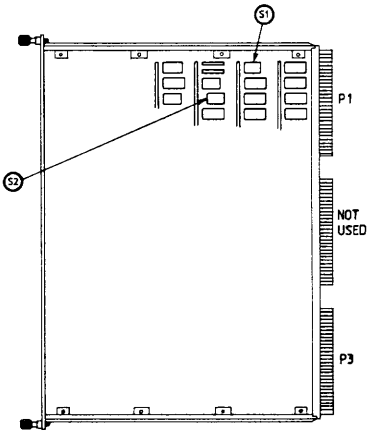
- Mechanical Dimensions:

Height:	443.0mm
Width:	Standard front panel: 6m or 103.2 mm
	Customized front panel: according to design.
Length:	305,0mm

- MTBF: 40,000 (H/F)

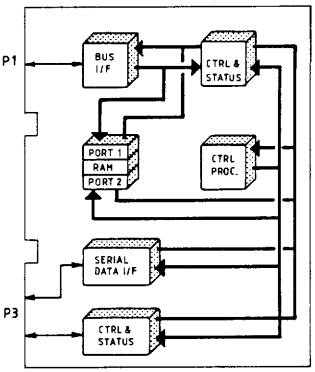
DISK CTRL. CR8044M/01XA-/XX

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



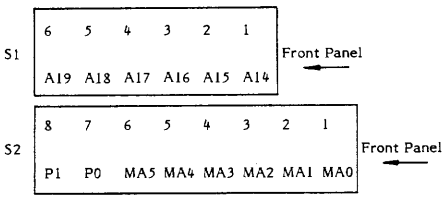
CHARACTERISTICS:

- Disk Controller/Formatter combined with 16K, 32K or 64K memory for CR80 system.
- Control/Status memory section accessible by privileged instructions only.
- Compartmentalized RAM is part of normal CR80 memory address space of A and B bus, and shared with the peripheral module internal controller processor.
- Interfaces to the disk adapter CR8084 for connection of 1 to 4 disk drives.
- Any combination of drives from CDC's SMD, MMD, and CMD families is possible.
- The dual channel feature of the disk drives is supported.
- Overlayed seeks supported.
- Main disk operations are:
  Seek, Read, Write, Seek and Read, Seek and Write, Format, Read address field.

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definition:



RAM Address Switch Setting:

The address switch S1 determine the bottom address of the address space occupied by the RAM.
16K RAM: all switches are valid
32K RAM: A14 is a 'don't care'.
64K RAM: A14 and A15 are 'don't cares'.
A19 is address bit 19 and MSB.

Module Address Switch Setting:

The switch S2 determine the CR80 module address and the interrupt priority.

P1-P0: interrupt priority P1: MSB.
MA5-MA0: module address MA5:MSB

0~switch ON, 1~switch OFF

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- RAM write response time: 350 ns $<T_{RWmin}<470$ns if RAM is not occupied by the controller, e.g. for refresh.
  Otherwise, $T_{RWmin}<T_{RWmax}<T_{RWmin}+440$ ns

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.

- Dimensions:

Width:	17.1 mm
Length:	326.0 mm
Height:	412.0 mm
Weight:	1.4 kg

- Power (including Disk CTRL Adapter) Consumption:

	16K RAM	32K RAM	64K RAM
+12V	15(600)mA	30(1000)mA	0mA
+5V	7A	7A	7A (8.1A)
-12V	160mA	160mA	150mA

  The figures in parentheses are max. current at max. speed operation in the RAM. The other figures are typical values with no RAM operation but refresh.

- Versions of CR8044M/01XA-/XX
  X = 0 16K RAM
  X = 1 32K RAM
  X = 2 64K RAM
  XX = 01 No built-in test. Chained buffers
  XX = 02 Built-in test. Chained buffers or buffer table.

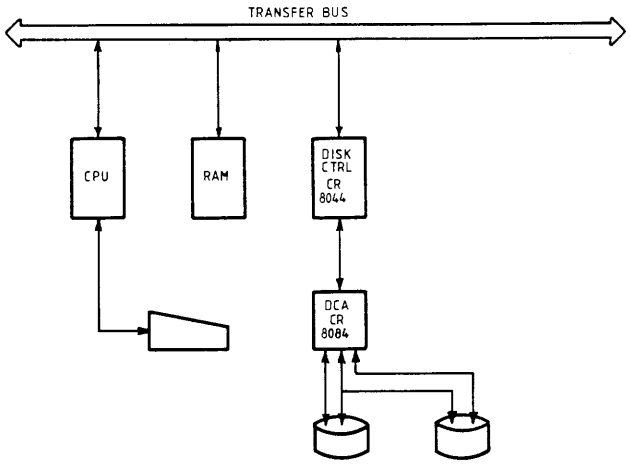
- RAM read response time $T_{RR} = T_{RW} + 50$ ns.

- 5% occupation of RAM due to refresh.

- Controller response time T_{CR}: 290ns $<T_{CR}<415$ ns.

- MTBF: 39,400 h

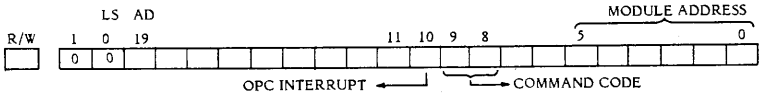
PROGRAMMERS REFERENCE: (CONTINUED)



CR8044 in single bus CR80 Minicomputer

COMMANDS

Commands:



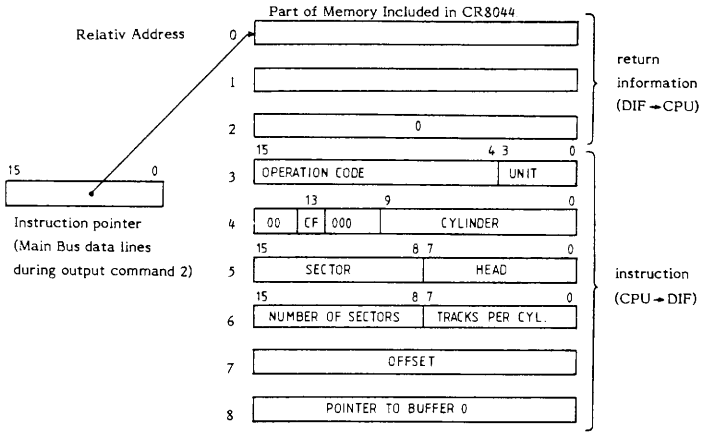
COMMAND CODE

R/W	Command Code	Command
0	0 - 3	read status word, clear interrupt
1	0	reset
1	1	load unit interrupt mask
1	2	load instruction pointer, initiate operation
1	3	terminate operation

(CONTINUED)

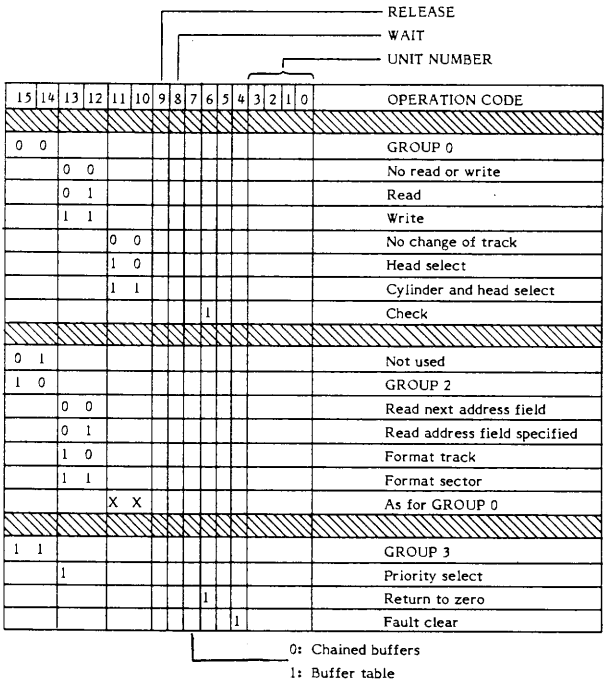
PROGRAMMERS REFERENCE: (CONTINUED)

INSTRUCTION FIELD



This does not apply to format.

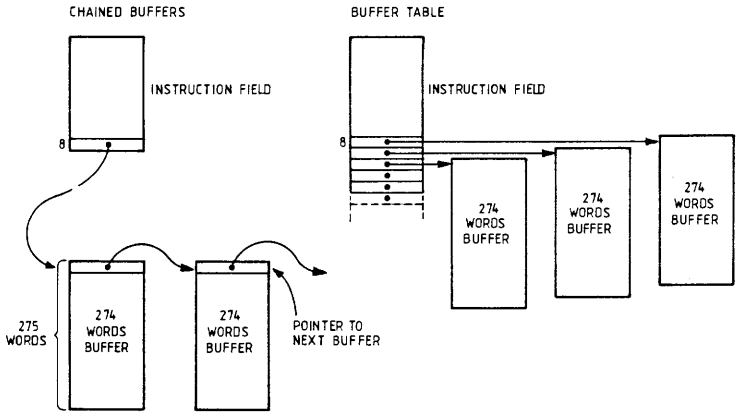
OPERATION CODE



(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

BUFFER POINTERS:

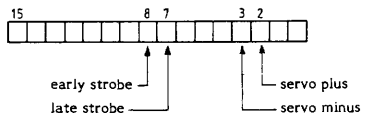


Only the last buffer pointer can be 0.

OBS: Firmware application XX = 01 will only handle chained buffers, while XX = 02 handles both buffer structures.

OFFSET:

Word 7 of instruction field



FORMAT AND READ ADDRESS FIELD:

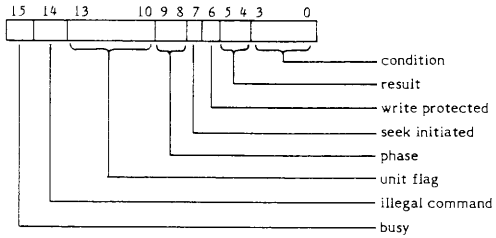
FORMAT		modifications of instruction field									read address field	
		15	14	13				9	8	7	0	
relativ address	4	B	W	CF				cylinder			0	relativ address
	5	sector							head		1	
	6	auxiliary bytes									2	

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

STATUS

Status Word (Input Command 0, 1, 2 or 3)



Condition Codes

0. Busy
1. Was seeking
2.
3. Unexpected drive status
4. Check or Sync data field
5. Check or Sync address field
6. Address field
7. Buffer pointer
8. Illegal sector
9. Timing error
10. Memory overrun
11. Parity error
12. Selftest running
13. Reset. Selftest succeeded.
14. Selftest failed
15. Group 1 operation

Bit 9-8 phase description:

0 0	0	unit selection, status check, waiting for a busy drive to get free.
0 1	1	initial seek
1 1	2	searching for the first sector to be handled
1 0	3	read, write or format running

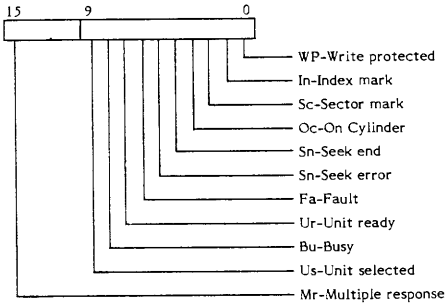
Result: Only valid when bit 15 is "0". Describes the latest operation:

1 0	Normal ending
0 1	Terminated by the command "Terminate operation" from CPU.
0 0	Terminated because of the condition indicated by bits 3 - 0.

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Re: 3. Unexpected drive status. Some of the 11 drive status lines had some contents not allowed at the time they were checked. The status signals are stored in word 0 of the instruction field:

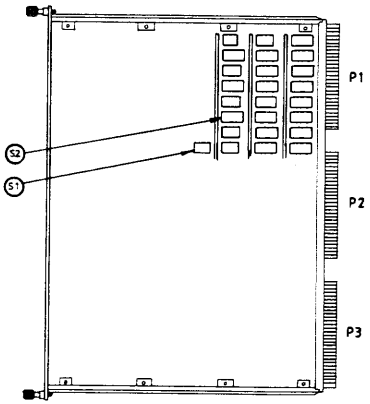


The signal Mr is generated by the Disk Adapter CR8084. A "1" indicates that wrong drive(s) responded to the unit selection. Word 1has a "1" for those bits in word 0 that are not as expected.

Volume Organization:

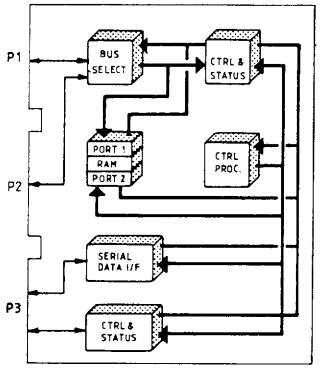
A disk volume is divided into a number of cylinders and tracks determined by head position and number. Each track is divided in 32 sectors, which contain an address field with full address of the sector and write protect, bad mark bits and 2 auxiliary bytes. A data field, which contains 548 bytes user data, is following the address field.

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Disk Controller/Formatter combined with 16K, 32K or 64K memory for CR80 system.
- Control/Status memory sections accessible by privileged instructions only.
- Compartmentalized RAM is part of normal CR80 memory address space of A and B bus, and shared with the peripheral module internal controller processor.
- Interfaces to the disk adapter CR8084 for connection of 1 to 4 disk drives.
- Any combination of drives from CDC's SMD, MMD, and CMD families is possible.
- The dual channel feature of the disk drives is supported.
- Dual bus ports.
- Overlayed seeks supported.

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definition:

S1	6	5	4	3	2	1	Front Panel
	A19	A18	A17	A16	A15	A14	

S2	8	7	6	5	4	3	2	1	Front Panel
	P1	P0	MA5	MA4	MA3	MA2	MA1	MA0	

RAM Address Switch Setting:

The address switch S1 determine the bottom address of the address space occupied by the RAM.
16K RAM: all switches are valid
32K RAM: A14 is a 'don't care'.
64K RAM: A14 and A15 are 'don't cares'.
A19 is address bit 19 and MSB.

Module Address Switch Setting:

The switch S2 determine the CR80 module address and the interrupt priority.

P1-P0: interrupt prioprity P1: MSB.
MA5-MA0: module address MA5:MSB

0~switch ON, 1~switch OFF

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Main disk operations are:
  Seek, Read, Write, Seek and Read, Seek and Write, Format, Read address field.

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.

- Dimensions:

Width:	17.1 mm
Length:	326.0 mm
Height:	412.0 mm
Weight:	1.4 kg

- Power (including Disk CTRL Adapter) Consumption:

	16K RAM	32K RAM	64K RAM
+12V	100(700)mA	100(1100)mA	100mA
+5V	9.5A	9.5A	9.5A (10.6A)
-12V	210mA	210mA	200mA

  The figures in parentheses are max. current at max. speed operation in the RAM. The other figures are typical values with no RAM operation but refresh.

- Applications for CR8044M/01XAB/XX
  X = 0 16K RAM
  X = 1 32K RAM
  X = 2 64K RAM
  XX = 01 No built-in test. Chained buffers.
  XX = 02 Built-in test. Chained buffers or buffer table.

- RAM write response time T_{RW}: 350 ns $< T_{RWmin} <$ 470ns if RAM is not occupied by the controller, e.g. for refresh.
  Otherwise, $T_{RWmin} < T_{RWmax} < T_{RWmin}$ + 440 ns

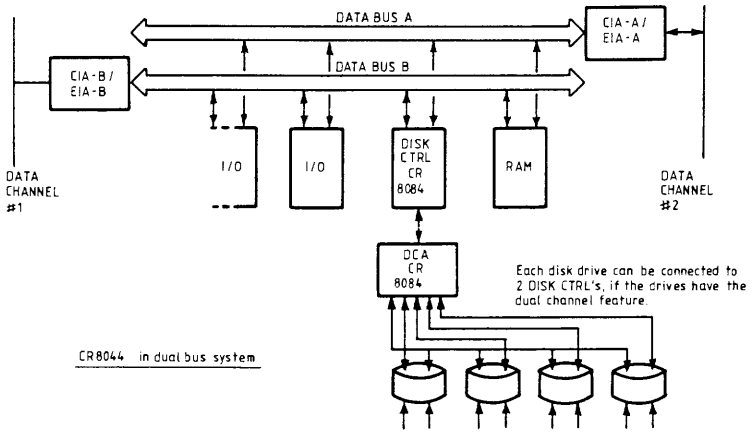
- RAM read response time $T_{RR} = T_{RW}$ + 50 ns.

- 5% occupation of RAM due to refresh.

- Controller response time T_{CR}: 290ns $< T_{CR} <$ 415 ns.

- MTBF: 30,200 h

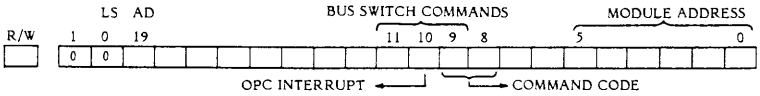
PROGRAMMERS REFERENCE: (CONTINUED)



CR8044 in dual bus system

COMMANDS

Commands:



BUS SWITCH COMMANDS

AD	11	10	9	
	1	0	0	DISABLE DISK CTRL, RESET
	1	0	1	ENABLE DISK CTRL, RESET

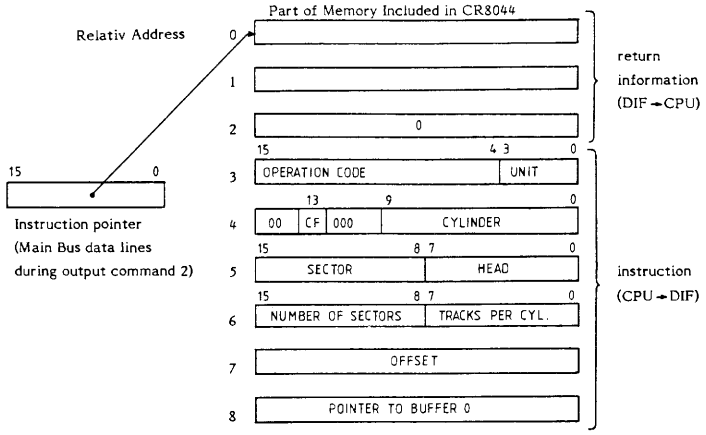
COMMAND CODE

R/W	Command Code	Command
0	0 - 3	read status word, clear interrupt
1	0	reset
1	1	load unit interrupt mask
1	2	load instruction pointer, initiate operation
1	3	terminate operation

(CONTINUED)

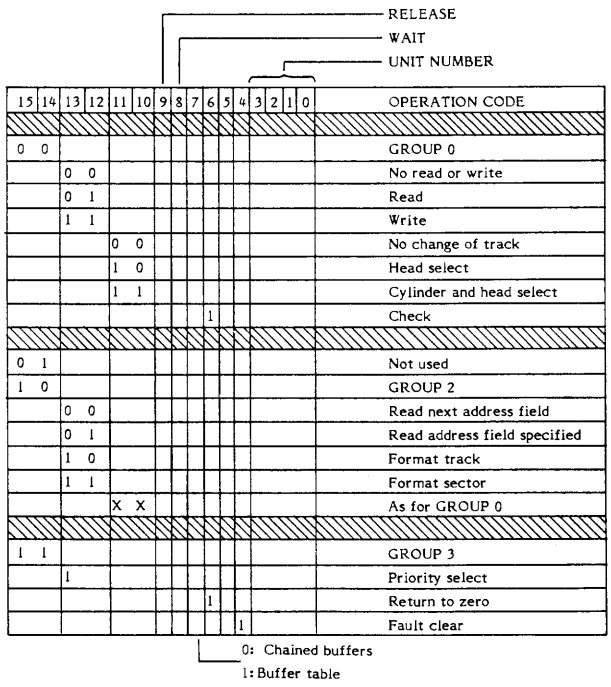
PROGRAMMERS REFERENCE: (CONTINUED)

INSTRUCTION FIELD



This does not apply to format.

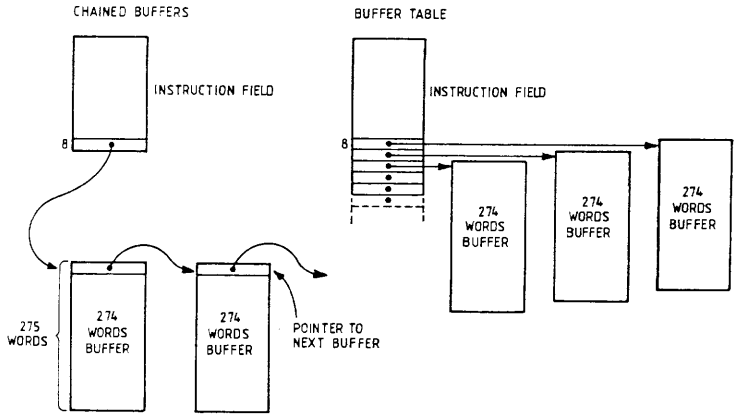
OPERATION CODE



(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

BUFFER POINTERS:

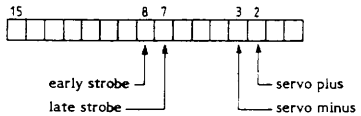


Only the last buffer pointer can be 0.

OBS: Firmware application XX = 01 will only handle chained buffers, while XX = 02 handles both buffer structures.

OFFSET:

Word 7 of instruction field



FORMAT AND READ ADDRESS FIELD:

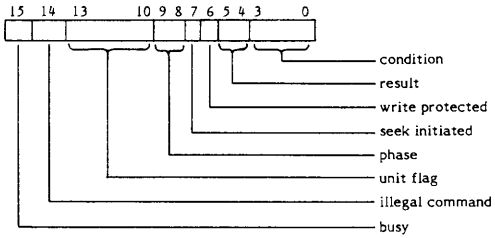
FORMAT		modifications of instruction field 15 14 13 9 8 7 0		read address field
relativ address	4	B W CF	cylinder 0	relativ address
	5	sector	head 1	
	6	auxiliary bytes	2	

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

STATUS

Status Word (Input Command 0, 1, 2 or 3)



Condition Codes

0. Busy
1. Was seeking
2.
3. Unexpected drive status
4. Check or Sync data field
5. Check or Sync address field
6. Address field
7. Buffer pointer
8. Illegal sector
9. Timing error
10. Memory overrun
11. Parity error
12. Selftest running
13. Reset. Selftest succeeded.
14. Selftest failed
15. Group 1 operation

Bit 9-8 phase description:

0 0	0	unit selection, status check, waiting for a busy drive to get free.
0 1	1	initial seek
1 1	2	searching for the first sector to be handled
1 0	3	read, write or format running

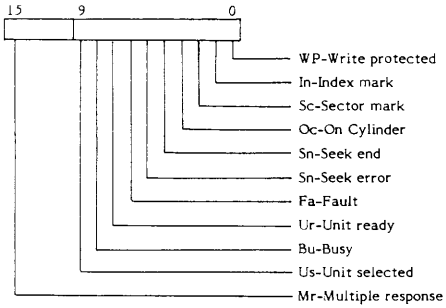
Result: Only valid when bit 15 is "0". Describes the latest operation:

1 0	Normal ending
0 1	Terminated by the command "Terminate operation" from CPU.
0 0	Terminated because of the condition indicated by bits 3 - 0.

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Re: 3. Unexpected drive status. Some of the 11 drive status lines had some contents not allowed at the time they were checked. The status signals are stored in word 0 of the instruction field:



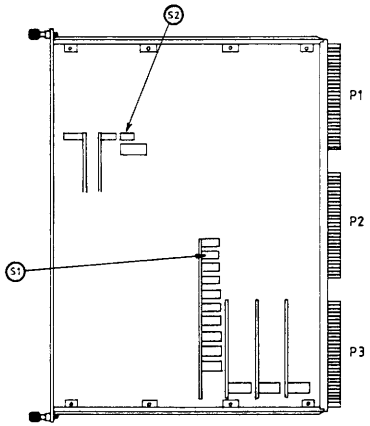
The signal Mr is generated by the Disk Adapter CR8084. A "1" indicates that wrong drive(s) responded to the unit selection. Word 1has a "1" for those bits in word 0 that are not as expected.

Volume Organization:

A disk volume is divided into a number of cylinders and tracks determined by head position and number. Each track is divided in 32 sectors, which contain an address field with full address of the sector and write protect, bad mark bits and 2 auxiliary bytes. A data field, which contains 548 bytes user data, is following the address field.

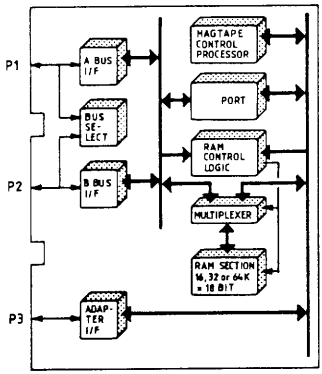
TAPE CTRL. CR 8045M/0XXAX/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Mag Tape Controller combined with 16K, 32K or 64K memory.
- Control/Status memory section accessible by privileged instructions only.
- Compartmentalized RAM is part of normal CR80 memory address space of A and B bus, and shared with the peripheral module internal controller processor.
- Connection of up to 8 daisy chained Tape Transports, selected from the PERTEC (F)T9000 series.
  Optionally the series (F)T6000, (F)T7000, (F)T8000 and (F)T1000 can be used.
- Main Mag Tape operations are:
  Rewind
  Skip forward, reverse
  Erase
  Write
  Edit
  Read forward, reverse.
- Memory Transfer rates:
  To/from Mag Tape:
  200K bytes/sec. (maximum)

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

S1

6	5	4	3	2	1
A5	A4	A3	A2	A1	A0

Front Panel ←

S2

8	7	6	5	4	3	2	1
P1	P0	M5	M4	M3	M2	M1	M0

Memory Address Space determined by:

16K version: A0-A5
32K version: A1-A5
64K version: A2-A5

Module Address and Priority determined by

M0-M5, P0-P1; M0 LSB, P1 MSB

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Versions of Tape Ctrl:
  CR8045M/010A-/00 16 K RAM, Single Bus
  CR8045M/011A-/00 32 K RAM, Single Bus
  CR8045D/012A-/00 64 K RAM, Single Bus
  CR8045M/040AB/00 16 K RAM, Dual Bus
  CR8045M/041AB/00 32 K RAM, Dual Bus
  CR8045M/042AB/00 64 K RAM, Dual Bus

- Power Consumption:

+5V:	6.5A
+12V:	0.3A
-12V:	0.1A

- Mechanical Dimensions:

Length:	305.0 mm
Heigth:	412.0 mm
Width:	17.1 mm (1 module)

- Weigth: 1200 g

- Mean Time between failure:
  $36 \cdot 10^3 h$

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.

PROGRAMMERS REFERENCE: (CONTINUED)

The module consists of a Mag Tape Controller part and a memory part. These are connected internally for data transfer between Mag Tape and memory concurrent with bus access to the memory.

The Mag Tape Controller receives commands and delivers status via its own bus control logic, whereas instruction and status concerning Mag Tape operations are fetched/stored in a command buffer consisting of 7 16 bits registers.

The Mag Tape Controller is built around a high speed programmable controller. The controller interprets instructions stored in the command buffer, controls and monitors the Mag Tape Transports and Formatter and performs the byte oriented data transfers.

The Tape Controller accepts the following commands:

R/W	AD11-6 Command Code	Command
0	000 $b_2b_1b_0$	Read Command Buffer - Location $b_2b_1b_0$
1	000 $b_2b_1b_0$	Write Command Buffer - Location $b_2b_1b_0$
-	100 - - -	Disable TAPE CTRL, RESET
-	101 - - -	Enable TAPE CTRL, RESET

Command Buffer Layout

FUNCTION	LOCATION (b2b1b0)	Read/Write	CPU 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Used by Controller	0																	
TRANSFER CONTROL	1	(R)/W	Not Used				BMO	BAD	OFS	DIN	OP 4-0							TN 2-0
TRANSFER COUNT	2	R/W	CT 15-0															
MEMORY ADDRESS	3	(R)/W	AD 15-0															
READ OVERFLOW/ READ OFFSET	4	R/W	OV15-0/OF 15-0															
READ OVERFLOW/ OFFSET, REWIND MASK	5	R/W	RM 7-0								OV23-16/ OF 23-16							
TRANSPORT STATUS	6	R/W	O	TCR		OO		ONL	RDY	RWD	WPT	BOT	EOT	NXZ	SRD	SPG	SDL	TN 2-0
TRANSFER STATUS	6	R/W	PFL	TCR	ERR	SC 3-0					THE	TCE	CPE	MEP	ETS	DTE		TN 2-0
REWIND, CONTROLLER STATUS	7	R	O	MER	ARE	CRE	ODP	HTS			OO	RW 7-0						

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

In order to initiate a new operation, the CPU loads TRANSFER CONTROL with the operation code and the transport number used to select between a maximum of 8 transports. If required, TRANSFER COUNT, MEMORY ADDRESS, READ OFFSETT or REWIND MASK are also loaded.

Finally, the TRANSFER STATUS is cleared. This makes TCR = "0". TCR = "0" Tape Controller Not Ready means that the Tape Controller is busy executing the last received operation. For Tape Controller Busy the CPU is not allowed to change the Command Buffer.

Having performed the last operation, the Tape Controller, if required, updates TRANSFER COUNT, READ OVERFLOW and loads the new status information into location 6. Finally, the status bit Tape Controller Ready is set, and an interrupt is generated. The transfer operations and the corresponding operation codes are:

OPERATION CODE (HEX)	OPERATION	CONTROL LOCATIONS USED
00	Start Rewind	
01	Start Rewind and Unload	
02	Read Transport Status	
03	Skip N Data-records Forward	CT
04	Skip N Data-records Reverse	CT
05	Skip N Filemarks Forward	CT
06	Skip N Filemarks Reverse	CT
07	Skip N Data-records or Filemarks Forward	CT
08	Skip N Data-records or Filemarks Reverse	CT
09	Skip Forward to Logical End of Tape	
0A	Write File Mark	
0B	Write Record	CT, MA
0C	Edit Record	CT, MA
0D	Erase Fixed Length	
0E	Erase Variable Length	CT
0F	Read Record Forward, Normal Treshold	CT, MA, (RO)
10	Read Record Forward, Low Treshold	CT, MA, (RO)
11	Read Record Forward, High Treshold	CT, MA, (RO)
12	Read Record Reverse	CT, MA
13	Read Record Reverse, Edit	CT, MA
14-1D	Not Used at Present (Diagnostic)	
1E	Diagnostic - Write I/F Reg., RAM	
1F	Diagnostic - Read I/F Reg., RAM	

The standard transport used (Pertec T9640-98-75) is a 9 track PE/NRZI read after write transport at a speed of 75 ips. It uses tape reels with a maximum diameter of 10.5 inches, containing 2400 feet of tape. The tape format - NRZI or PE - is selectable on the tape transport.

The rewind time is 115 secs.

The forward move time of a 2400 feet tape is about 385 secs.

Using PE recording the data density is 1600 bytes/inch. This gives an unformatted capacity of 46M bytes.

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

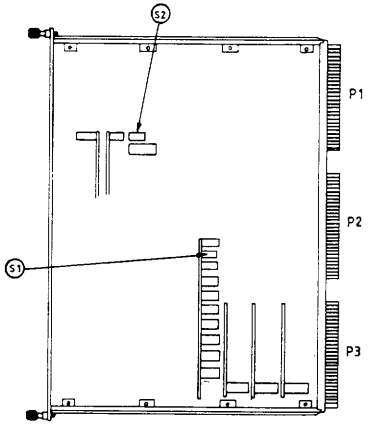
The formatted capacity is about 30M bytes with records of 2048 bytes. Error detection and signle error correction is provided for PE. Odd parity is used. The data transfer rate is 120K bytes/sec.

Using NRZI recording the data density is 800 bytes/inch. This gives an unformatted capacity of 23M bytes.

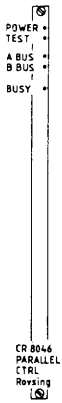
The formatted capacity is about 18M bytes with records of 2048 bytes. Error detection with odd parity is provided for NZRI. The data transfer rate is 60K bytes/sec.

PARALLEL CTRL., LINE PRINTER APPL. CR 8046M/0XXAX/00

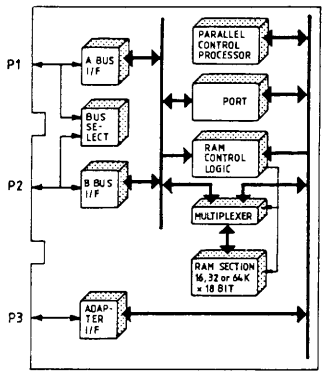
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Parallel Controller combined with 16K memory.
- Control/Status memory section accessible by privileged instructions only.
- Compartmentalized RAM is part of normal CR80 memory address space of A and B bus, and shared with the peripheral module internal controller processor.
- Connection of up to 4 line printers selected from the Data Products Corporation or Control Data Corporation series of line printers.
- Dual bus port.
- Memory Transfer Rates:

  To line printer:
  250K bytes/sec.

- Versions of Par. Ctrl., Line Printer Application:
  CR8046M/010A-/00 Single Main Bus
  CR8046M/040AB/00 Dual Main Bus

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

S1

6	5	4	3	2	1
A5	A4	A3	A2	A1	A0

Front Panel

S2

8	7	6	5	4	3	2	1
P1	P0	M5	M4	M3	M2	M1	M0

Memory Address Space determined by A0-A5

Module Address and Priority determined by M0-M5, P0-P1, M0 LSB, P1 MSB

The module consists of a Controller part and a memory part. These are connected internally for data transfer between line printer and memory concurrent with bus access to the memory.

The controller receives commands and delivers status via its own bus control logic, whereas instruction and status concerning line printer operations are fetched/stored in a command buffer consisting of 5 16 bits registers and a command area in the local memory.

The Parallel Controller is built around a high speed programable controller. The controller interprets instructions stored in the command buffer, controls and monitors the line printers, and performs the byte oriented data transfer.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Power Consumption:
  - +5V: 6.5A
  - +12V: 0.3A
  - -12V: 0.1A
- Mechanical Dimensions:
  - Length: 305.0 mm
  - Heigth: 412.0 mm
  - Width: 17.1 mm (1M)
- Weigth: 1200 g
- Mean Time Between Failure:
  $36 \cdot 10^3$ h
- Built-in test activated on power up.

PROGRAMMERS REFERENCE: (CONTINUED)

The Parallel Controller accepts the following commands:

R/W	AD 11-6 Command Code	Command
0	000 $b_2b_1b_0$	Read Command Buffer - Location $b_2b_1b_0$
1	000 $b_2b_1b_0$	Write Command Buffer - Location $b_2b_1b_0$
-	100 - - -	Disable PAR CTRL, RESET
-	101 - - -	Enable PAR CTRL, RESET

Command Buffer Layout

FUNCTION	LOCATION (b2b1b0)	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
	0	Not Used
COMMAND POINTER PRINTER #0	1	INT Ø EC Ø CMDP0 MSB LSB
COMMAND POINTER PRINTER #1	2	INT 1 EC 1 CMDP1 MSB LSB
COMMAND POINTER PRINTER #2	3	INT 2 EC 2 CMDP2 MSB LSB
COMMAND POINTER PRINTER #3	4	INT 3 EC 3 CMDP3 MSB LSB
SELF TEST STATUS	5	UNDEFINED ALWAYS ZERO STE

Associated with each line printer is a Command Area. The Command Area may be placed anywhere in the local 16K RAM. The Command Area is set up by the CPU and the access right to the area is transferred to the PAR CTRL by:

- placing the address of the first word in the Command Area in CMDP.
- clearing INT
- setting EC

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

The PAR CTRL uses the Command Area during execution of the command, updates a status word upon completion and transfers the access right to the area to the CPU by setting INT in the Command Buffer and generating an interrupt. EC is set if any of the bits in STATUS are set.

Command Area Layout

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATABUFFER ADDRESS	NOT USED 1)								DATABUFFER ADDRESS							
BYTECOUNT	BYTECOUNT															
STATUS	NOT USED 2)						PEDA	PECA	NOT USED 2)			OEC	OOP	ONL	READY	CONN
REMAINING BYTECOUNT	REMAINING BYTECOUNT															
PAR CTRL WORK STORAGE	PAR CTRL WORK STORAGE															

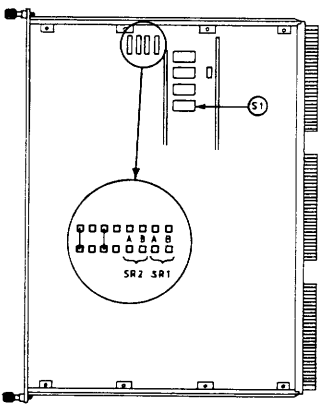
1): DON'T CARE, PAR CTRL addresses local 16K RAM only.

2): Always logical "0".

The CPU may investigate the status of the line printers by setting up a transfer of zero bytes, wait for an interrupt, and read the status word in the command area.

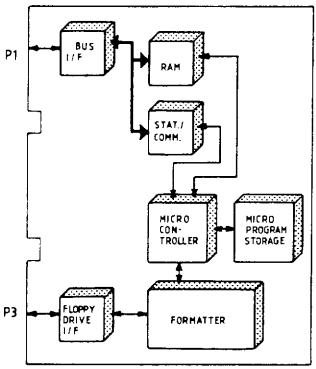
ST. FD CTRL. CR8047M/XXXA-/XX

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM

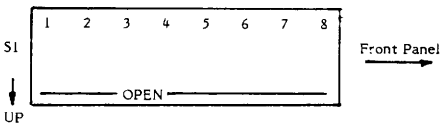


CHARACTERISTICS:

- Floppy disk controller for the CR80 unmapped system.
- Interfaces up to 4 Floppy disk drives
- Single and 2-sided diskettes
- IBM 3740 Data Format (single density)
- Interrupt after each operation
- Internal RAM contains 64 words (1 sector)
- Obtainable speed app. 1 word/2usec.
- Module device address and interrupt priority programable by S1.
- Power Consumption:
  +5V: 3.5A
  +12V: 25mA
- Size: 412.6 x 304 x 17
- Weight: 0.9 kg
- MTBF: 84700 hours

PROGRAMMERS REFERENCE:

Switch Orientation and Definition:



Switch Number 1 - 6 select module device address in binary code SW1 LSB, SW6 MSB

Switch Number 7 - 8 select interrupt priority in binary code SW7 LSB, SW8 MSB

Note: An open switch is equal to logical "1".

The straps SR1 and SR2 select one of two possible command formats:

SR1 selects: AD6 or AD15
SR2 selects: AD7 or AD17

The formats shown overleaf are valid with SR1 and SR2 in pos. B (AD6 and AD7 selected).

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

CPU INSTRUCTION

	AD15 AD14 AD13 AD12 AD11 AD10 AD9 AD8 AD7 AD6 AD5 AD4 AD3 AD2 AD1 AD0	
WIO	X X X X 1 X X X 0 1 A A A A A A	Reset FD CTRL,
WIO	X X X X 1 X X X 0 1 A A A A A A	Reset FD CTRL,
WIO	X X X X X X X X 0 1 A A A A A A	Reset Pointer
WIO	X X X X 0 X X X 1 0 A A A A A A	Write date to buffer, the buffer pointer is increment 1 address
WIO	X X X X 0 X X X 0 0 A A A A A A	Normal system command, command type is specified in data word

Data Word for Normal System Command

D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0	
0 0 0 0 0 0 0 D D X X X X X X X X	Select drive
0 0 1 0 0 0 0 D D T T T T T T T T	Seek track
0 1 0 0 0 0 0 D D X X X X X X X X	Restore head
0 0 0 1 0 S D D R R R R R R R R	Read initial record No.
0 0 0 0 1 S D D R R R R R R R R	Write initial record No.
0 0 1 0 1 S D D R R R R R R R R	Write deleted record

TT Track Number (0:77)
RR Record (sector) Number (1-26)
S Diskette side (1:2)
DD Drive Number (0:3)
XX Don't care
XXXXXX Don't care
AAAAAA Module device address

AD15 AD14 AD13 AD12 AD11 AD10 AD9 AD8 AD7 AD6 AD5 AD4 AD3 AD2 AD1 AD0	
RIO	X X X X 0 X X X 1 0 A A A A A A	Read data from buffer, Buffer pointer is increment 1 address
RIO	X X X X 0 X X X 0 X A A A A A A	Read Auto Status
RIO	X X X X 1 X X X 0 X A A A A A A	Read additional status

ADDITIONAL STATUS CODE

Bit Position	Description	Definition
0	DISK CH 0	Disk Change, Drive 0
1	DISK CH 1	Disk Change, Drive 1
2	DISK CH 2	Disk Change, Drive 2
3	DISH CH 3	Disk Change, Drive 3
4	DEL REC	DELETED RECORD, i.e. an F8 AM found during a read operation
5	OVERRUN	HANDSHAKE ERROR, transfer between floppy disk drive and sector buffer failed.
6	BUSY	The formatter is executing a command
7	TWO SIDED	The selected drive contains a two sided diskette
8	UNDEFINED	

(CONTINUED)

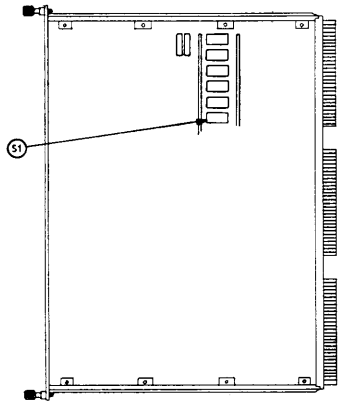
PROGRAMMERS REFERENCE: (CONTINUED)

AUTOMATIC STATUS ERROR CODE

Bit Position	Description	Definition
0	Seek Complete	Seek command was executed successfully
1	ID DATA CHECK	CRC failed to compare in the ID field
2	PROGRAM ERROR	Invalid command
4	DATA CHECK	CRC check failed on the data field
5	ID NOT FOUND	Search of the ID field was not successful after revolutions
6	EQUIPMENT CHECK	Resident micro diagnostics failed
7	ADD.STATUS NOT ZERO	One or more bits in the additional status (see later) are set
8	WRITE PROTECT	Selected drive is write protected
11	DRIVE NOT READY	Selected drive is not ready
12	SEEK ERROR	Upon completion of the SEEK or RESTORE command, the track address does not compare
13	RECALIBRATE ERROR	Track 00 was not detected in response to a RESTORE command

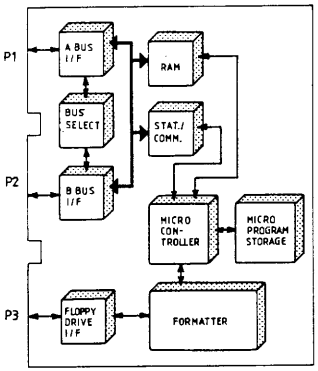
ST. FD CTRL. CR8047M/XXXAB/XX

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM

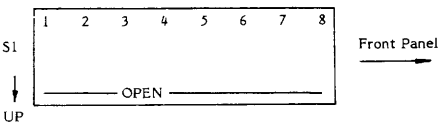


CHARACTERISTICS:

- Floppy disk controller for the CR80 system
- Interfaces up to 4 Floppy disk drives
- Single and 2-sided diskettes.
- IBM 3740 Data Format (single density)
- Compartmentalized RAM is part of normal CR80 memory address space of A and B bus, and shared with the peripheral module internal controller processor.
- Parity check/generation, at each module access
- Interrupt after each operation
- Internal RAM contains 64 words (1 sector)
- Obtainable speed app. 1 word/2usec.
- Module device address and interrupt priority programable by S1

PROGRAMMERS REFERENCE:

Switch Orientation and Definition:



Switch Number 1 - 6 select module device address in binary code SW1 LSB, SW6 MSB

Switch Number 7 - 8 select interrupt priority in binary code SW7 LSB, SW8 MSB

Note: An open switch is equal to logical "1". (CONTINUED)

- MTBF: 59500 hours
- Size: 412.6 x 304 x 17
- Weight: 1.1 kg
- Power Consumption:
  +5V: 3.5A
  +12V: 25mA
- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.

PROGRAMMERS REFERENCE: (CONTINUED)

CPU INSTRUCTION

	AD15 AD14 AD13 AD12 AD11 AD10 AD9 AD8 AD7 AD6 AD5 AD4 AD3 AD2 AD1 AD0	
WIO	X X X X I 0 0 X 0 1 A A A A A A	Reset FD CTRL, disable Bus A and B
WIO	X X X X 1 0 1 X 0 1 A A A A A A	Reset FD CTRL, enable switch
WIO	X X X X X X X X 0 1 A A A A A A	Reset Pointer
WIO	X X X X 0 X X X 1 0 A A A A A A	Write date to buffer, the buffer pointer is increment 1 address
WIO	X X X X 0 X X X 0 0 A A A A A A	Normal system command, command type is specified in data word

Data Word for Normal System Command

D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0	
0 0 0 0 0 0 D D X X X X X X X X	Select drive
0 0 1 0 0 0 D D T T T T T T T T	Seek track
0 1 0 0 0 0 D D X X X X X X X X	Restore head
0 0 0 1 0 S D D R R R R R R R R	Read initial record No.
0 0 0 0 1 S D D R R R R R R R R	Write initial record No.
0 0 1 0 1 S D D R R R R R R R R	Write deleted record

TT Track Number (0:77)
RR Record (sector) Number (1-26)
S Diskette side (1:2)
DD Drive Number (0:3)
XX Don't care
XXXXXX Don't care
AAAAAA Module device address

AD15 AD14 AD13 AD12 AD11 AD10 AD9 AD8 AD7 AD6 AD5 AD4 AD3 AD2 AD1 AD0	
RIO X X X X 0 X X X 1 0 A A A A A A	Read data from buffer, Buffer pointer is increment 1 address
RIO X X X X 0 X X X 0 0 A A A A A A	Read Auto Status
RIO X X X X 1 X X X 0 1 A A A A A A	Read additional status

ADDITIONAL STATUS CODE

Bit Position	Description	Definition
0	DISK CH 0	Disk Change, Drive 0
1	DISK CH 1	Disk Change, Drive 1
2	DISK CH 2	Disk Change, Drive 2
3	DISH CH 3	Disk Change, Drive 3
4	DEL REC	DELETED RECORD, i.e. an F8 AM found during a read operation
5	OVERRUN	HANDSHAKE ERROR, transfer between floppy disk drive and sector buffer failed.
6	BUSY	The formatter is executing a command
7	TWO SIDED	The selected drive contains a two sided diskette
8 - 15	UNDEFINED	

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

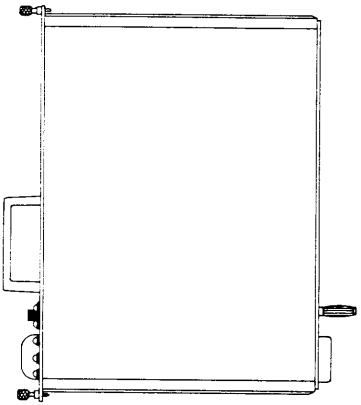
AUTOMATIC STATUS ERROR CODE

Bit Position	Description	Definition
0	Seek Complete	Seek command was executed successfully
1	ID DATA CHECK	CRC failed to compare in the ID field
2	PROGRAM ERROR	Invalid command
4	DATA CHECK	CRC check failed on the data field
5	ID NOT FOUND	Search of the ID field was not successful after revolutions
6	EQUIPMENT CHECK	Resident micro diagnostics failed
7	ADD.STATUS NOT ZERO	One or more bits in the additional status (see later) are set
8	WRITE PROTECT	Selected drive is write protected
11	DRIVE NOT READY	Selected drive is not ready
12	SEEK ERROR	Upon completion of the SEEK or RESTORE command, the track address does not compare
13	RECALIBRATE ERROR	Track 00 was not detected in response to a RESTORE command

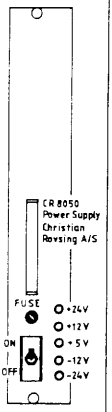
DATASHEET FOR:

POWER SUPPLY CR 8050M/010 - -/00

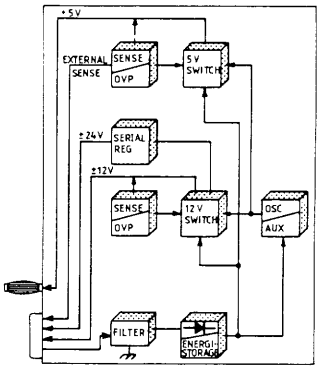
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Standard CR80 plug in (4M).
- Input:
  110V/220V -10 +20%, 50 - 440 Hz
- 28 msec energy storage at max. output and min. input.
- Built-in mains filter.
- Output:
  5V or 5.5V/60A
  +/-12V or +/-12.5V/4A
  +/-24V/150mA

  OVP on the 5V and the +/-12V
- Current limitting on all outputs.
- 5V external sense.
- 40KHz switching.
- LED indication on all outputs.
- MTBF 26800 H/F

PROGRAMMERS REFERENCE:

Main Fuse:

110V/10A - 220V/5A

Strapping on rear plug for selecting output voltage.

Output voltage for power sharing with another module.

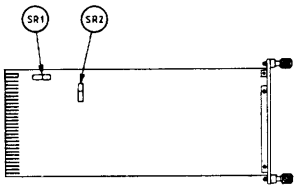
The +/-12V group can be loaded as a power generator (+7A/-1A).

If the +/-12V group is shortened, the +5V is turned off to protect load circuits.

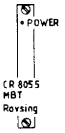
- Dimensions:

Height:	412.0 mm
Width:	69.0 mm
Depth:	305.0 mm
Weight:	6.5 kg

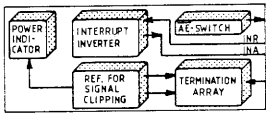
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- The module does perform proper termination and voltage limitation of the different types of main bus signals of CR80.

- The signal lines:

  DA (15:0), UP, LP
  AD (19:0)
  LS0, LS1
  R/W

  are pulled up to a logical "1" level when the lines are tri-stated.

- The "0" levels will never be less than -0.1V.
  The "1" levels will never be greater than +4.1V.

- The signal lines, BD, MC, TRQ, RS, INA, INR, AE, are terminated to the ground and supply level. The quicent voltage of the signal lines is app. 2.5V.

- The clock signals, Ø1, Ø2, are terminated to ground level only.

- MTBF: 286,000 hours

PROGRAMMERS REFERENCE:

There is no possibility of controlling the module in software aspects.

The only programing is hardware and two straps (SR1 and SR2) must be mounted in correct position before a "new" computer system is started up for the first time. In position A strap SR1 will shortcircuit the AE-signal to ground, forcing the modules in the system to work without MAP module.

System without MAP-module, SR1 in position A.
System equipped with MAP-module, SR1 in position B.

The strap SR2 is used to introduce or remove a static voltage drop in supply voltage to the module. This strapping is necessary due to increased supply voltage in channel crates.

MBT used in channel crate, SR2 in position A.
MBT used in Processing Unit, SR2 in position B.

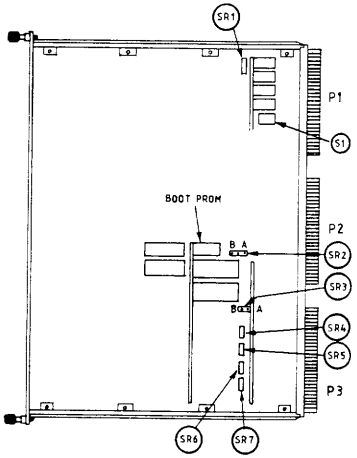
- Furthermore, the module contains the inverter which is a vital function in the interrupt system.

- Power Consumption:
  5V 0.4A

- Weight: 0.3 kg

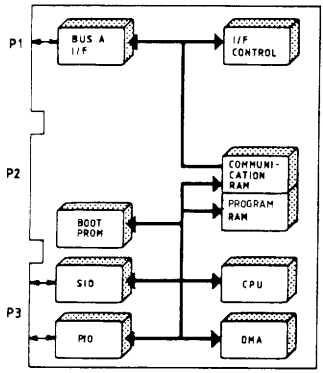
- Size: 131 x 300 x 17

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Standard CR80 Communications I/F, protocol (Async., sync. and bit synchroneous) and comm. line characteristics software defined by CR80 downloaded program.
- Single bus interface
- Module address is set by switch 1
- Compartmentalized communication RAM is part of normal CR80 memory address space of A-Bus and shared with LTU internal controller processor. Program RAM only accessible by LTU internal controller processor, program (communication protocol dependent) is down loaded by CR80 into communication RAM and moved by LTU internal controller processor into program RAM, ensuring integrity against accidental modification of program.
- Bootloader Prom 2K or 4K selected by strap 2.
- 4 x V24/V28 communication lines
- 9.6K baud full duplex each channel or 48K baud full duplex on 2 channel with DMA utilization
- Interfaces to a Line Interface Adapter (LIA)
- MBTF: 45000 H/F

PROGRAMMERS REFERENCE:

Switch Orientation and Definitions

S1

8	7	6	5	4	3	2	1
P1	P0	A5	A4	A3	A2	A1	A0

Front Panel

Modul Address Setting:

Address determined by A5-A0, A5 MSB.

Interrupt Priority Setting:

Priority determined by P1-P0. P1 MSB.

(CONTINUED)

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.
- Dimensions:

Height:	412.6 mm
Width:	17.1 mm (1M)
Length:	305.0 mm
Weight:	1.5 kg

- Power Consumption:

+5V:	3.0A
+12V:	0.3A
-12V:	0.15A

PROGRAMMERS REFERENCE: (CONTINUED)

Jumper and Strap Setting:

SR1: if mounted: masterclear is active on LTU
SR2: position A 2K EPROM is selected
position B 4K EPROM is selected
SR3: position A GNS
position B LIA sense signal connected on LTU
SR4: transmitter clock channel A to internal clock
SR5: transmitter clock channel B to internal clock
SR6: transmitter clock channel C to internal clock
SR7: transmitter clock channel D to internal clock

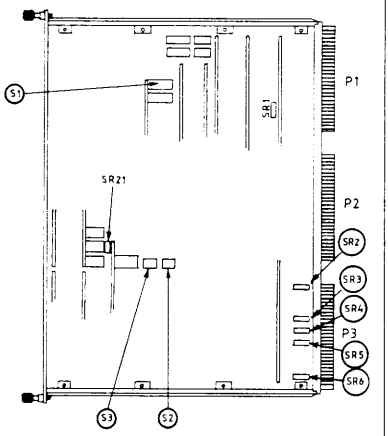
I/O Commands:

	ADDR 11-6						
R/W	11	10	9	8	7	6	Command
0	1	0	0	0	0	0	Programmed Clear
0	1	1	0	0	0	0	Set LTU in Bootload Mode
0	0	1	0	0	0	0	Interrupt request to microprocessor
1	0	0	0	0	1	1	Load Address Counter
0	0	0	0	0	0	0	Read Status Word B(0), B(1), P -> 2
0	0	0	0	1	0	0	Read word B(P), B(P+1) Fetch word B(P+1), B(P+2)
0	0	0	0	1	0	1	Read Lower Byte B(P) P -> P+1
1	0	0	1	0	0	0	Write word B(P), B(P+1) P -> P+2
1	0	0	0	0	0	1	Write lower byte B(P), P -> P+1
1	0	0	0	0	1	0	Write upper byte B(P), P -> P+1
0	0	0	1	0	1	1	Read Parity Status

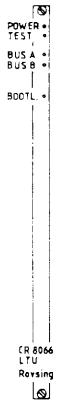
- Two basic versions available:
  CR8066M/020A-/00: Synchronous communication
  CR8066M/030A-/00: Asynchronous communication

LTU CR8066M/XXXAB/XX

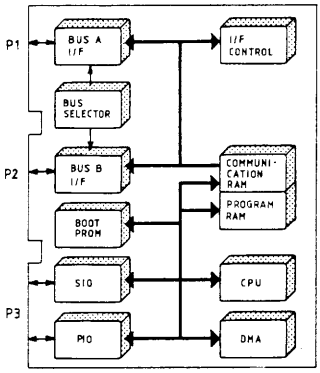
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Standard CR80 Communications I/F, protocol (Async., sync. and bit synchroneous) and comm. line characteristics software defined by CR80 downloaded program.
- Compartmentalized communication RAM is part of normal CR80 memory address space of A and B-Bus and shared with LTU internal controller processor. Program RAM only accessible by LTU internal controller processor, program (communication protocol dependent) is down loaded by CR80 into communication RAM and moved by LTU internal controller processor into program RAM, ensuring integrity against accidental modification of program.
- Dual bus interface
- Module address is set by switch S1
- RAM location is set by switch 2
- Bootloader Prom 2K or 4K selected by strap.
- 4 V24/V28 communication lines.
- Interface to a Line Interface Adapter (LIA)
- Dimensions:

Height:	412.6 mm
Width:	17.1 mm (1M)
Length:	305.0 mm

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definitions

S1

8	7	6	5	4	3	2	1
P1	P0	A5	A4	A3	A2	A1	A0

Front Panel

S2

4	3	2	1
M3	M2	M1	M0

S3

10	9	8	7	6	5
M9	M8	M7	M6	M5	M4

Module Address Setting:
Address determined by A5-A0, A5 MSB

Interrupt Priority Setting:
Priority determined by P1-P0, P1 MSB

RAM Start Address Setting:
M0 - M7 address 10 - 17
M8 - M9 Bank select BS0 - BS1

Jumpers and Straps:
SR1: if mounted: masterclear is active on LTU
SR21: position A 2K EPROM is selected
position B 4K EPROM is selected

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

SR4: position A GNS
position B LIA sense signal connected on LTU
SR2: transmitter clock channel A to internal clock
SR3: transmitter clock channel B to internal clock
SR5: transmitter clock channel C to internal clock
SR6: transmitter clock channel D to internal clock

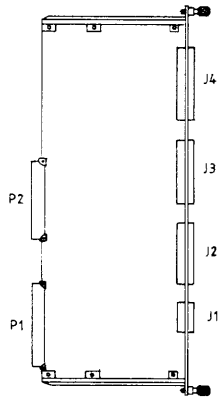
I/O Command:

	ADDR 11-6						
R/W	11	10	9	8	7	6	Command
0	1	0	1	0	0	0	Enable LTU, Programmed Clear
0	1	0	0	0	0	0	Disable LTU, Programmed Clear
0	1	1	1	0	0	0	Enable LTU, Set LTU in Boot-load Mode
0	0	1	0	0	0	0	Interrupt request to micro-processor
1	0	0	0	0	1	1	Load Address Counter
0	0	0	0	0	0	0	Read Status Word B(0), B(1), P -> 2
0	0	0	0	1	0	0	Read word B(P), B(P+1) Fetch word B(P+1), B(P+2)
0	0	0	0	1	0	1	Read Lower Byte B(P) P -> P+1
1	0	0	1	0	0	0	Write word B(P), B(P+1) P -> P+2
1	0	0	0	0	0	1	Write lower byte B(P), P -> P+1
1	0	0	0	0	1	0	Write upper byte B(P), P -> P+1
0	0	0	1	0	1	1	Read Parity Status

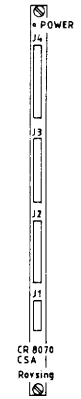
CHARACTERISTICS: (CONTINUED)

- Weight: 1.5 kg
- Power Consumption:
  +5V. 4.0A
  +12V: 0.3A
  -12V: 0.15A
- MTBF: 27,600 H/F
- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.
- Async. LTU:
  2K Firmware RAM
  2K Communication RAM
  9.6K Baud full duplex each channel.
  (CR8066M/030AB/XX)
- Sync. LTU:
  16K/32K firmware RAM
  16K/32K Communication RAM
  9.6K Baud full duplex each channel or
  48K Baud full duplex on two channels
  with DMA Utilization
  (CR8066M/010AB/XX)

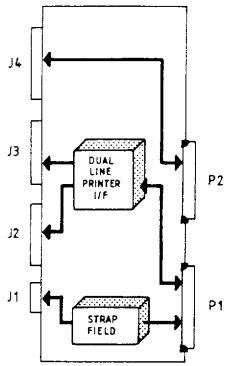
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- CPU-SCM (CR8002M) Adapter.
- Two line printer interfaces (CR8331 or CR8332).
- One terminal interface (V.24 I/F).
- Plugs for connection of Floppy Disk Controller (CR8047M) to Floppy Station (CR8308).
- Dimensions:
  Height: 412 mm
  Width: 17.1 mm
  Depth: 163 mm
  Weight: 0.5 kg
- Power consumption:
  (from CPU-SCM)
  5V: 440 mA
  +12V: 0
  -12V: 0
- MTBF: 90,000 hours

PROGRAMMERS REFERENCE:

Connectors:

J1:	V.24 I/F
J2:	Line Printer I/F No. 1
J3:	Line Printer I/F No. 2
J4:	Floppy Disk Station

Strap Survey:

STRAP		
SR1	P1, pin C9	AUX A
SR2	P1, pin C18	AUX B
SR3	J3, pin 8	(GND/5V)
SR4	J2, pin 8	(GND/5V)
SR5A	J3, pin 6	PRINTER: ON LINE
SR5B	J3, pin 4	PRINTER: I/F VERIFY SENSE
SR6A	J2, pin 6	PRINTER: ON LINE
SR6B	J2, pin 4	PRINTER: I/F VERIFY SENSE
SR7A	J1, pin 13	
SR7B	J1, pin 25	
SR7C	J1, pin 12	
SR7D	J1, pin 24	
SR7E	J1, pin 11	
SR7F	J1, pin 23	
SR7G	J1, pin 10	
SR7H	J1, pin 22	
SR7J	J1, pin 9	
SR7K	J1, pin 21	
SR7L	J1, pin 8	
SR7M	J1, pin 20	V.24: Data Terminal Ready

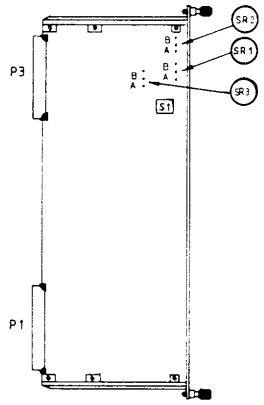
(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

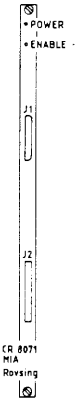
SR7N	J1, pin 7	V.24: Signal Ground
SR7O	J1, pin 19	
SR7P	J1, pin 6	V.24: Data Set Ready
SR7Q	J1, pin 18	
SR7R	J1, pin 5	V.24: Clear to Send
SR7S	J1, pin 17	V.24: Transmit S.E.T.
SR7T	J1, pin 4	V.24: Ready to Send
SR7U	J1, pin 16	
SR7V	J1, pin 3	V.24: Received Data
SR7X	J1, pin 15	V.24: Receive S.E.T.
SR7Y	J1, pin 2	V.24: Transmitted Data
SR7Z	J1, pin 14	
SR7Æ	J1, pin 1	

MIA CR8071M/010--/00

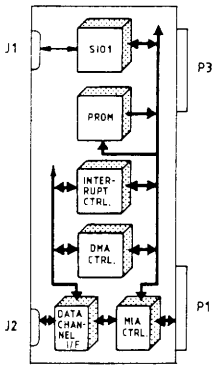
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



USED IN MAPPED SYSTEMS

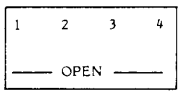
CHARACTERISTICS:

- Interface between MAP and DATA CHANNEL
- Performs the master control of the DATA CHANNEL
- Serial asynchroneous communication (CCITT recommendation V24)
- 4K words system PROM
- Polling of interrupts from 15 Channel Units.
- DC insulation from the DATA CHANNEL to avoid ground current problems.
- Transfer Cycle (word):
  Single trf: 1375 ns (min.)
  Reduced trf: 1000 ns (min.)
- Power Consumption:
  5V: 4A
  +12V: 25mA
  -12V: 15mA
- Mechanical Dimensions:
  Height: 412.6 mm
  Width: 17.1 mm
  Depth: 163.5 mm
- Weight: 800 g
- MTBF: 85500 h

PROGRAMMERS REFERENCE:

V24 Communication

1 out of 16 different baud rates may be selected by the switch S1.



1 2 3 4	Baud Rate	1 2 3 4	Baud Rate
0 0 0 0	50	0 0 0 1	1800
1 0 0 0	75	1 0 0 1	2000
1 1 0 0	134.5	0 0 1 1	3600
0 0 1 0	150	1 0 1 1	4800
1 0 1 0	300	1 0 1 1	7200
0 1 1 0	600	0 1 1 1	9600
1 1 1 0	1200	1 1 1 1	19200

1=open

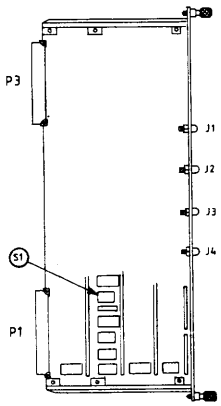
Strap SR2A: Connects protective ground to J1 pin 1
Strap SR2B: Disconnetcs protective ground from J1 pin 1
Strap SR3A: Connects +12V to J1 pin 9
Strap SR3B: Disconnect +12V from J1 pin 9

Adapter Disabling

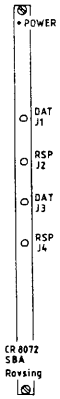
It is possible to insulate the PU from the connected busses by disabling the PU-adapters. This function is activated either

a) via the front panel connector J1 (strap SR1A)
or
b) via the connector P3 (strap SR1B) on the MIA.

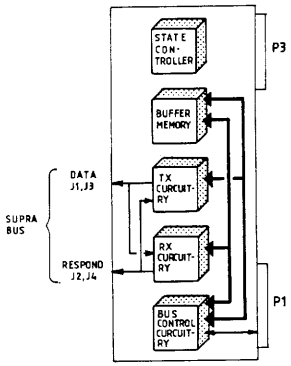
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM

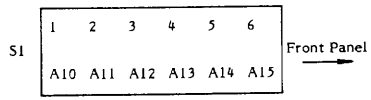


CHARACTERISTICS:

- STI adapter to the SUPRA BUS
- SUPRA BUS arbiter circuitry.
- 1.6M byte/sec. data transfer rate.
- Max. distance between any of two SBAs: 50m.
- Transmission error detection by CRC controller.
- 640 bytes buffer memory.
- Module Self Test facility.
- Transformer isolated SUPRA BUS.
- Variable transmission frame length.
- Transmission time for max. frame (256 bytes).
  t_{TX} = 150 usecs + t_{arbit}
  (t_{arbit} = Arbitration time - depends on actual SUPRA BUS load).
- Power Consumption:
  +5V: 3.3A (typ)
  -12V: 50mA
- MBTF: 90,100 hours/failure
- Size: 412 x 164 x 17.1
- Weight: 650 g

PROGRAMMERS REFERENCE:

Switch Orientation and Definition:



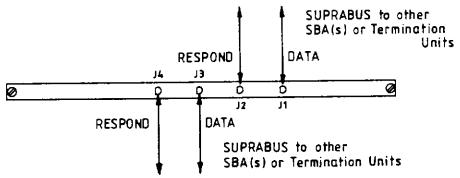
Module Address Setting:

Address determined by A15-A10; A15: MSB.
0=switch ON, 1= switch OFF

SUPRA BUS Connector J1-J4:

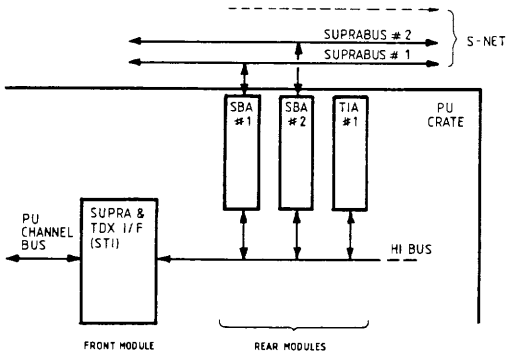
J1-J3 and J2-J4 are internally connected.

Typical SUPRA BUS ADAPTER Installation:



(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)



SBA in a Processor Unit

Module Initialization

RESTART activates module clearing sequence (internally generated at power up)

Module Disabling

PUAEN (low) disables SUPRA BUS drivers (SUPRA BUS not affected).

Module Address Map

- 1K byte address space divided in 16 address segments
- AB5-AB0 only used when addressing buffer memories
- Status register accessable in two address segments

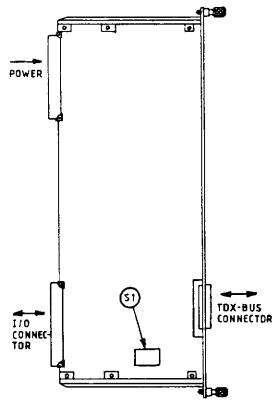
AB9	AB8	AB7	AB6		
0	0	Ø	Ø	RX RAM	(received frame, read only)
0	1	0	0		
0	1	0	1	STATUS REGISTER	(read only)
0	1	1	0	RX INITIALIZE	commands
0	1	1	1	CHANGE MODE	
1	0	Ø	Ø	TX RAM	(frame for transmission, write only
1	1	0	0		
1	1	0	1	STATUS REGISTER	(read only)
1	1	1	0	TX START	commands
1	1	1	1	TX STOP	

STATUS REGISTER

Bit No.	7	6	5	4	3	2	1	0 (LSB)
	TAEN	RAEN	TX	RX	MODE	RX.INTRPT	NO/ILL RSP	CRC ERROR

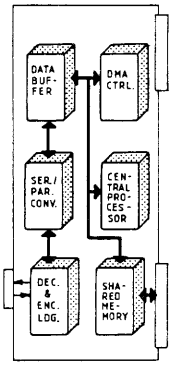
TAEN: TX RAM Access Enable
RAEN: RX RAM Access Enable
TX: Transmitting
RX: Receiving
MODE: SUPRA BUS Handling/Self Test Mode
NO/ILL RSP: No or Illegal Responce Received
RX INTRPT: Receiving Interrupted (2us time out)
CRC ERROR: Error detected in Received Frame

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- TDX Interface Adapter (TIA)
- The TIA is an intelligent slave module which interfaces one or two TDX-Busses to the STI.
- High bandwidth device - up to half the bandwidth of the TDX-Bus-400K Baud full duplex.
- Is able to interface all devices on the TDX-Bus.
- Reliability: MTBF = 117600 H/F
- Power Consumption:

  Driving two TDX-Outlets

  +5V: 4A +/- 0.5A

  -12V: 0.1A + 0.05A
- Physical Dimension:

  (412.6 x 152.5 x 17.2)mm.
- Weight: 0.6 kg

PROGRAMMERS REFERENCE:

Switch Orientation and Definition:

S1: LSB MSB NC 0=SWITCH ON

PIN1 . . . 1=SWITCH OFF

. . .

. . .

. . .

. . .

A10 A11 A12

SET BASE ADDRESS BIT A10-A12 IN SHARED RAM ON TIA SEEN FROM STI:

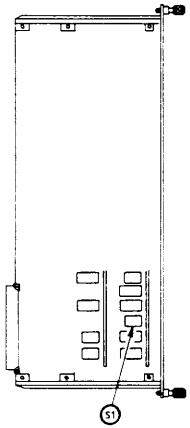
BIT (A15, A14, A13) = (0,1,1)

BIT A0 -> A9 = don't care.

TDX-HOST NO.: Is set-up on STI and read by TIA via the I/O connector.

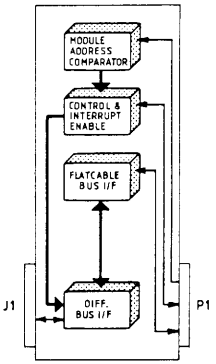
All communication between TIA and STI takes place in the 1K BYTE shared RAM on TIA. The RAM has a structure which divide it into a ringbuffer for outgoing data, a ringbuffer for ingoing data and a area for interchanging status and control informations.

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Active Bus Extension for the CR80 system together with EPM and EIA modules in unmapped systems.
- Interface between the Flatcable Bus of an EPM module & the Differential Flatcable Bus in a Processor Crate.
- One EIA-A&B in a CU-Crate can be connected to each EPA module at the Differential Flatcable Bus.
- An address switch S1 on the module assigns every EPA module a different address (up to eight EPAs are possible).
- A select LED on the front panel indicates when a data transfer is in progress.
- Mechanical Dimensions:

Height:	412.6 mm
Width:	17.1 mm (1M)
Length:	155.0 mm
Weight:	500 g

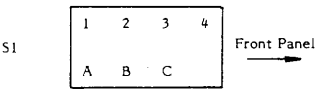
- Power Consumption:

+5V:	0.9A typ.

- MTBF: 255000 (H/F)

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

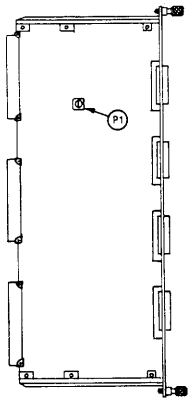


Interrupt Address Switch

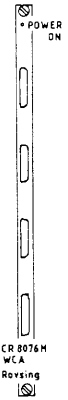
Address determined by A, B, C; A: LSB

(0~switch ON; 1~switch OFF).

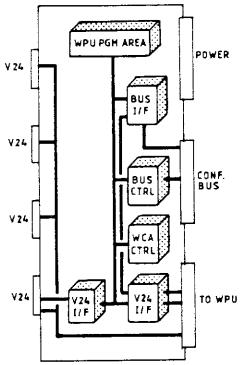
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

Watchdog CPU Adapter.

- The WCA connects to the WPU (LTU) via the standard LTU 4 ch. V24 interface, but use the non essential control lines (12 input and 12 output lines) for information interchange.
- Placed on the WCA is the serial interface between the WPU and the Configuration Bus - UART, Bus Driver/Receiver and failsafe circuit. Also on the WCA is found 4xV24 ports of which 1 has capability of running on Communication line.
- WPU software is placed in EPROM on the WCA board, and it is transferred to the program memory of the WPU in "bootload"-mode.
- Mechanical Dimensions:

Height:	375mm
Width:	17.2mm
Length:	152mm
Weight:	0.8 kg

- MTBF: 47,800 hours

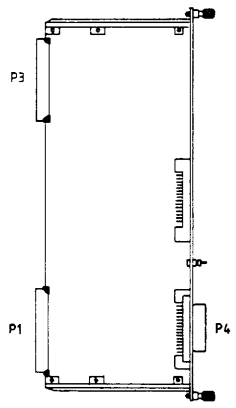
PROGRAMMERS REFERENCE:

P.1 is used to adjust the "power on" - reset voltage. Ajustment is done by measuring Vcc and "power on" - reset signal.
"Power on" - reset must go inactive (i.e. from low to high) when Vcc reaches 4.85V.

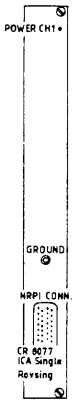
- Power Consumption:

+5V	1.5A
+12V	0.3A
-12V	150mA

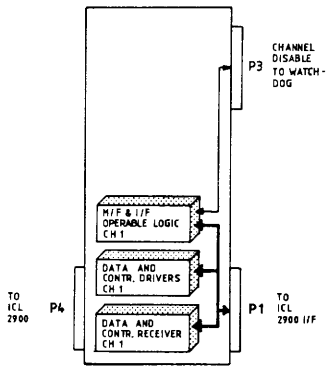
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Designed to convert the signals to and from the ICL 2900 I/F CR8038 to the correct levels, and to terminate the lines from the ICL 2900 NRP1.
- The adapter will support one ICL 2900 I/F.
- Mechanical dimensions:

Height:	412 mm
Width:	51.6 mm (3M)
Length:	155 mm
Weight:	0.8 kg

- Power Consumption:

+5V	1mA
-12V	400mA

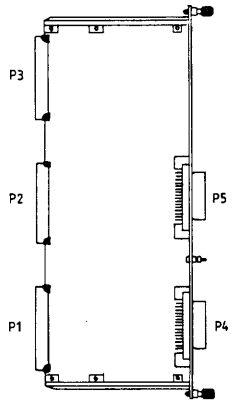
(Supplied via the ICL 2900 I/F)

- MTBF: 91 000 hours

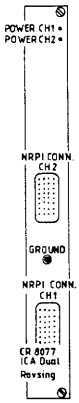
PROGRAMMERS REFERENCE:

(not applicable)

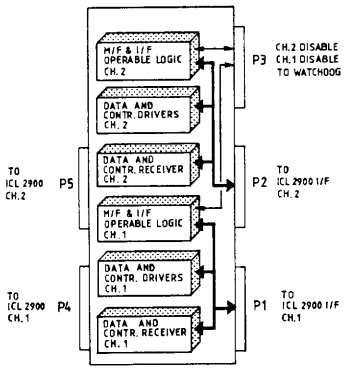
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Designed to convert the signals to and from the ICL 2900 I/F CR8038 to the correct levels, and to terminate the lines from the ICL 2900 NRP1.
- The adapter will support two ICL 2900 I/F.
- Mechanical dimensions:

Height:	412 mm
Width:	51.6 mm (3M)
Length:	155 mm
Weight:	0.8 kg

- Power Consumption:

+5V	500mA
-12V	200mA

(Supplied via the ICL 2900 I/F)

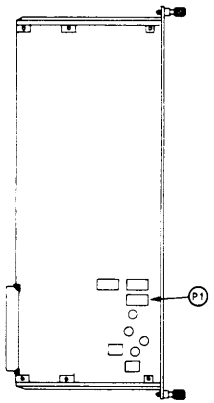
- MTBF: 45,500 hours

PROGRAMMERS REFERENCE:

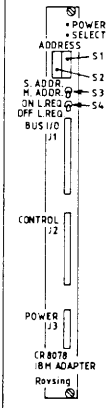
(not applicable)

IBM ADAPTER CR8078M/010--/00

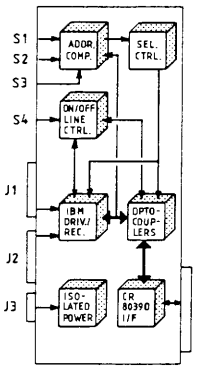
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Provides interface between an IBM 360/370 Block Multiplexer Channel and the IBM CH. I/F CR8039.
- Drivers/Receivers, which fulfils the IBM requirements.
- Electrical insulation (optocouplers) between the IBM Channel and CR8039.
- IBM device addresses switch selectable in blocks of 32, 61 or as a single address.
- Selectable priority towards the IBM Channel (high or low).
- Handles power up/down sequences (IBM specifications).
- Size:

Length:	164.0 mm
Height:	412.0 mm
Width:	34.2 mm (2 M)

- Weight: 600 g
- MTBF: 21600 (H/F)

PROGRAMMERS REFERENCE:

S1, S2

Two HEX Code switches used for device address setting towards the IBM Channel.

S3

Single Addr.: In this position S1, S2 represent an eight bit address, S1: LSBs, S2: MSBs.
Multiple Addr.: In this position S1, S2 give an address area limited by the four MSBs, set by S1 (lower limit) and S2 (upper limit); both settings included.

S4

Used to bring the I/F ON LINE or OFF LINE. The SELECT indicator shows the actual transition, controlled by the IBM CH. I/F.

(CONTINUED)

- Power Consumption:

+5V: (CR80 system)	0.5A (typic)
+5V: (Isolated Power)	1.2A (typic)

PROGRAMMERS REFERENCE: (CONTINUED)

P1

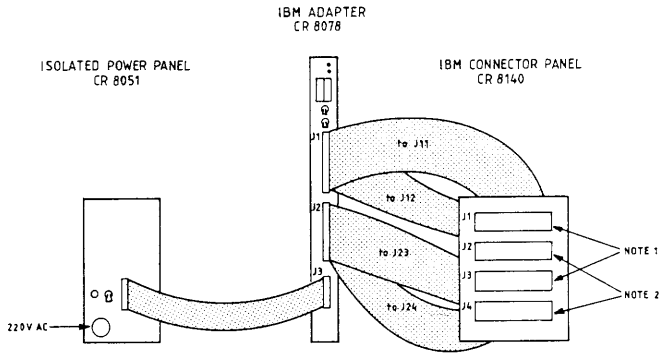
Priority Plug.

High Priority: The plug is normally mounted in the socket (pin 1 to pin 1 in Socket).

Low Priority: The plug is inversely mounted in the socket (pin 1 to pin 9 in Socket).

The priority is only significant when the channel interface works in daisy chain.

External Connections:



Note 1: Connection with IBM cables to the IBM channel.

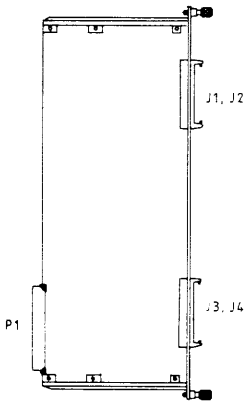
Note 2: Connection with IBM cables to the next control unit on the daisy chain or termination with IBM termination blocks.

The connectors J1 and J2 are connected via twisted pair flatcable to the rear of the IBM connector panel CR8140 (J1 is connected to J11 and J12, J2 to J23 and J24).

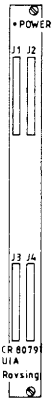
The front of the panel contains the IBM connectors: J1, J2 containing the Bus In/Out lines and J3, J4 containing the control (tags) lines. J1 and J3 are connected with IBM cables to the IBM channel. J3, J4 are either connected with IBM cables to the next control unit on the daisy chain or terminated with IBM termination blocks.

The connector J3 is connected via flatcable to the Isolated Power Panel CR8051.

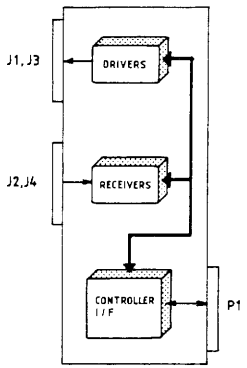
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM

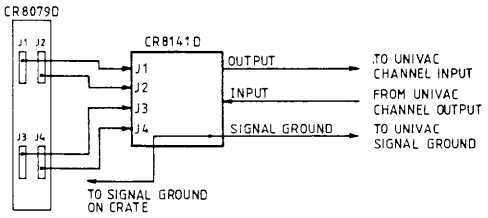


CHARACTERISTICS:

- Contains line drivers and receivers for a UNIVAC series 1100 I/O channel pair.
- Generates/checks parity and handles channel handshaking
- Operation controlled by the CR8037M UNIVAC I/F module.
- Power Consumption:
  +5V: 3.5A
- Size:
  Length: 155.0 mm
  Height: 412.0 mm
  Width: 34.3 mm (2M)
- Weight: 1.1 kg
- MTBF: 15,600 H/F

PROGRAMMERS REFERENCE:

The module is connected to the two UNIVAC I/O cables by means of the CR8141M UNIVAC CONNECTOR PANEL. The connection is as shown below:



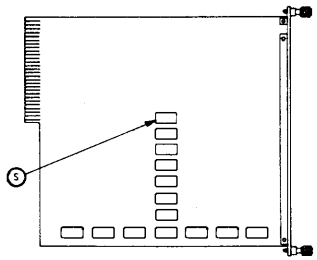
The two modules are connected by four 50-poled twisted pair faitcables.

J1 to J4 on the connector panel are mounted on the PCB on the rear side of the front panel.

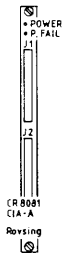
IMPORTANT: Before the I/O cables are connected, a connection between UNIVAC signal ground and signal ground on the CR80 must be established.

CIA-A CR8081M/010A-/00

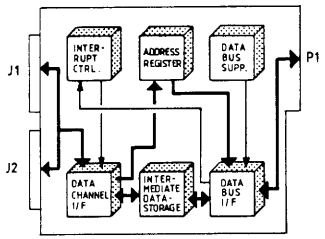
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



USED IN MAPPED SYSTEMS ONLY

CHARACTERISTICS:

- Crate Interface Adapter between Data Channel and Data Bus A of a Channel Unit (CU)
- Accesses modules in CU by handshaking procedure on request by Processor Unit (PU)
- Relays interrupts to PU
- Clock generation, Data Bus termination, and power detection
- Employs pulse transformers in Data Channel I/F to avoid ground current problems
- Access cycle (word)
  block trf.: 1000 ns (min)
  single trf.: 1250 ns (min)
  Maximum block size: 256 words.
- Physical Dimensions:

Height:	263.0 mm
Width:	17.1 mm
Depth:	280.0 mm
Weight:	600 g

- Power Comsumption:

5V:	3A

- MTBF: 71400 h

PROGRAMMERS REFERENCE:

The switch sets the Channel Unit (CU) No.

N0	N1	N2	N3

front panel

CU No. = 8*N3+4*N2+2*N1+N0

NX = 1, if switch Nx is open
NX = 0, if switch Nx is closed

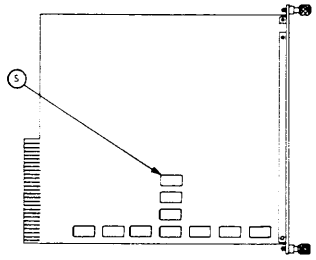
(N3N2N1N0 = 0000 is reserved for PU internal use)

Strap (SR1) positions:
A: 5.5V supply voltage
B: 5.0V supply voltage

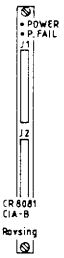
DATASHEET FOR:

CIA-B CR8081M/010-B/00

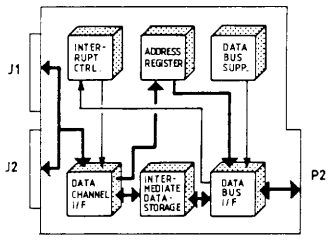
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



USED IN MAPPED SYSTEMS ONLY

CHARACTERISTICS:

- Crate Interface Adapter between Data Channel and Data Bus B of a Channel Unit (CU)
- Accesses modules in CU by handshaking procedure on request by Processor Unit (PU)
- Relays interrupts to PU
- Clock generation, Data Bus termination, and power detection
- Employs pulse transformers in Data Channel I/F to avoid ground current problems
- Access cycle (word)
  block trf.: 1000 ns (min)
  single trf.: 1250 ns (min)
  Maximum block size: 256 words.
- Physical Dimensions:

Heigth:	263.0 mm
Width:	17.1 mm
Depth:	280.0 mm

- Weigth: 600 g

PROGRAMMERS REFERENCE:

The switch sets the Channel Unit (CU) No.

N0	N1	N2	N3

front panel

CU No. = 8*N3+4*N2+2*N1+N0

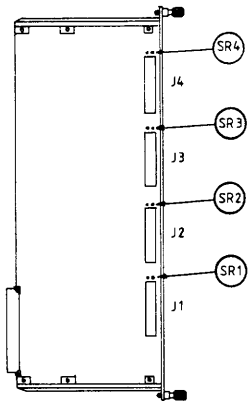
NX = 1, if switch Nx is open
NX = 0, if switch Nx is closed

(N3N2N1N0 = 0000 is reserved for PU internal use)

Strap (SR1) positions:
A: 5.5V supply voltage
B: 5.0V supply voltage

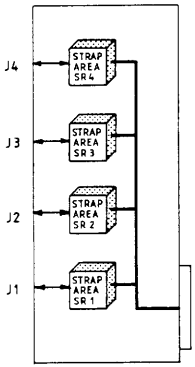
- Power Consumption:
  5V: 3A
- MTBF: 71400 h

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Non switching Line Interface Adapter for modems and terminals connector to LTU.
- Adapts 4 communication lines each 14 signals.
- All signals are fed through a strap area to meet most applications
- Output connectors are standard DP25P connectors
- Sense signal to LTU indicating existing LIA.
- Dimensions:

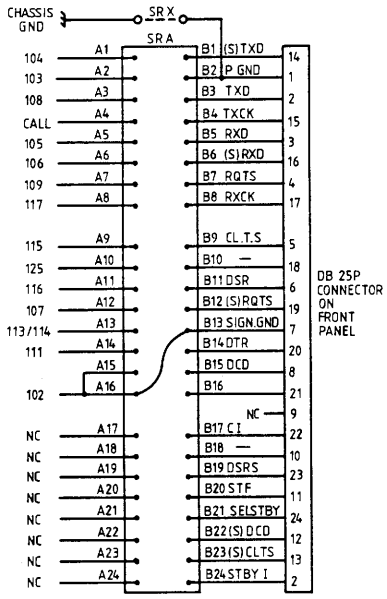
Height:	375 mm
Width:	17.1 mm
Length:	152 mm
Weight:	0.5 kg

MTBF: 10,000,000 hours

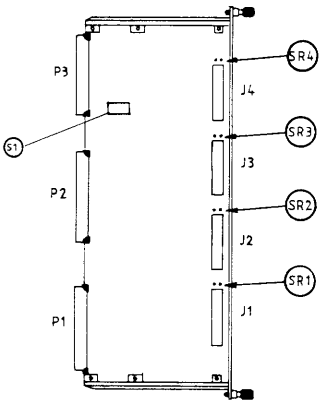
PROGRAMMERS REFERENCE:

Strap Area and DB 25 P Connector Layout

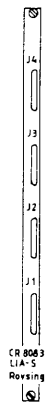
SR1, 2, 3 and 4 can strap protective ground (circuit No. 111) to CR80 frame ground individually for each channel.



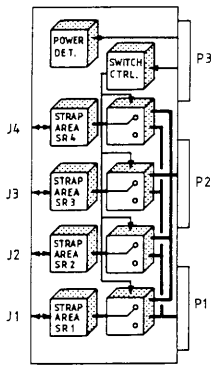
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Switching Line Interface Adapter for modens and terminals connector to LTU.
- Adapts 4 communication lines each 14 signals
- Fulfills requirements of RS 232 and V 28
- Intended for use in highly reliable CR80 systems with 1 out of n redundant LTU
- Solid state switches are used to assure high system reliability and to fulfill overload requirements
- Dual Power supplied from the CR80D power busses.
- All signals are fed through a strap area to meet most applications.
- Output connectors are standard DB25 connectors.
- Sense signal to LTU indicating existing LIA.

PROGRAMMERS REFERENCE:

Switch Setting:

S1

1	2	3	4	5	6	7	8
A0	$\overline{A0}$	A1	$\overline{A1}$	A2	$\overline{A2}$	A3	$\overline{A3}$

Front Panel

Module Address Setting:

Address determined by A0-A3

A3 MSB

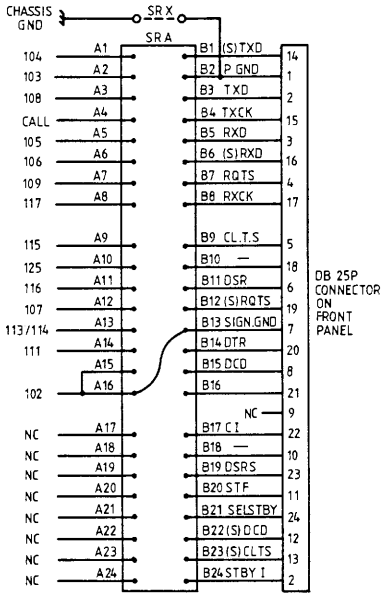
(CONTINUED)

- Dimensions:
  - Height: 375 mm
  - Width: 17.1 mm
  - Length: 152 mm
  - Weight: 0.6 kg
- MTBF, equivalent: 3,500,000 hours

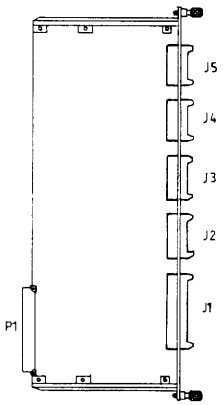
PROGRAMMERS REFERENCE: (CONTINUED)

Strap Area and DB 25 P Connector Layout

SR1, 2, 3 and 4 can strap protective ground (circuit No. 111) to CR80 frame ground individually for each channel.

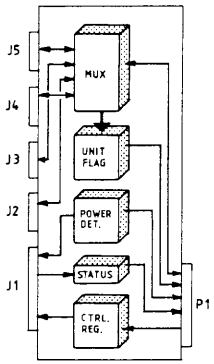


SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

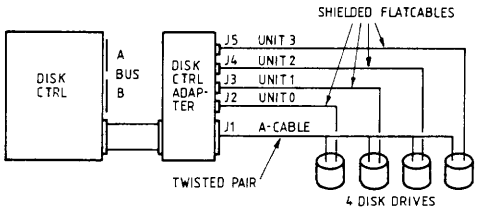
- Adapter for Disk CTRL CR8044
- Interface to 1 to 4 disk drives in combined star/daisy chain connection
- Drives selected from CDC's SMD, MMD and CMD families in any combination
- Differential line drivers and receivers for twisted pair signals to/from drives.
- Buffers/Receivers for signals to/from Disk CTRL
- Multiplexers for individual drive signals
- Individual interrupt generation for each drive:

  seek over
  drive not busy

- Drive(s) disabled by power or clock failure in Disk CTRL or adapter
- Data transfer rate:
  10 M bits per sec.

(CONTINUED)

PROGRAMMERS REFERENCE:



The B-cables have to be connected to disk drives with corresponding number, see table below:

Connector Number:	Drive Number:
J5 UNIT3	3, 7, 11 or 15 (ØF)
J4 UNIT2	2, 6, 10 or 14 (ØE)
J3 UNIT1	1, 5, 9 or 13 (ØD)
J2 UNIT0	1, 4, 8 or 12 (ØC)

The A-cable has to be terminated in the last drive of the daisy chain.

CHARACTERISTICS: (CONTINUED)

- Dimensions:

Width:	17.1 mm
Length:	182.0 mm
Heigth:	412.0 mm
Weight:	0.54 kg

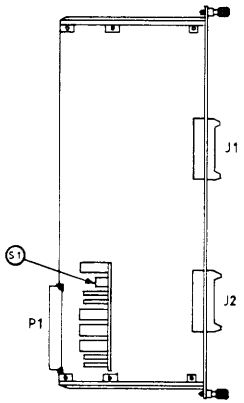
- Power:

+12V: not used
+5V: 1.2A
-12V: 0.15mA

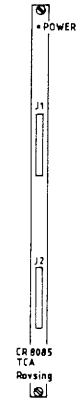
The power is supplied from the DISK 'RL.
CTRL.

- MBTF: 47000 h

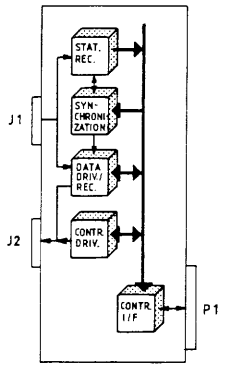
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Interface between Tape Transport/ Formatter and Tape Controller.
- Connection of up to 8 daisy chained Tape Transports selected from Pertec (F)T9000 series.
  Optionally the series (F)T6000, (F)T7000, (F)T8000 and (F)T1000 can be used
- Synchronization of data
- Power Comsumption:
  +5V: 1.5A
- Mechanical Dimensions:
  Length: 164.0 mm
  Height: 412.0 mm
  Width: 17.1 mm (1M)
- Weigth: 550 g
- Mean Time Between Failure:
  $128 \cdot 10^3$ h.

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

S1

1	2	3	4
B0	B1	B2	B3

Front Panel ←

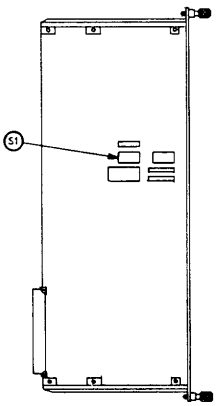
B0 — Indicates the data transfer rate of the tape transports connected to the adapter:
OPEN: 200 Kchar/sec.
CLOSED: less then 200 Kchar/sec.

B1 — Indicates whether to generate/check for odd or even parity when transferring data to/from tape:
OPEN: odd parity
CLOSED: even parity

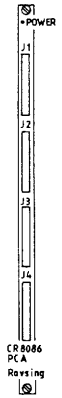
B2 — Indicates the type of transports connected to the adapter:
OPEN: (F)T9000 or (F)T1000
CLOSED: (F)T6000, (F)T7000 or (F)T8000

B3 — Not used

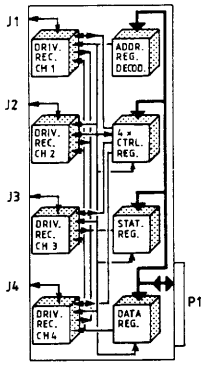
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

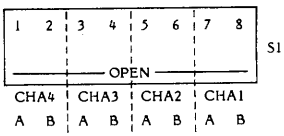
- Parallel controller adapter for CR80
- Provides the interface between the parallel controller CR8046 and four peripheral units with parallel data transfer.
- Handshaking of signals between the PCA and the four outputs depends on firmware in the parallel controller.
- The PCA contains 4 registers, the data register (write only), the address register (read-write), the control register (read-write) (one for each unit), and the status register (read only).
- Firmware for line printer application is available for the parallel controller CR8046.
- The line printer application allows selection between two different printer types for each output on PCA (CDC or Data Products) option is made by S1.
- MTBF: 185,000 h

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Setting

The switch S1 is used to select between two printer types for each channel: CDC and Data Products. Only the A-switches are used to this selection. The B-switches are not used.



CHA4:	1	2	PRINTER TYPE
	OPEN	X	CDC
	CLOSED	X	DATA PRODUCTS
CHA3:	3	4	PRINTER TYPE
	OPEN	X	CDC
	CLOSED	X	DATA PRODUCTS
CHA2:	5	6	PRINTER TYPE
	OPEN	X	CDC
	CLOSED	X	DATA PRODUCTS
CHA1:	7	8	PRINTER TYPE
	OPEN	X	CDC
	CLOSED	X	DATA PRODUCTS

Switch A = OPEN → IVB1 = "0" (when read status reg.)
Switch B = OPEN → IVB0 = "0" (when read status reg.)

CHARACTERISTICS: (CONTINUED)

- Power Consumption:
  +5V: Typ. 1.5A

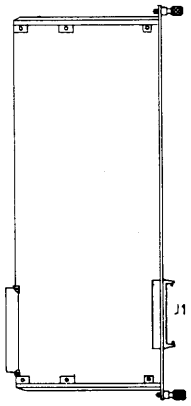
- Mechanical Dimensions:
  Height: 412.0 mm
  Width: 17.1 mm (1M)
  Length: 155.0 mm

- Weight: 0.6 kg

DATASHEET FOR:

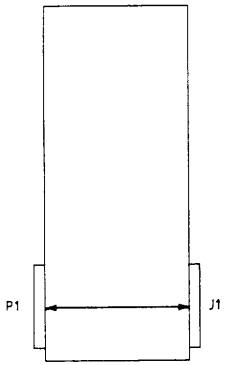
SFA CR8087M/XXX--/XX

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

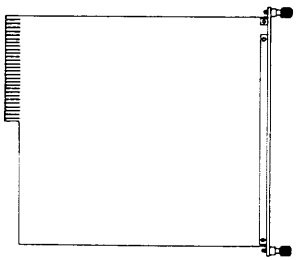
- Standard Floppy disk controller adapter for the CR80 system
- Interfaces Floppy Disk controller (dual or single bus) to floppy drives.
- Standard 50 lead flat cable connection between drive and SFA.
- MTBF: 10,000,000 hours
- Size: 412.6 x 155 x 17
- Weight: 0.4 kg
- Power Consumption: 0A

PROGRAMMERS REFERENCE:

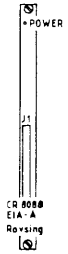
(not applicable)

EIA-A CR8088M/010A-/00

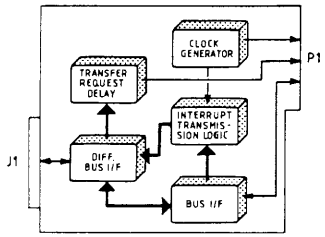
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Active Main Bus Extension for the CR80 system together with EPM and EPA modules in unmapped systems.
- Interface between the Differential Flat Cable Bus from one EPA in a Processor Crate & Bus A in an extension Crate.
- All Main Bus signals are transfered from the Processor Crate through an EPM, EPA and EIA-A module
- However, the two Main Bus clocks Ø1 and Ø2 are generated on the module
- Main Bus termination included
- Rear mounted: the EIA-A does not take up module space in the extension Crate.
- Mechanical Dimensions:

Height:	263.0 mm
Width:	17.1 mm (1M)
Length:	270.0 mm

- Power Consumption:

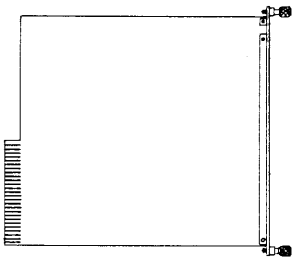
+5V:	1.6A typ.

- MTBF: 114000 (H/F)
- Weight: 500 g

PROGRAMMERS REFERENCE:

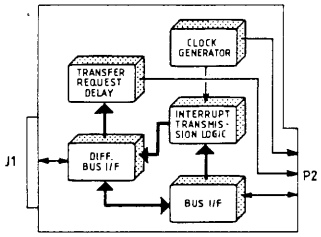
(not applicable)

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Active Main Bus Extension for the CR80 system together with EPM and EPA modules in unmapped systems.

o Interface between the Differential Flat Cable Bus from one EPA in a Processor Crate & Bus B in an extension Crate.

- All Main Bus signals are transfered from the Processor Crate through an EPM, EPA and EIA-B module

- However, the two Main Bus clocks Ø1 and Ø2 are generated on the module

- Main Bus termination included

o Rear mounted: the EIA-B does not take up module space in the extension Crate.

- Mechanical Dimensions:
  Height: 263.0 mm
  Width: 17.1 mm (1M)
  Length: 270.0 mm

- Power Consumption:
  +5V: 1.6A typ.

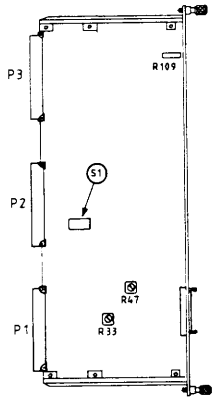
- MTBF: 114000 (H/F)

- Weight: 500 g

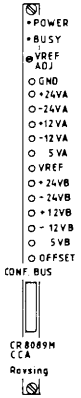
PROGRAMMERS REFERENCE:

(not applicable)

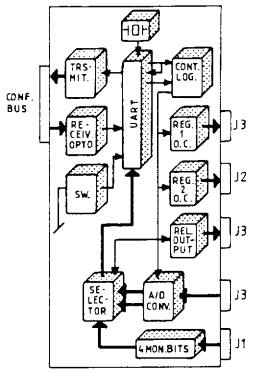
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

Crate Configuration Adapter

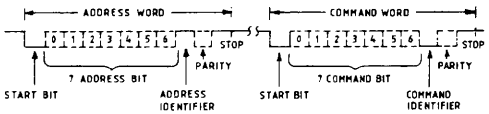
- Switching module in the watchdog system, placed in CU-Crates and PU-Crates.
- Communication to/from the CCA takes place on the configuration Bus a 16 p flatcable, using serial communication at 4800 baud.
- The CCA is capable of switching two banks of each up to 8 LTU's + 1 spare or one bank of up to 16 LTU's + 1 spare. The CCA is also able to switch 4 other devices, monitoring 14 analogue Channels (e.g. all Crate Power Supply voltaes) and 4 inputs signals.
- The CCA is a slave module under control of either the WD-CPU or the WPC (depending on mode AUTO/MANUAL).
- Up to 32 CCA's can be placed on a configuration bus.

Length = 152,5mm
Height = 375,5mm
Width = 17,4mm

(CONTINUED)

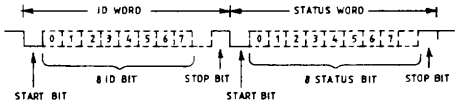
PROGRAMMERS REFERENCE:

Transmission to the CCA:



When the CCA recognize the address and accept the following word as a command, it will transmit the two words shown below automatically:

Transmission from the CCA:



The 8 status bit contain the just received command bits.
The 8 ID bit contain the 8 msb of the A/D conversion, or the 2 lsb and the four monitoring bits.

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

S1 is a 6 bit switch containing the Address of the CCA

1	2	3	4	5	6
A0	A1	A2	A3	A4	X

Only five of the six bit are used:

CCA addresses: 0-31

On "0"
Off "1"

R33 and R109 is potentiometers for adjusting the Voltage Reference of the A/D converter. Vref = 10.24V. R33 is course adjustment and R109 fine adjustment of only a few mV.

R47 is a potentiometer for adjusting the offset voltage of the A/D converter. Voffset = 2.56V.

CHARACTERISTICS: (CONTINUED)

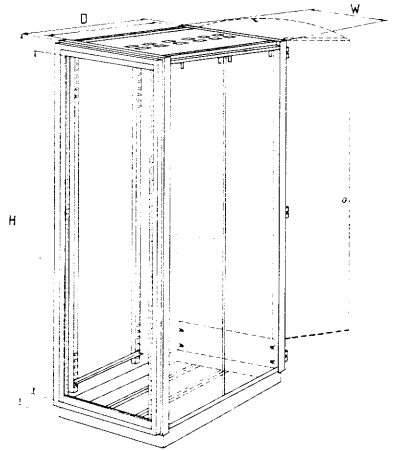
- Power Requirement

+12V	max 250mA	A or B
-12V	max 100mA	A or B
+24V	2mA	A and B
-24V	2mA	A and B
5V	2mA	A and B

- MTBF: 29700 h

DATASHEET FOR:

RACK CR8101-/X42--/00



CHARACTERISTICS:

- Dimensions:

  H: 2062.5 mm (overall), 1868.5 mm = 42U (useful)

  W: 597 mm (overall), 19" (useful)

  D: 1040 mm (overall), 1010 mm (useful)

  Weight: 106 kg

- Available in two versions:

  Without side panels:
  CR8101-/142--/00

  With side panels:
  CR8101-/042--/00

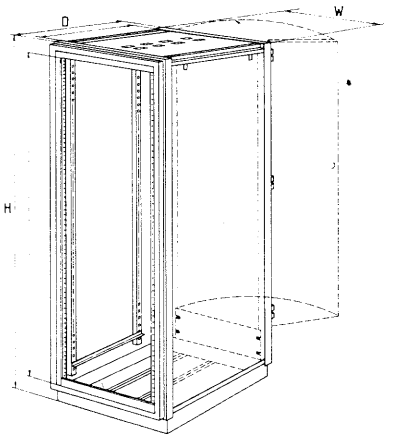
- Supporting bars for installation of 7 units.

- Depth of rack allows rack mounting of disk drive.

- More racks may be assembled side by side using mech. kit CR parts No.: 5.31712-00.

DATASHEET FOR:

RACK CR8101-/X36--/00



CHARACTERISTICS:

- Dimensions:

  H: 1795.5 mm (overall), 1602 mm = 36U (useful)

  W: 597 mm (overall), 19" (useful)

  D: 728 mm (overall), 700 mm (useful)

  Weight: 77 kg

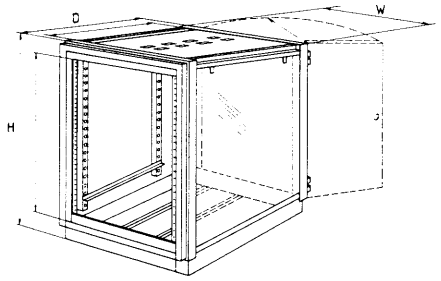
- Available in two versions:

  Without side panels:
  CR8101-/136--/00

  With side panels:
  CR8101-/036--/00

- Supporting bars for installation of 7 units.

- More racks may be assembled side by side using mech. kit CR parts No.: 5.31712-00.



CHARACTERISTICS:

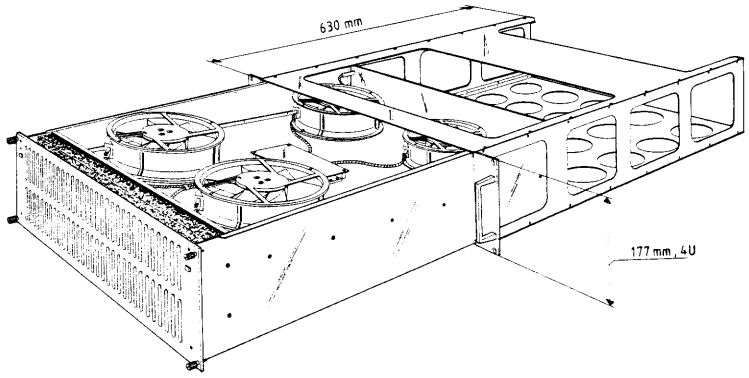
- Dimensions:
  - H: 995.5 mm (overall), 802 mm = 18U (useful)
  - W: 597 mm (overall), 19" (useful)
  - D: 728 mm (overall), 700 mm (useful)
  - Weight: 44 kg
- Available in two versions:

  Without side panels:
  CR8102-/100—/00

  With side panels:
  CR8102-/000—/00
- Supporting bars for installation of 7 units.

DATASHEET FOR:

FAN UNIT (M) CR 8105M/020- -/00



CHARACTERISTICS:

- Used with PU and CU CR80 M-Crates.
- Dual Mains supply.
- Easy replacement of filter cushion.
- Exceptional dust accumulation capacity in filter cushion.
- Dimensions:

H:	177 mm (4U)
W:	482 mm (19" Rack)
D:	630 mm
Weight:	30 kg

DETAILS

Data:

Power Input: 2 x 220V, 50/60 Hz
2 X 110W
Fuse: 2 x 1A SB.
Air Flow Capacity (no load): 20 m^3/min.

Calculated MTBF: 197 years

Filter cushion dimensions: 165 x 408 x 15 mm
CR parts No.: 7.17401-00

Description

The FAN UNIT employs two electrically independent systems (A and B), each consisting of 4 axial fan motors with ball-bearings. Each system is capable of cooling the associated M-crate. The two systems are located in two seperate floors and the air passes both systems even if one system is passive.

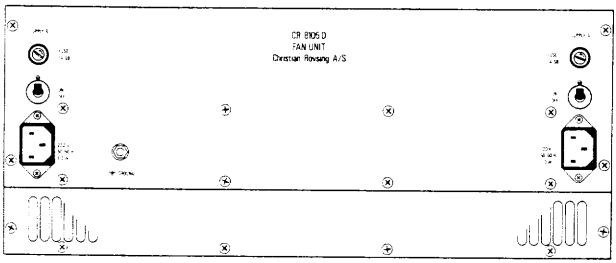
For each system a POWER-ON lamp is located on the front. Power plug, ON-OFF switch and fuse are located on the rear side of unit.

(CONTINUED)

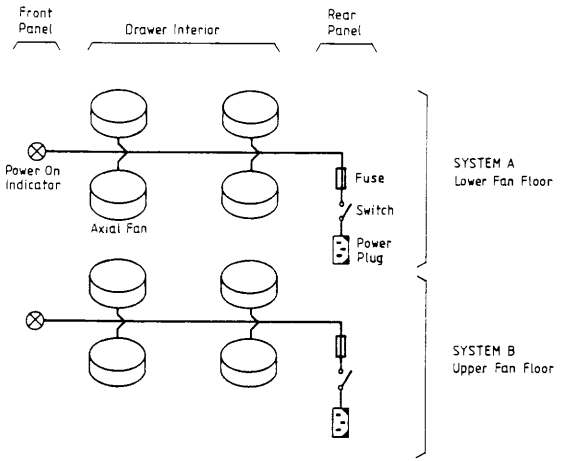
DETAILS: (CONTINUED)

The FAN UNIT consists of a permanently mounted metal frame and a drawer which carries electrical circuits, fan motors and the filter cushion. Upper side clearances in the frame allow the air to dissipate vertically into front and rear magazine of the M-crate.

The filter cushion may be replaced by unlocking the drawer and pulling it forward. In that position the filter cushion is accessible without tools and may easily be removed. Filter cushion replacement can take place in 10 seconds without switching off the power supply.



Rear View

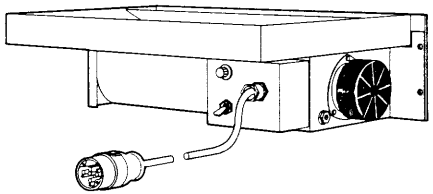


System Schematics

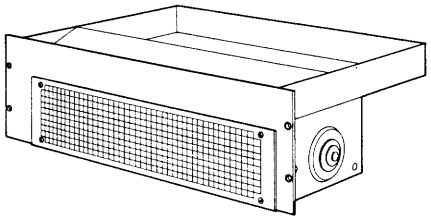
DATASHEET FOR:

FAN UNIT (S) CR 8105S/010--/00

REAR VIEW



FRONT VIEW



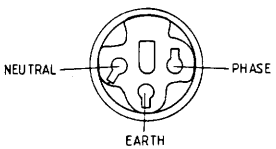
CHARACTERISTICS:

- FAN unit for S-crate and TDX-crate.
- Airflow:
- Dimensions:
  H: 133 mm (3U)
  W: 482 mm (19" Rack)
  D: 300 mm
  Weight: 4.5 kg
- Power input:
  220VAC, 50/60Hz
  40W
  Fuse: 2A SB
- Fan Life: 20,000 hours

INSTALLATION INSTRUCTION:

The mains power plug corresponds to power distribution panel.

MALE PLUG

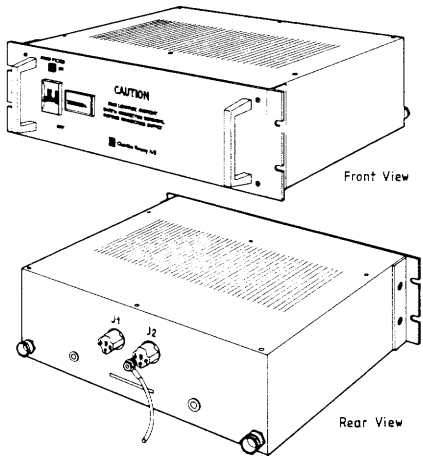


Air filter has to be changed during preventive maintenance. Interval dependent of environmental conditions.

The FAN unit is fixed by means of four screws just below the crate and should rest on supporting bars.

DATASHEET FOR:

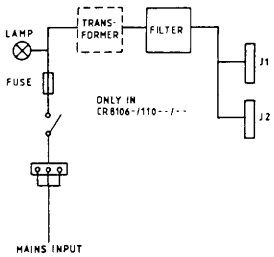
MAINS FILTER CR8106-/XXX--/00



Front View

Rear View

BLOCK DIAGRAM:



CHARACTERISTICS:

- Filter and circuit breaker for a number of CR80 units.
- Normally one mains filter for each rack.
- Two power distribution panels can be connected.
- Available for various inputs as defined below:

  CR8106-/220—/00
  220VAC, 6A, 50Hz

  CR8106-/240—/00
  240VAC, 6A, 50Hz

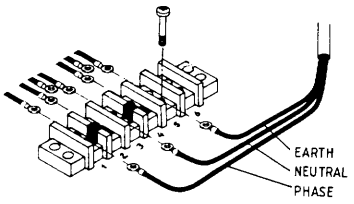
  CR8106-/110—/00
  110VAC, 10A, 60Hz

- Dimensions:
  H: 133 mm (3U)
  W: 482 mm (19" Rack)
  D: 350 mm
  Weight: 10 kg

INSTALLATION INSTRUCTION:

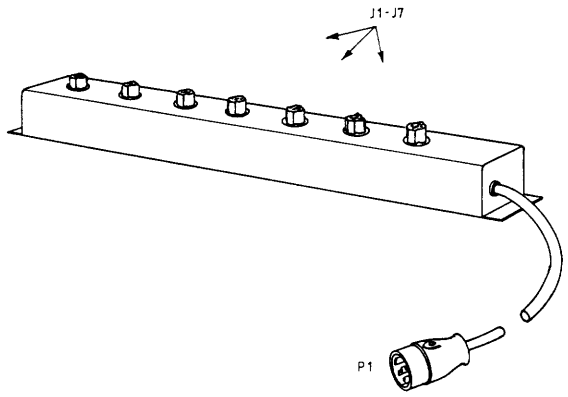
The filter is located either in top or bottom of the rack, rests on supporting bars and is fixed by four screws.

Mains input cable is connected by screw-terminals as shown below. The terminals are located behind a cover plate.



DATASHEET FOR:

POWER DISTRIBUTION PANEL CR8107-/010--/00



CHARACTERISTICS:

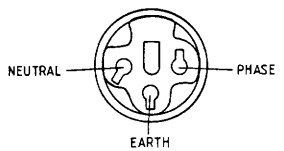
- Distribution of mains supply within rack.
- Designed to fit into the standard racks.
- Seven outputs available.
- Plug compatible with standard power cables and mains filters.
- Dimensions:
  H: 54 mm
  W: 482 mm (19" Rack)
  D: 50 mm
  Weight: 1.5 kg
- Max. power:
  220V/240V, 10A

INSTALLATION INSTRUCTION:

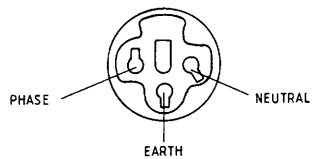
The panel is located at the rear of the rack in a height as determined by the application and fixed by four screws.

Connector specification:

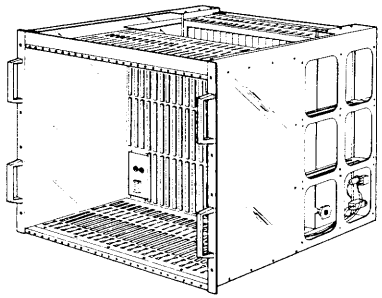
P1: MALE



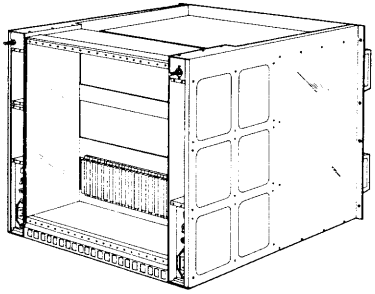
J1-J7: FEMALE



COMBI CRATE for PU & CU CR 8112M/112PA/00



Front View



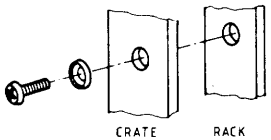
Rear View

CHARACTERISTICS:

- For use in memory mapped non-faulttolerant systems.
- Capable of housing 1 PU and 1 CU with a power supply plus eight front modules in each unit.
- Rear crate can be tipped to allow access to internal cabling.
- P-Bus, C-Bus and Data Bus A implemented as PCB.
- Dimensions:
  H: 10U~ 445 mm
  W: 19"~ 485 mm
  D: 620 mm
  Weight: 26 kg
- MTBF: 230,000 hours

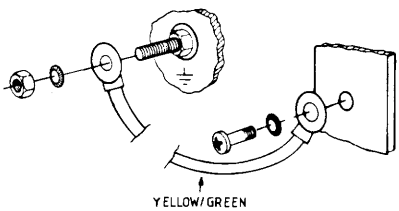
INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from the front of the rack and fixed by means of six screws, see below:



The crate must rest on supporting bars and a fan unit (CR8105M) must be located just below.

The protection GND terminal (chassis) should be connected to the rack by means of 6 mm^2 wires.

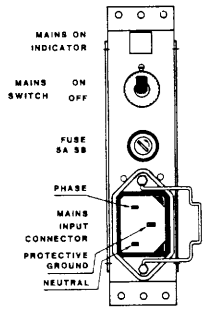


(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

The DC ground terminal is for connection to other units as required by the application.

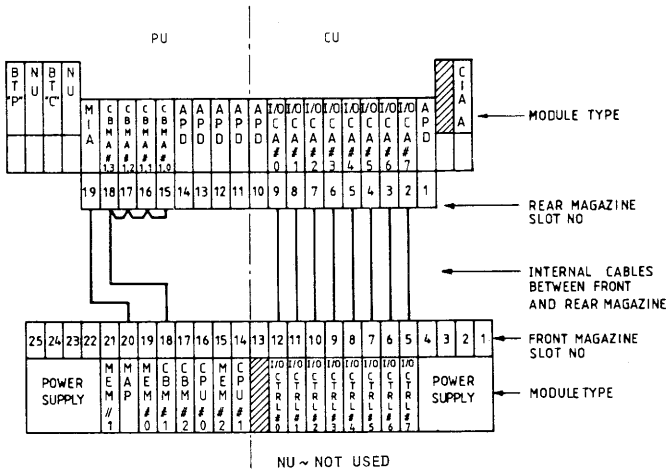
Two mains power inputs as defined below are available, one for each power supply. The crate mains inputs are connected to the power distribution panel by means of the power cable CR8201M/015--/00.



The inter-crate cabling supports the configuration possibilities defined in the slot specification below.

If required, the basis configuration can be adapted to special applications by procurement and installation of additional internal cables.

Slot Specifications:



(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

Notes to figure:

1) When configurating the crate, the power comsumption should be calculated and checked against the power supply outputs.

2) For a minimum configuration of the COMBI Crate (No I/O CTRL) the following modules have to be procurred: BT 'P', BT 'C' (Bus Termination Modules), MAP + MIA (Memory Mapping Module plus data channel I/F), CPU, MEM (RAM), two Power Supplies and CIA-A (CU data channel I/F).

3) For the remaining slot positions not specified in 2) above, the following rules apply:

Front Magazine: Slots 21 and 15 can only be used for memory modules.

Slots 18 and 17 can be used for C-Bus modules (CBM) or memory modules, if slot 17 is to be used for a CBM, then the internal cable has to be procured separately.

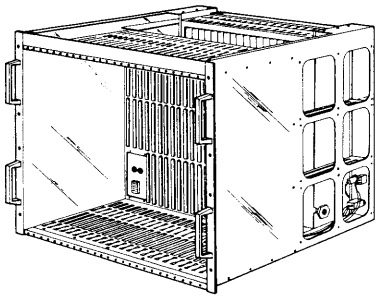
Slot 14 can be used for a CPU or memory.

Slots 5 - 12 can be used for single bus peripheral CTRLs, memory modules or a mixture of these products.

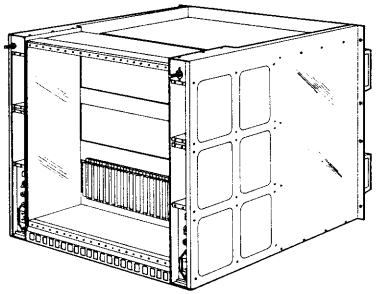
Rear Magazine: Slots 15 - 18 are prepared for adapter modules connected to CBM No. 1.

Slots 1 and 10 - 14 can be used for special purposes.

Slots 2 - 9 are available for adapter modules for the peripheral CTRLs.



Front View



Rear View

CHARACTERISTICS:

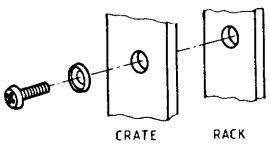
- For the use in mapped systems.
- Capable of housing 2 PUs, each with one power supply plus eight front modules in each unit.
- Rear crate can be opened to allow access to the internal cabling.
- P-Bus and C-Bus implemented as PCB.
- Dimensions:

H:	10 U	~ 445 mm
W:	19"	~ 485 mm
D:		620 mm
Weight:		26 kg

- MTBF: 230,000 hours

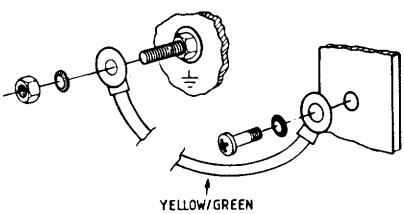
INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from the front of the rack and fixed by means of six screws, see below:



The crate must rest on supporting bars and a fan unit (CR8105M) must be located just below.

The protection GND terminal (chasis) should be connected to the rack by means of 6 mm^2 wires.

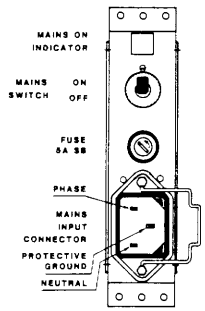


(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

The DC ground terminal is for connection to other units as required by the application.

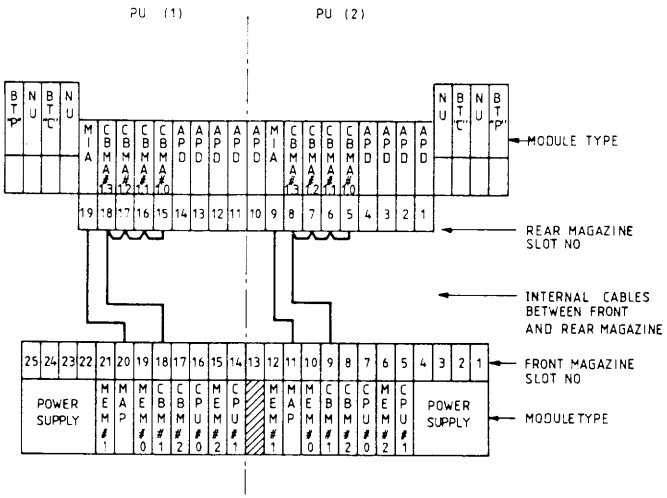
Two mains power inputs as defined below are available, one for each power supply. The crate mains inputs are connected to the power distribution panel by means of the power cable CR8201M/015--/00.



The inter-crate cabling supports the configuration possibilities defined in the slot specification below.

If required, the basis configuration can be adapted to special applications by procurement and installation of additional internal cables.

Slot Specifications:



NU ~ NOT USED

(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

Notes to figure:

1) When configurating the crate, the power consumption should be calculated and checked against the power supply output.

2) For a minimum configuration of the crate, the following module set has to be procured for each PU:

   BT 'C', BT 'P' (Bus Termination Modules), MAP MIA (Memory Mapping Modules plus Data Channel Interface), CPU, MEM 0 (RAM), Power Supply.

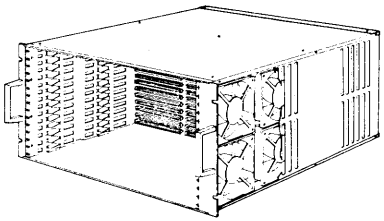
3) For the remaining slot portions not specified in 2) above, the following rules apply:

   Front Magazine: Slots 21, 15, 12 and 6 can only be used for memory modules.

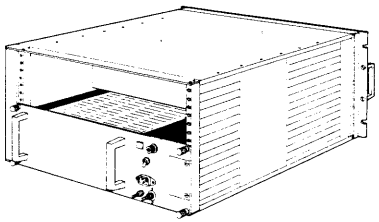
   Slots 18, 17, 9 and 8 can be used for C-Bus modules (CBM) or memory modules. If slot 17 and 8 have to be used for a CBM, then the internal cables have to be procured separately.

   Slots 14 and 5 can be used for CPUs or memory modules.

   Rear Magazine: Slots (5 to 8) and (15 to 18) are prepared for interfacing C-bus module adapters (four for each CBM).



Front View



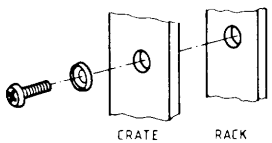
Rear View

CHARACTERISTICS:

- For use in non memory mapped systems (single bus).
- Power Supply, Fan and Bus Termination included.
- Capable of housing 15 front modules and 7 rear modules.
- Available DC power:
  5V: 50A
  +/-12V: 8A
- Mains Power Input
  220V +/- 20% or 110V +/- 20%
  45Hz - 440 Hz
- Dimensions:
  H: 6U 270 mm
  W: 19" 485 mm
  D: 620 mm
  Weight: 31 kg
- MTBF: 26,300 hours

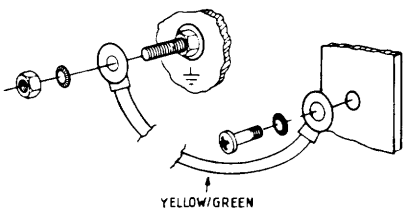
INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from the front of the rack and fixed by means of four screws, see below:



The crate should rest on supporting bars.

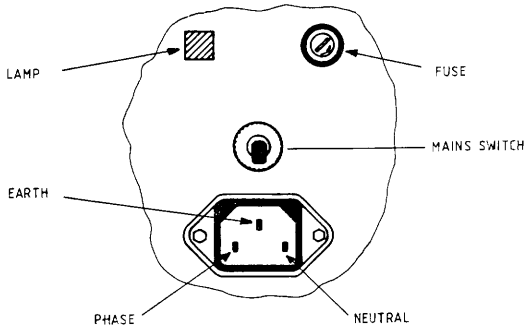
The protective GND terminal (chassis) should be connected to the rack by means of 6 mm^2 wires, see below:



(CONTINUED)

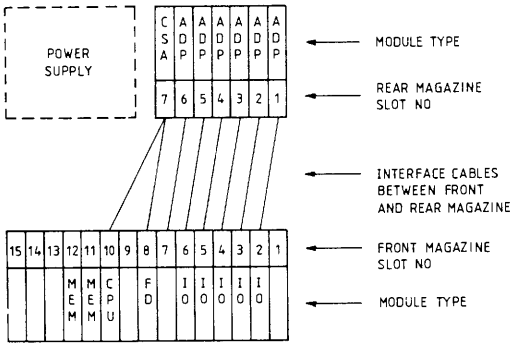
INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

The crate's mains power input is as defined below:



The intra-crate cabling supports the configuration possibilities as defined in slot specifications below. If required, the basis configuration can be adapted to special application by procurement and installation of additional intra-crate cables.

Slot Specification:

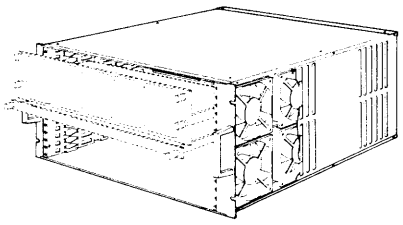


Notes to figure above:

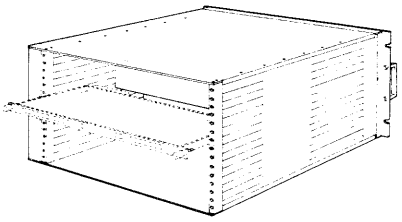
CPU ~ CPU/SCM
MEM ~ Memory
FD ~ Floppy Disc Ctrl. or other I/O Ctrl.
IO ~ I/O Ctrl.
CSA ~ CPU/SCM adapter
ADP ~ I/O Ctrl. adapter
(no modules included)

DATASHEET FOR:

WATCHDOG CRATE CR 8115M/005PC/00



Front View



Rear View

CHARACTERISTICS:

For use as frame for Watchdog system modules:
WD CPU
WD Panel CTRL

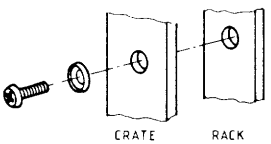
- Dual busses included to allow for connection of two external DC sources.
- Dimensions:

H:	6U	~	270 mm
W:	19"	~	485 mm
D:			620 mm
Weight:			23 kg

- MTBF: 50,000 hours

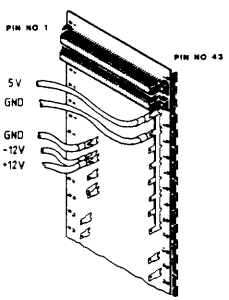
INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from the front of the rack and fixed by means of four screws, see below:



The crate should rest on supporting bars.

The DC supplies are connected directly to the two PCB busses by means of spade plugs, see below:



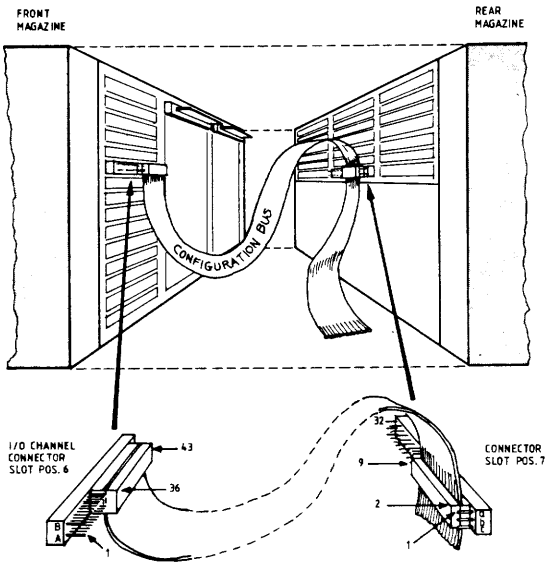
The two bus board power connections are identical.

(CONTINUED)

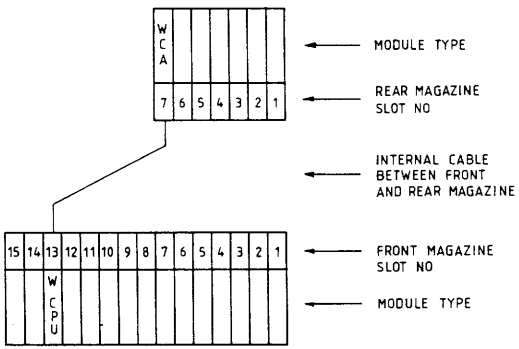
INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

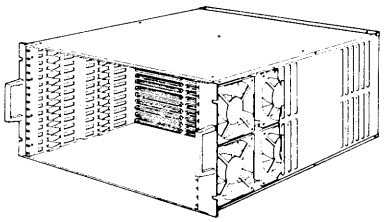
The intra-crate cabling implemented to support the Watchdog system is defined in slot specifications below. If required, the basis configuration can be adapted to special applications by procurement and installation of additional crate cables.

The configuration bus cable is connected to the crate by means of two connectors as shown below:

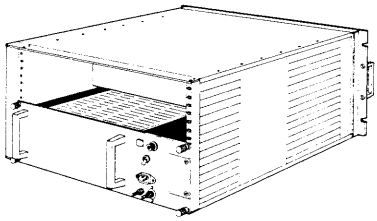


Slot Specification:





Front View



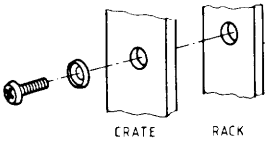
Rear View

CHARACTERISTICS:

- For use in
  a) memory mapped systems as processor unit
  b) non memory mapped systems when Channel Bus modules are to be used.
- Power supply, fan and bus termination included.
- Capable of housing 15 front modules and 7 adapters.
- Available DC power:
  5V: 50A
  +/-12V: 8A
- Mains power input:
  220V +/- 20% or 110V +/- 20%
  45Hz - 440 Hz
- Dimensions:
  H: 6 U ~ 270 mm
  W: 19" ~ 485 mm
  D: 620 mm
  Weight: 32 kg
- MTBF: 24,900 hours

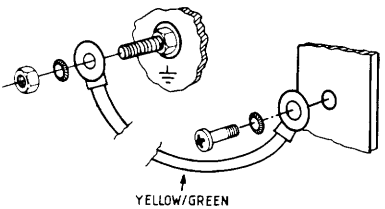
INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from the front of the rack and is positioned by four screws.



The weight should be relieved by rack-mounted sliding bars.

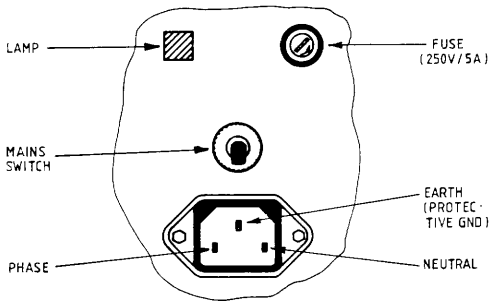
The protective ground terminal should be connected to the rack by means of 6 mm^2 wires.



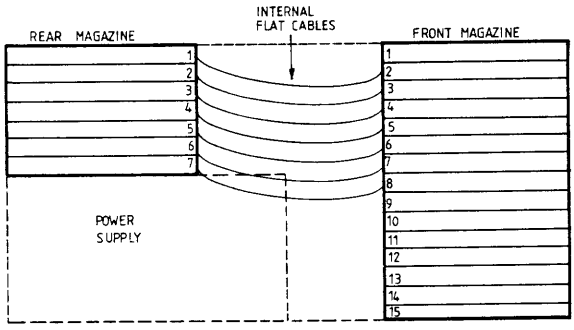
(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

The mains power input is defined as follows:



Slot Specification:

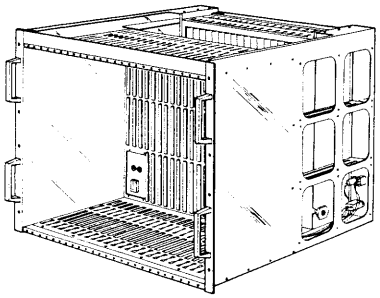


In non memory mapped systems, the two Main Busses should be connected by an extension flat cable assembly, which must be procured separately.

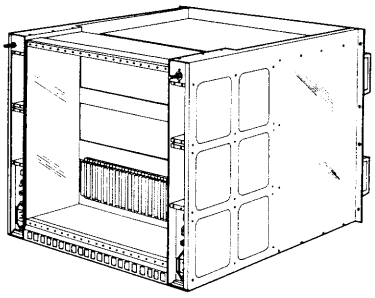
In systems with more than one address sourcing module, bus arbitration signals must be wire-wrapped.

DATASHEET FOR:

PU CRATE Dual Bus CR 8125M/ 225PC/ 00



Front View



Rear View

CHARACTERISTICS:

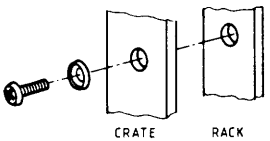
- For use in memory mapped systems
- Capable of housing 1 PU with up to 17 front modules and 19 adapter modules.
- Rear crate can be tipped to allow access to internal crate cabling.
- P-Bus and C-Bus implemented as PCB.
- Prepared for two power supplies operating in parallel.
- Dimensions:

H:	10U	~	445 mm
W:	19"	~	485 mm
D:			620 mm
Weight:			26 kg

- MTBF: 230,000 hours

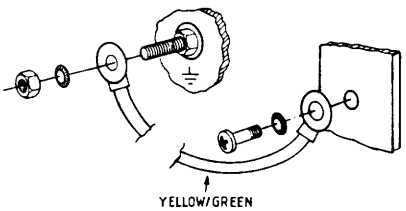
INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from the front of the rack and fixed by means of six screws, see below:



The crate must rest on supporting bars and a fan unit (CR8105M) must be located just below.

The protection GND terminal (chassis) should be connected to the rack by means of 6 mm^2 wires.

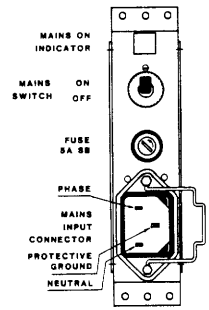


(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

The DC ground terminal for connection to other units as required by the application.

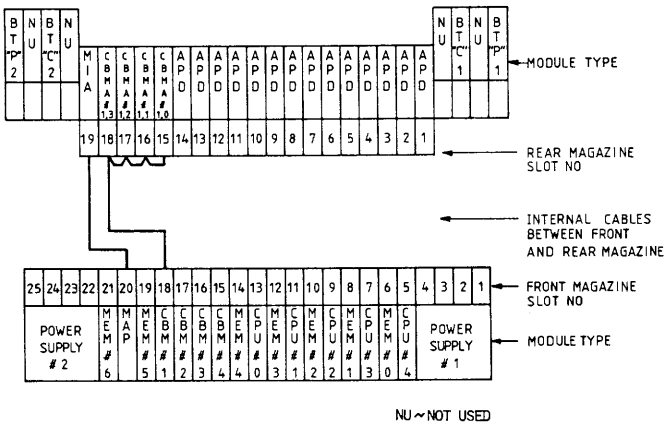
Two mains power inputs as defined below are available, one for each power supply. The crate mains inputs are connected to the power distribution panel by means of the power cable CR8201M/015—/00.



The inter-crate cabling supports the configuration possibilities defined in the slot specification below.

If required, the basis configuration can be adapted to special applications by procurement and installation of additional internal cables.

Slot Specifications:



(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

Notes to figure above:

1) When configurating the crate, power consumption should be calculated and checked against the available power supply outputs:

2) For a minimum configuration of the PU crate following modules have to be procured: BT 'P', BT 'C' x2 (four bus termination modules), MAP + MIA (Memory Mapping Module plus Data Channel I/F), CPU, MEM (RAM) and Power Supply.

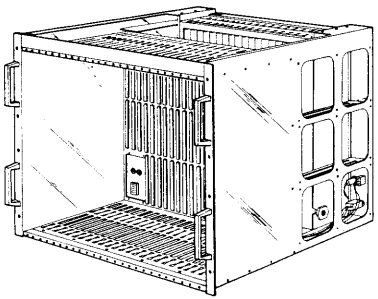
3) For the remaining slot positions not specified in 2) above, the following rules apply:

Front Magazine:
Slots 22 - 25 for additional Power Supply.
Slots 19, 21 for memory modules only.
Slots 6, 8, 10, 12, 14 for memory modules, the I/O area (J3) is not usable, but in special application J3 could be made available for interfacing to an adapter module.
Slots 15 - 18 can be used for C-Bus modules (CBM) or memory modules, if other than slot 18 is used for CBM, the cables have to be procurred reparately.

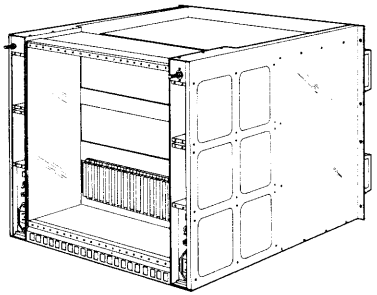
Slots 5, 7, 9, 11 can be used for additional CPU modules or memory modules.

Rear Magazine:
Slots 15 - 18 are prepared for adapter modules connected to CBM # 1.
Slots 1 - 14 are available for application defined purposes.

PU CRATE Single Bus CR8125M/125P-/00



Front View



Rear View

CHARACTERISTICS:

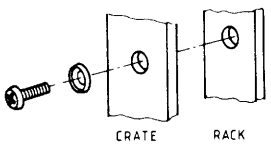
- For use in non-memory mapped systems for combined Processor and I/O systems.
- Capable of housing up to 17 front modules and 19 adapter modules.
- Rear crate can be tipped to allow access to internal crate cabling.
- The P-Bus implemented as PCB.
- Prepared for two power supplies operating in parallel.
- Dimensions:

  H: 10 U ~ 445 mm
  W: 19" ~ 485 mm
  D: 620 mm
  Weight: 26 kg

- MTBF: 350,000 hours

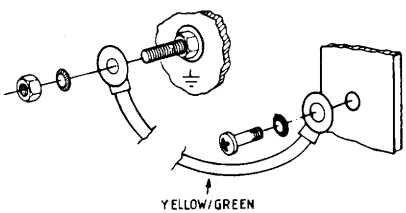
INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from the front of the rack and fixed by means of six screws, see below:



The crate must rest on supporting bars and a fan unit (CR8105M) must be located just below.

The protection GND terminal (chassis) should be connected to the rack by means of 6 mm^2 wires.

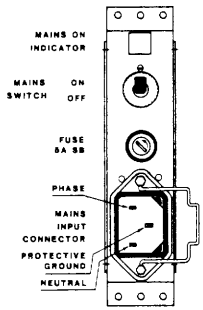


(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

The DC ground terminal is for connection to other units as required by the application.

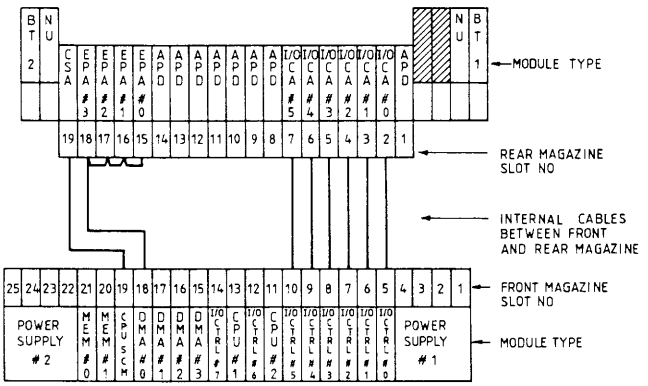
Two mains power inputs as defined below are available, one for each power supply. The crate mains inputs are connected to the power distribution panel by means of the power cable CR8201M/015--/00.



The inter-crate cabling supports the configuration possibilities defined in the slot specification below.

If required, the basis configuration can be adapted to special applications by procurement and installation of additional internal cables.

Slot Specifications:



NU ~ NOT USED

(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

Notes to figure above:

1) When configurating the crate, power consumption should be calculated and checked against the available power supply outputs.

2) For a minimum configuration of the crate (excl. peripheral CTRLs) the following modules have to be procured: BT 1, 2 (Bus Termination Modules), CPU+SCM, CSA (System Control and Central Processor), MEM (RAM) and Power Supply.

3) For the remaining slot positions not specified in 2) above, the following rules apply:

Front Magazine: Slots 20, 21 for memory modules only.
Slots 15 - 18 can be used for DMA modules or memory modules, slot 18 is prepared for insertion of an EPM module connected to four EPAs (active extension of the data transfer bus), while cables for the other slots have to be procured separately.
Slots 5 - 10, 12, 14 can be used for peripheral CTRLs (single bus), but only slot 5 - 10 are equipped with cables for the adapter interface. If more peripheral CTRLs are required by the application additional cables have to be procured.
Slots 11 and 13 are available for insertion of more CPUs

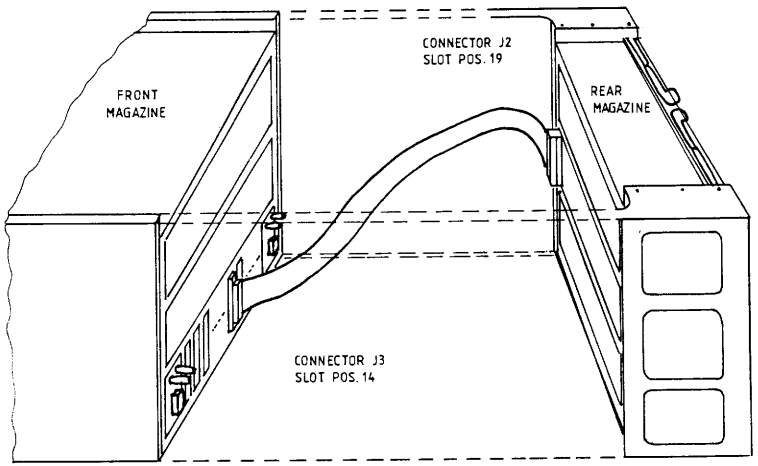
Rear Magazine: Slots 1, 8 - 14 APD can be used for special applications.

Slots 2, 7 are prepared for peripheral CTRL adapter modules.

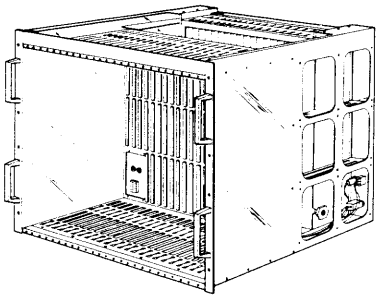
Slots 15 - 18 are prepared for extension adapter modules.

4) The CSA module has a port for interfacing a CR standard floppy station. If this part is to be used, a special cable is installed for the interface between the ST. FD CTRL front slot 14 and the CSA, as shown below.

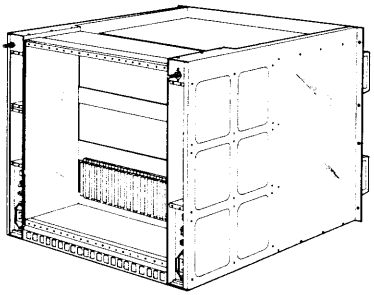
5) When a C-Bus module is required in the crate, the module is inserted in the DMA slot 18 (front magazine) and a special bus jumper is installed.



CU CRATE Dual Bus CR 8125M/425AB/00



Front View



Rear View

CHARACTERISTICS:

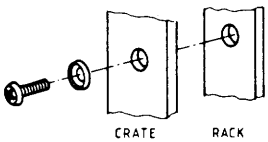
- For use in memory mapped or unmapped systems.
- Dual data transfer busses (Data Bus A and Data Bus B) redundant implemented as PCB.
- Two independent mains power input for redundant power supplies (A Bus and B Bus).
- Capable of housing up to 17 dual ported peripheral Controllers and 19 adapter modules.
- Rear crate can be tipped to allow access to internal crate cabling.
- Dimensions:

  H: 10 U ~ 445 mm
  W: 19" ~ 485 mm
  D: 620 mm
  Weight: 26 kg

- MTBF: 230,000 hours

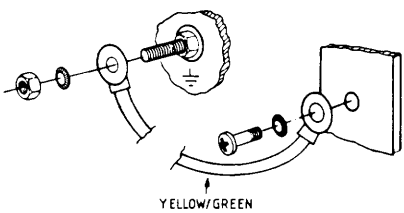
INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from front of rack and fixed by means of six screws, see below:



The crate must rest on supporting bars and a fan unit (CR8105M) must be located just below.

The protection GND terminal (chassis) should be connected to the rack by means of 6 mm^2 wires.

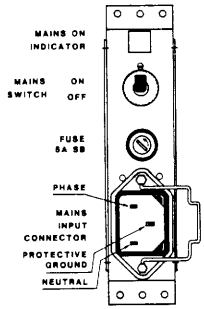


(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

The DC ground terminal is for connection to other units as required by the application.

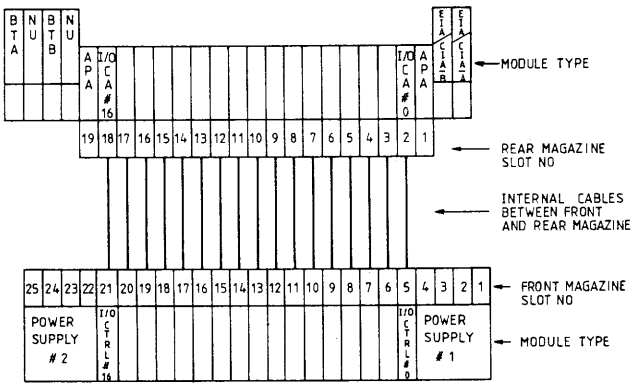
Two mains power inputs as defined below are available, one for each power supply. The crate mains inputs are connected to the power distribution panel by means of the power cable CR8201M/015--/00.



The inter-crate cabling supports the configuration possibilities defined in the slot specification below.

If required, the basis configuration can be adapted to special applications by procurement and installation of additional internal cables.

Slot Specifications:



(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

Notes to figure above:

1) When configurating the crate, power consumption should be calculated and checked against the available power supply outputs (only one supply due to redundancy).

2) For a minimum configuration (without peripheral CTRLs) of the crate, the following modules have to be procurred: BT A, BT B (Bus Termination Modules), Power Supply No. 1 and No. 2 and if mapped system CIA-A, CIA-B (Data Channel I/F) or if unmapped system EIA-A, EIA-B (Bus extension).

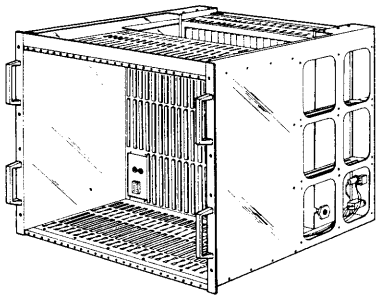
3) For the remaining slot positions, not specified in 2) above, the following racks apply:

   Front magazine: Slots 5 - 21 available for peripheral CTRLs.

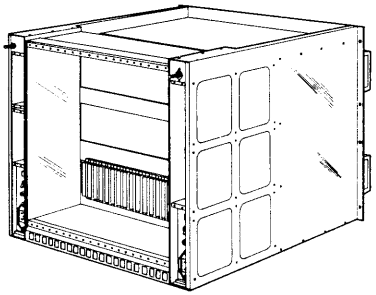
   Rear magazine: Slots 2 - 18 module locations for peripheral CTRLs adapters.
   Slots 1, 19 available for special application requirements.

4) When switching adapter modules are required (one spare peripheral CTRL for a group of active peripheral CTRLs) the crate cabling has to be modified as described in CR80 Handbook, Watchdog System.

CU CRATE Single Bus CR 8125M/325A-/00



Front View



Rear View

CHARACTERISTICS:

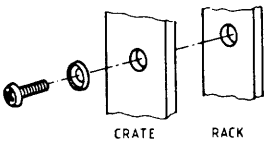
- For use in memory mapped or unmapped systems.
- Single data transfer bus (Data Bus A) implemented as PCB.
- Two mains input allows for two power supplies operating in parallel.
- Capable of housing up to 17 single ported I/O Controllers and 19 adapter modules.
- Rear magazine can be tipped to allow access to internal crate cabling.
- Dimensions:

H:	10 U	~	445 mm
W:	19"	~	485 mm
D:			620 mm
Weight:			26 kg

- MTBF: 350,000 hours

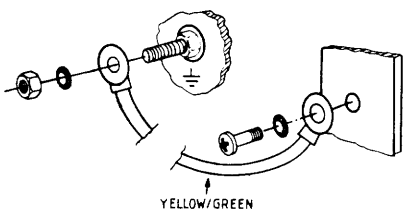
INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from the front of the rack and fixed by means of six screws, see below:



The crate must rest on supporting bars and a fan unit (CR8105M) must be located just below.

The protection GND terminal (chassis) should be connected to the rack by means of 6 mm^2 wires.

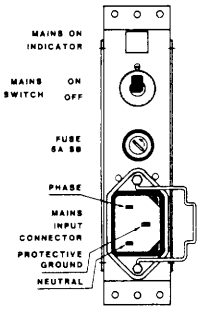


(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

The DC ground terminal is for connection to other units as required by the application.

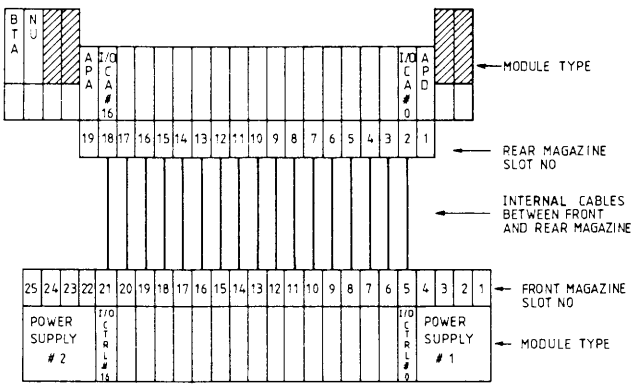
Two mains power inputs as defined below are available, one for each power supply. The crate mains inputs are connected to the power distribution panel by means of the power cable CR8201M/015--/00.



The inter-crate cabling supports the configuration possibilities defined in the slot specification below.

If required, the basis configuration can be adapted to special applications by procurement and installation of additional internal cables.

Slot Specifications:



NU ~ NOT USED

(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

Notes to figure above:

1) When configurating the crate, power consumption should be calculated and checked against the available power supply outputs.

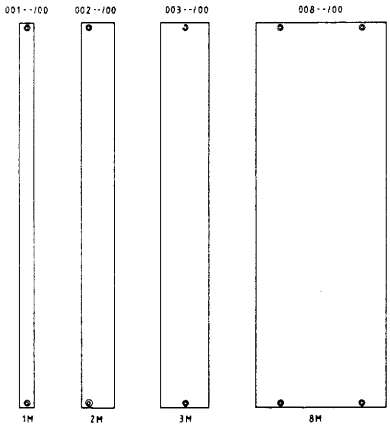
2) For a minimum configuration (without peripheral CTRLs) of the crate, the following modules have to be procured: BT (Bus Termination Module), Power Supply and if mapped system CIA-A (Data Channel I/F) or if unmapped system EIA-A (Bus extension I/F).

3) For the remaining slot positions, not specified in 2) above, the following rules apply:

Front Magazine:	Slots 5 - 21 available for peripheral CTRLs.
Rear Magazine:	Slots 2 - 18 module locations for peripheral CTRLs adapters. Slots 1 and 19 available for special applications.

DATASHEET FOR:

BLANK FRONT PANEL CR8147M/XXX--/00



MOUNTING SCREWS M4-10, csx (countersunk)

CHARACTERISTICS:

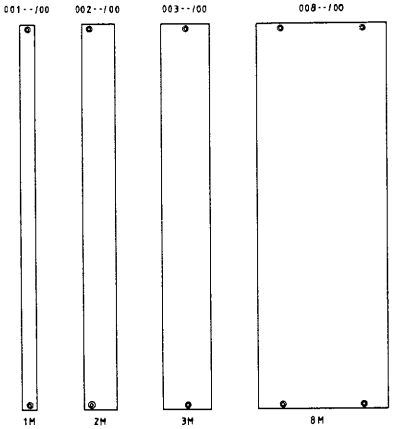
- Installed in front crate where no modules are present.
- Dimensions:
  H: 412 mm
  T: 3 mm
  W: as defined below

  CR8147M/001--/00: 17.1 mm 1 M
  CR8147M/002--/00: 2 M
  CR8147M/003--/00: 3 M
  CR8147M/008--/00: 8 M

- Painted in CR standard grey colour.

DATASHEET FOR:

BLANK REAR PANEL CR 8148M/XXX--/00



MOUNTING SCREWS M4 – 10, csx (countersunk)

CHARACTERISTICS:

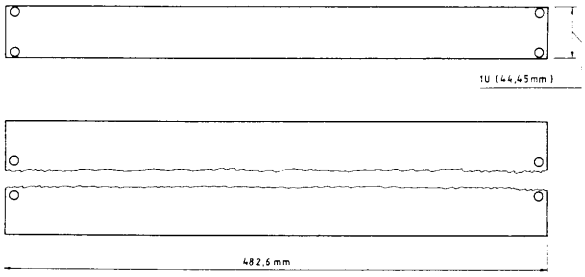
- Inserted in rear crate where no modules are present.

- Dimensions:
  H: 412 mm
  T: 3 mm
  W: as defined below:

  CR8148M/001--/00: 17.1 mm (1 M)
  CR8148M/002--/00: 2 M
  CR8148M/003--/00: 3 M
  CR8148M/008--/00: 8 M

DATASHEET FOR:

BLANK FRONT PANEL CR 8149M/XXX--/00



1U: CR 8149M/001--/00
2U: /002--/00
3U: /003--/00
4U: /004--/00
5U: /005--/00
6U: /006--/00

CHARACTERISTICS:

- Installed in front of rack where no units are present.

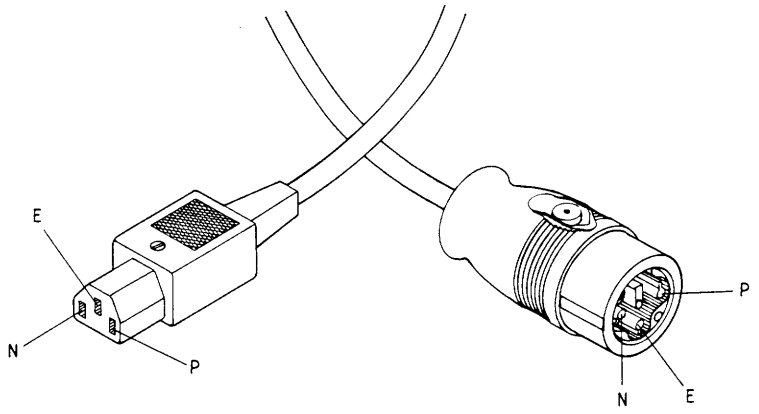
- Dimensions:
  W: 482.6 mm
  T: 4 mm
  H: as defined below

  CR8149M/001--/00: 1U ~ 44.5 mm
  CR8149M/002--/00: 2U ~ 88.9 mm
  CR8149M/003--/00: 3U ~ 133.0 mm
  CR8149M/004--/00: 4U ~ 179.0 mm
  CR8149M/005--/00: 5U ~ 222.0 mm
  CR8149M/006--/00: 6U ~ 267.0 mm

- Painted in CR standard grey colour.

DATASHEET FOR:

POWER CABLE CR8201M/015--/00

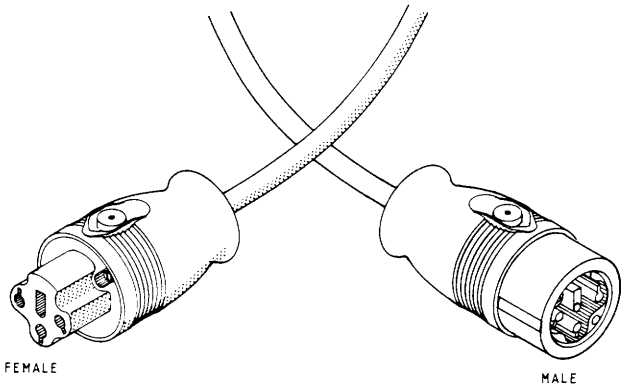


CHARACTERISTICS:

- Mains connection from power distribution plug bar to power supply in CR80M crate or CR8105M fan unit.
- 3 x 0.75mm^2 with screen.
- Length: 1500 mm
- Diameter: 8 mm

DATASHEET FOR:

POWER CABLE CR8201S/015--/00

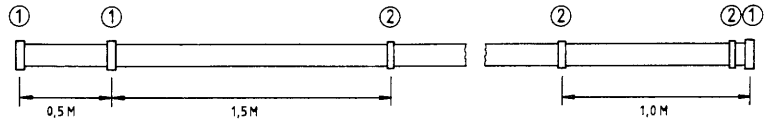


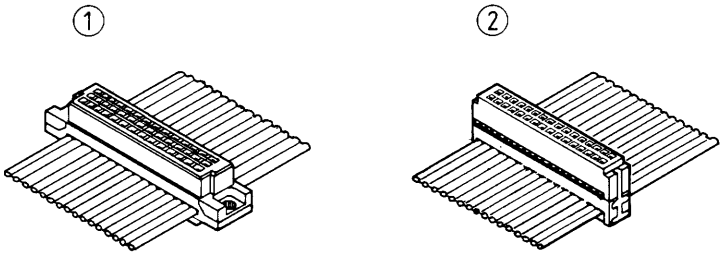
CHARACTERISTICS:

- Mains connection from power distribution plug bar to power supply in CR80S crate.
- 3 x 0.75mm^2 with screen.
- Length: 1500 mm
- Diameter: 8 mm

DATASHEET FOR:

CONFIGURATION BUS FLAT CABLE CR8209M/XXX--/00





CHARACTERISTICS:

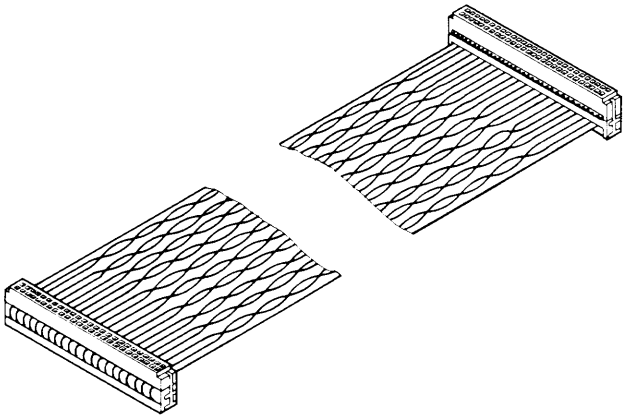
- 16 wired flat cable
- 2 types of female connectors:
  - ① For connection to Watchdog
  - ② For connection to Crate Controller Adaptor
- Available in 3 lengths:

	Number of type 2 connectors	Length
CR8209M/100--/00	9	10.5m
CR8209M/200--/00	19	20.5m
CR8209M/300--/00	29	30.5m

- The type 2 connectors are placed with a spacing of 1.0m

DATASHEET FOR:

DATA CHANNEL FLAT CABLE CR8211M/XXX - -/00



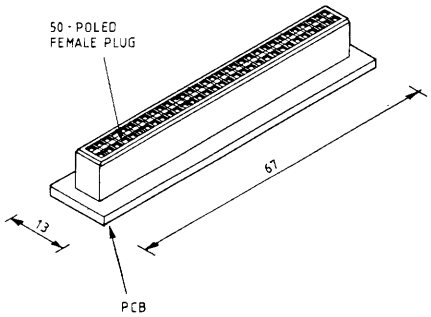
CHARACTERISTICS:

- Connection between Processor Unit (PU) and a number of Channel Units (CUs) in mapped systems.
- 50 wires arranged in 25 twisted pairs for differential transmission.
- 50 poled female connectors.
- Length: CR8211M/005--/00: 500 mm
  CR8211M/015--/00: 1500 mm
  CR8211M/030--/00: 3000 mm
  CR8211M/060--/00: 6000 mm
  CR8211M/120--/00: 12000 mm
- Width: 63 mm

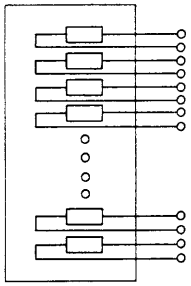
DATASHEET FOR:

DATA CHANNEL TERMINATION CR8211M/738--/00

PHYSICAL



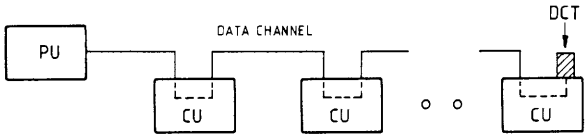
SCHEMATIC



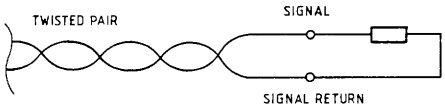
25 RESISTORS
(120 OHM)

CHARACTERISTICS:

- Terminates the CR80 Data Channel.
- Inserted instead of continuing flat cable.

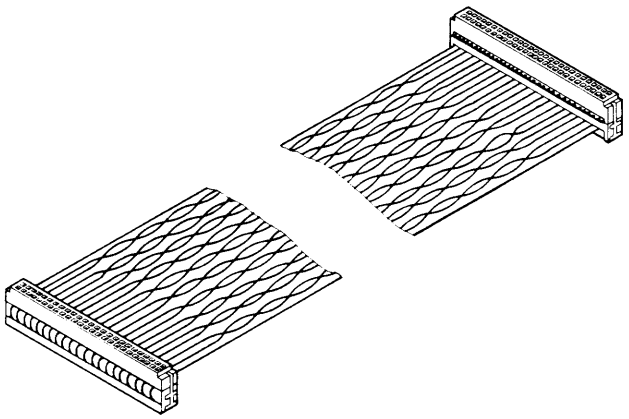


- Terminating resistor value: 120 ohm.
- Termination principle (one line):



DATASHEET FOR:

EXTENSION BUS FLAT CABLE CR8212M/XXX--/00

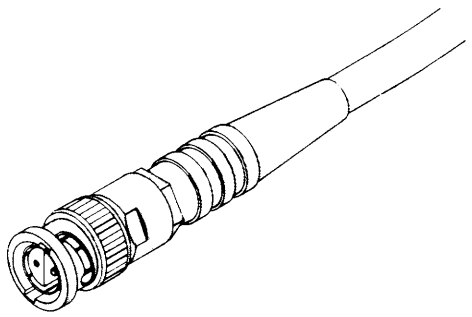


CHARACTERISTICS:

- Connection between Processor Unit (PU) and Channel Unit (CU) in unmapped systems.
- 60 wires arranged in 30 twisted pairs for differential transmission.
- 60 poled female connectors.
- Length: CR8212M/015—/00: 1500 mm
  CR8212M/030—/00: 3000 mm
- Width: 75 mm

DATASHEET FOR:

SUPRABUS CABLE CR8215M/XXX--/00

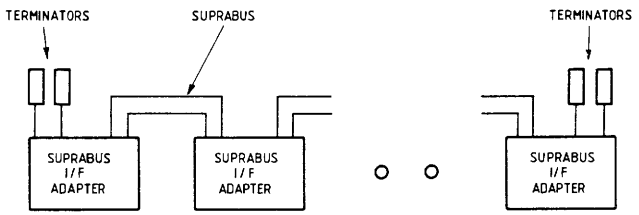


CHARACTERISTICS:

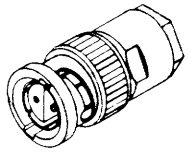
- Twisted pair with screen.
- Length:

CR8215M/005--/00:	0.5m
CR8215M/015--/00:	1.5m
CR8215M/030--/00:	3.0m
CR8215M/060--/00:	6.0m
CR8215M/120--/00:	12.0m

- Connector: BNO-plug
- The Supra Bus consists of two parallel cables
- Configuration:

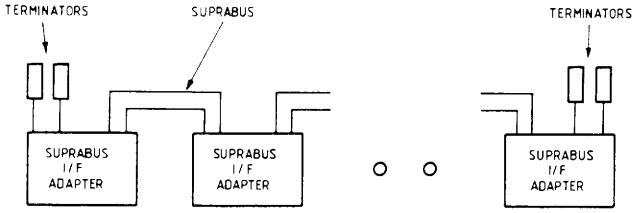


SUPRABUS TERMINATOR CR8215M/778--/00



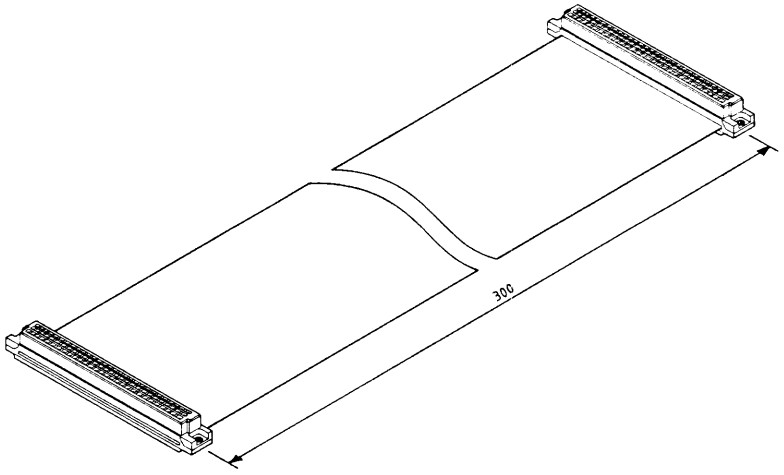
CHARACTERISTICS:

- 82 ohm termination of Supra Bus.
- Replaces continuing Supra Bus.
- Two needed in each end of the Supra Bus.
- Configuration:



DATASHEET FOR:

INTERNAL FLAT CABLE CR8216M/000- -/00 (1 ADAPTER)

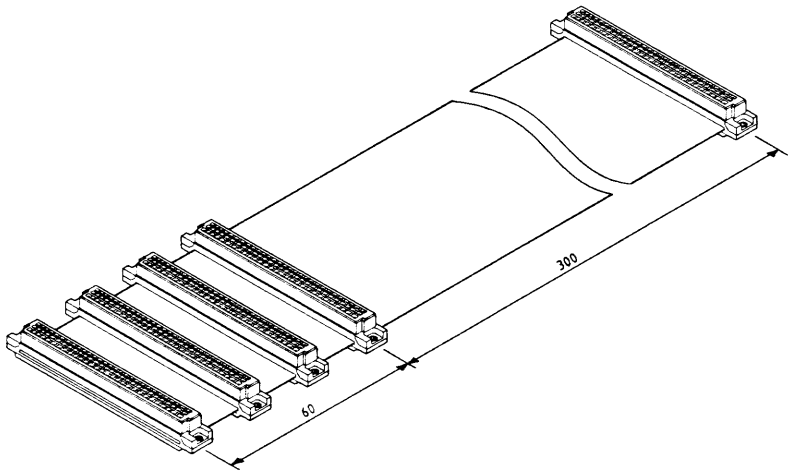


CHARACTERISTICS:

- Connects 1 adapter to a front module.
- 64 wires.
- 64 poled female connectors.
- Length: 300 mm
- Width: 80 mm

DATASHEET FOR:

INTERNAL FLAT CABLE CR8217M/004--/00 (4 ADAPTERS)

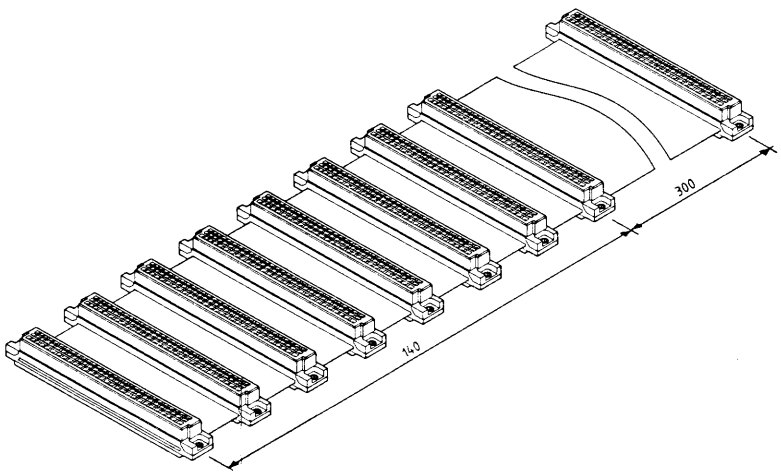


CHARACTERISTICS:

- Connects 4 adapters to a front module.
- 64 wires.
- 64 poled female connectors.
- Length: 360 mm
- Width: 80 mm

DATASHEET FOR:

INTERNAL FLAT CABLE CR8217M/008--/00 (8 ADAPTERS)

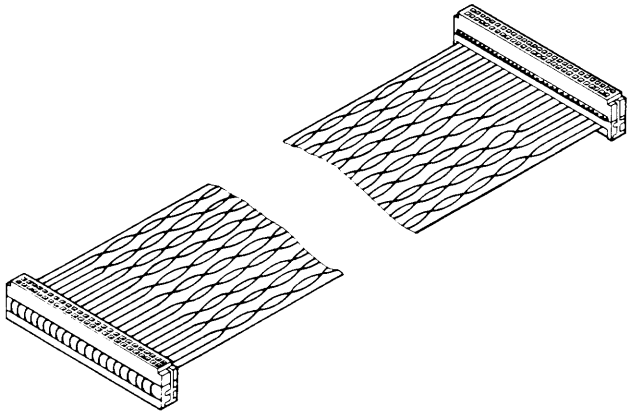


CHARACTERISTICS:

- Connects 8 adapters to a front module.
- 64 wires.
- 64 poled female connectors.
- Length: 440 mm
- Width: 80 mm

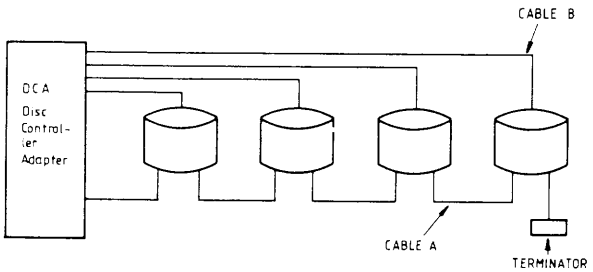
DATASHEET FOR:

DISC DRIVE CABLE A CR8221M/010--/00



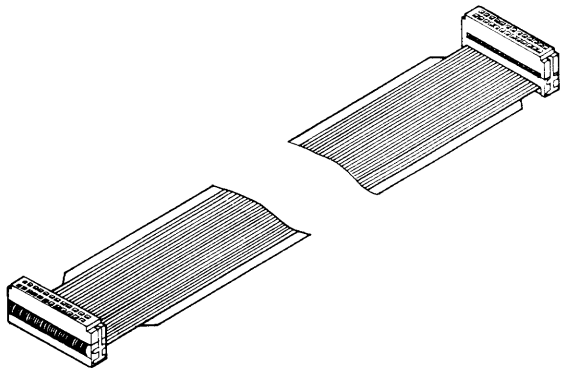
CHARACTERISTICS:

- Flat cable with 30 twisted pairs.
- Length: 7.5 m (25ft)
- Connector: 60 poled female plug.
- Configuration:



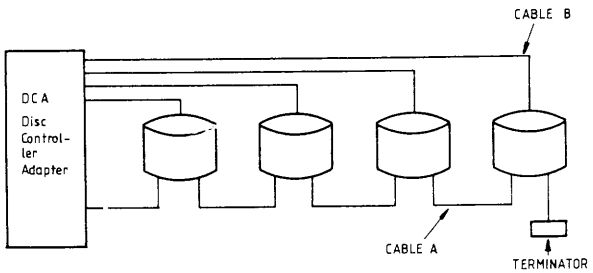
DATASHEET FOR:

DISC DRIVE CABLE B CR8221M/020--/00



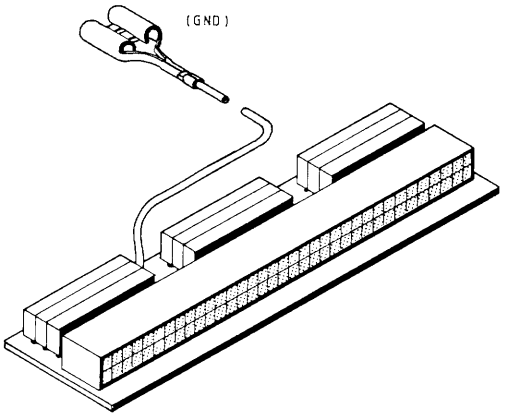
CHARACTERISTICS:

- 26 wires with ground plane.
- Length 7.5 m (25 ft).
- Connector: 26 poled female plug.
- Configuration:

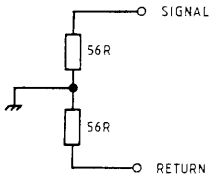


DATASHEET FOR:

DISC DRIVE CABLE A TERMINATOR CR8221M/758--/00

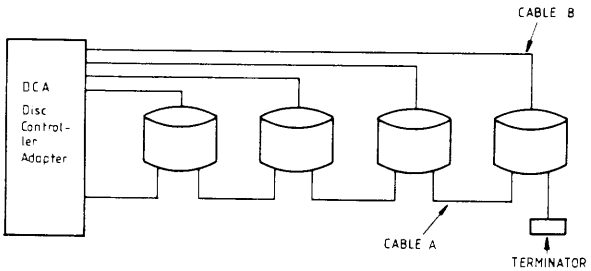


TERMINATION OF EACH TWISTED PAIR:



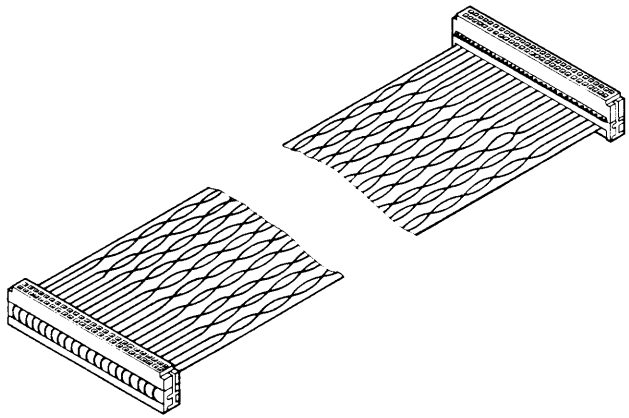
CHARACTERISTICS:

- Terminates Cable A in last Disc Drive.
- Connector: 60 poled female plug.



DATASHEET FOR:

FLOPPY DISK FLAT CABLE CR8222M/XXX--/00

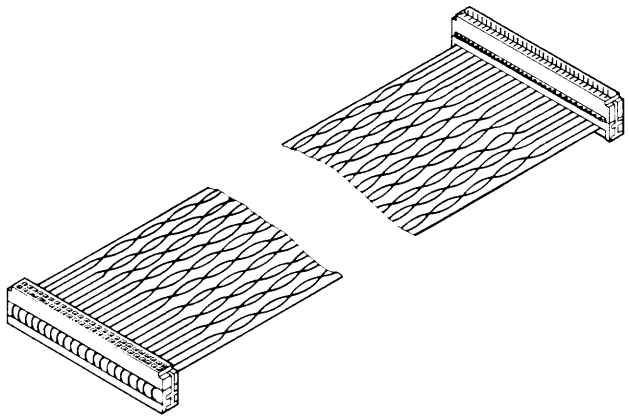


CHARACTERISTICS:

- Connection between Standard Floppy Disk Adapter (SFA) or CSA and floppy disk drive.
- 50 wires arranged in 25 twisted pairs for differential transmission.
- 50 poled female connectors.
- Length: CR8222M/010--/00: 1000mm
  CR8222M/030--/00: 3000mm
- Width: 67 mm

DATASHEET FOR:

MAGTAPE FLAT CABLE CR8223M/050--/00



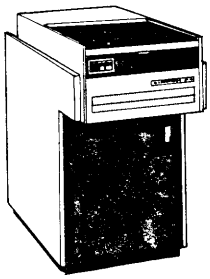
CHARACTERISTICS:

- Connection between Tape station and Tape Controller Adaptor (TCA).
- 50 wires arranged in 25 twisted pairs for differential transmission.
- 50 poled female connectors in one end and 50 poled edge connector in the other.
- Length: 5000mm
- Width: 63mm

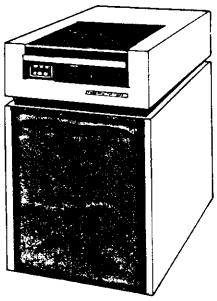
CR80

DATASHEET FOR:

SMD DISK DRIVE CR 8300-/XXX--/00



PEDESTAL CABINET
CR 8300-/040--/00
CR 8300-/080--/00



ACOUSTIC CABINET
CR 8300-/150--/00
CR 8300-/300--/00

CHARACTERISTICS:

- Disk Drive with removable Disk Pack.
- Average access time: 38 msec.
- Data transfer rate: 1.2 MB per sec.
- Interface to the Disk CTRL CR8044 through the Disk CTRL Adapter CR8084.
- Daisy chain possible.
- Power: 220V 50Hz single phase (See Programmer's Reference for options).

  Voltage tolerance max +7% -11%
  Frequency tolerance max +1% -2%
- Operation environment:
  Temperature: 15°C - 32°C
  Change of temperature max 6°C/h
  Relative humidity: 20% to 80%
  non condensing.
- MTBF: 4000 h

PROGRAMMERS REFERENCE:

	CAPACITY	CABINET
CR8300-/040--/00	40MB	PEDESTAL *
CR8300-/080--/00	80MB	PEDESTAL *
CR8033-/150--/00	150MB	ACOUSTIC
CR8300-/300--/00	300MB	ACOUSTIC
* CR8300-/001--/00	ACOUSTIC CABINET OPTION	
CR8300-/002--/00	DUAL CHANNEL OPTION	

CAPACITY**	33.669MB	67.420MB	127.942MB	256.196MB
UNFORMATTED	40MB	80MB	150MB	300MB
POWER	900W	900W	1600W	1600W
WIDTH	48.3 cm	48.3 cm	58.4 cm	58.4 cm
DEPTH	86.4 cm	86.4 cm	91.4 cm	91.4 cm
HEIGHT	86.4 cm	86.4 cm	92.0 cm	92.0 cm
WEIGHT	110 kg	110 kg	252 kg	252 kg
DISK PACK	CR8319/ 040	CR8319/ 080	CR8319/ 150	CR8319/ 150
POWER OPTION	120V 60Hz	120V 60Hz	220V 60Hz	220V 60Hz

SECTOR COUNT SWITCH SETTINGS:

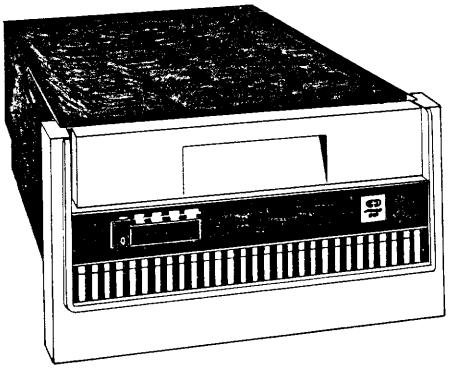
SWITCH NO.	0	1	2	3	4	5	6	7	8	9	10	11	0 = ON
	1	0	1	1	1	0	1	0	0	1	1	1	1 = OFF

** FORMATTED WITH 512 BYTES PER SECTOR
32 SECTORS PER TRACK

CR80

DATASHEET FOR:

CMD DISK DRIVE CR8301-/XXX--/00



CHARACTERISTICS:

- Disk drive with fixed disk and with 16MB removable disk pack.
- Average access time: 38 msec.
- Data transfer rate: 1.2 MB per sec.
- Interface to the Disk CTRL CR8044 through the Disk CTRL Adapter CR8084.
- Daisy chain possible.
- Power: 220V 50Hz single phase
  120V 50Hz or 60Hz optional.
  950W
  Voltage tolerance max +7% -13%
  Frequency tolerance max +1% -2%
- Operation environment:
  Temperature 10°C - 35°C
  Temperature Gradient 10°C/hour
  Relative humidity 10% to 90%
  non condensing
- MTBF: 8000 h

PROGRAMMERS REFERENCE:

CAPACITY: FIXED + REMOVABLE PART

	UNFORMATTED	FORMATTED *
CR8301-/016--/00	16MB + 16MB	13.484MB + 13.484MB
CR8301-/048--/00	48MB + 16MB	40.452MB + 13.484MB
CR8301-/080--/00	80MB + 16MB	67.420MB + 13.484MB

* FORMATTED WITH 512 BYTES PER SECTOR
32 SECTORS PER TRACK
Removable disk pack 16MB: CR8319/016

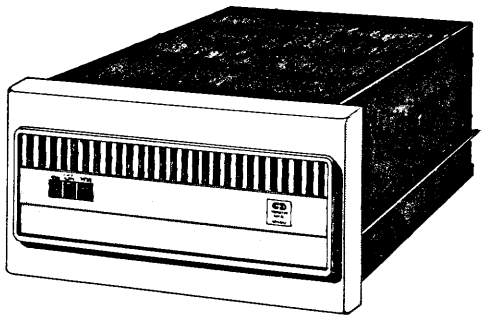
SECTOR COUNT SWITCH SETTINGS:

SWITCH NO	1	2	3	4	5	6	7	8	
	0	0	0	0	0	1	0	0	0 = ON, 1 = OFF

- Dimensions
  Width: max. 48.3 cm
  Depth: 80.6 cm
  Height: 26.4 cm
  Weigth: 77.1 kg
  For rack mount
  Pedestal cabinet optional

DATASHEET FOR:

MMD DISK DRIVE CR8302-/XXX--/00



CHARACTERISTICS:

- Disk drive with fixed disk. Fixed head option.
- Average access time:
  12MB/24MB 48 msec.
  80MB 38 msec.
- Data transfer rate: 1.2MB per second.
- Interface to the Disk CTRL CR8044 through the Disk CTRL Adapter CR8084.
- Daisy chain possible.
- Power 220V 50Hz single phase
  120V 60Hz optional
  400W
  Voltage tolerance +6% -11%
  Frequency tolerance +1% -2%
- Operation environment:
  Temperature 10°C - 40°C
  Temperature gradient 10°C/hour
  Relative humidity 20% - 80%
  non condensing.
- MTBF: 4000 h

PROGRAMMERS REFERENCE:

CAPACITY

	UNFORMATTED	FORMATTED *
CR8302-/012--/00	12MB	10.485MB
* CR8302-/024--/00	24MB	20.971MB
**CR8302-/080--/00	80MB	67.420MB

OPTIONS:

* CR8302-/001--/00	1MB FIXED HEADS	0.786432MB
**CR8302-/002--/00	1MB FIXED HEADS	0.786432MB
**CR8302-/003--/00	2MB FIXED HEADS	1.572864MB

* FORMATTED WITH 512 BYTES PER SECTOR
  32 SECTORS PER TRACK

SECTOR COUNT SWITCH SETTINGS

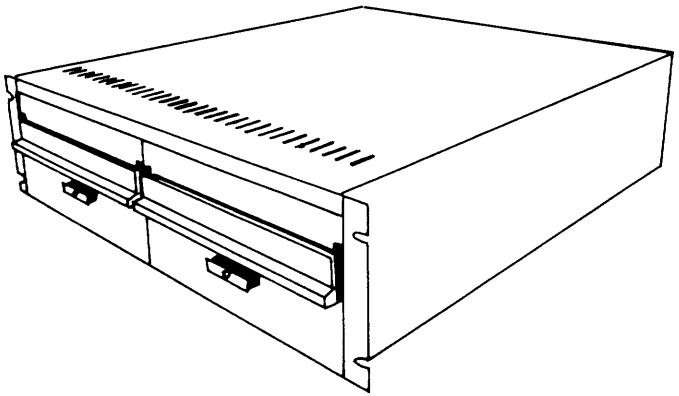
SWITCH NO.	0	1	2	3	4	5	6	7	8	9	10	11
	1	0	1	1	1	0	1	0	0	1	1	1

0 = ON
1 = OFF

- Dimensions:
  Width: 42.55 cm
  Depth: 67.20 cm
  Height: 25.91 cm
  Weight: 38.60 kg
  For rack mount

DATASHEET FOR:

FLOPPY DISK STATION CR8308-/XXX--/00



CHARACTERISTICS:

- Floppy disk station with 1 or 2 single or dual side drives, incl. 19" rack mountable enclosure and power supply.
- Data transfer rate: 250 kilobits/sec.
- Interface to the ST. FLOPPY CTRL CR8047 through the SFA CR8087 or the CSA CR8070.
- Power:
  220V optional 115V
  50Hz optional 60Hz

  Voltage tolerance +10% -6%
  Frequency tolerance +/- 0.5Hz
  110W/drive
- Dimensions:
  Height: 13.34 cm (3 U)
  Depth: 50.80 cm (without fuses and power cord)
  Weight: 20 kg
- MTBF: 8000h

PROGRAMMERS REFERENCE:

	DRIVES	SIDES	CAPACITY *
CR8308-/108--/00	single	single	250K BYTES
CR8308-/116--/00	single	dual	500K BYTES
CR8308-/216--/00	dual	single	500K BYTES
CR8308-/232--/00	dual	dual	1000K BYTES

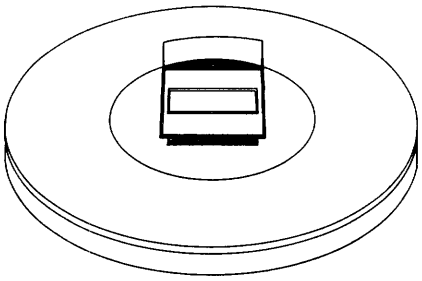
* With IBM 3740 formatted diskettes.
77 tracks with 26 sectors with 128 bytes/side.

Average access time:

single side	dual side
351 msec.	174 msec.

DATASHEET FOR:

DISC CARTRIDGE for CMD DISC DRIVE CR8319-/016--/00



CHARACTERISTICS:

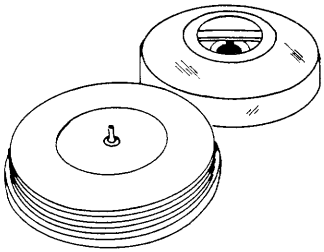
- Disk Cartridge for CMD Disk Drive.
- Bit Density 6000BPI.
- Disk Diameter 35.5 cm.
- Disk Thickness .19 cm.
- Rotational Speed 3600 r/min.

PROGRAMMERS REFERENCE:

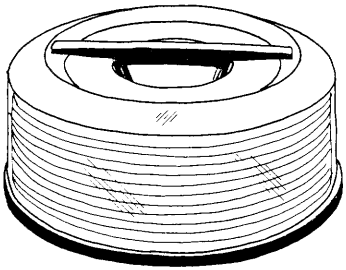
	CR8319-/016--/00
CAPACITY	
UNFORMATTED	16M Bytes
FORMATTED	13.484M Bytes
DATA SURFACES	1
SERVO SURFACES	1
TPI	384

DATASHEET FOR:

DISC PACKS for SMD DRIVES CR 8319-/XXX--/00



CR 8319-/040--/00
CR 8319-/080--/00



CR 8319-/150--/00

CHARACTERISTICS:

- Disk Packs for SMD Disk Drives.
- Bit Density 6000BPI.
- Disk Diameter 35.5 cm.
- Disk Thickness .19 cm.
- Rotational Speed 3600 r/min.

PROGRAMMERS REFERENCE:

	CAPACITY	
	UNFORMATTED	FORMATTED*
CR8319-/040--/00	40M Bytes	33.669M Bytes
CR8319-/080--/00	80M Bytes	67.420M Bytes
CR8319-/150--/00+	150/300M Bytes	127.196/256.196M Bytes

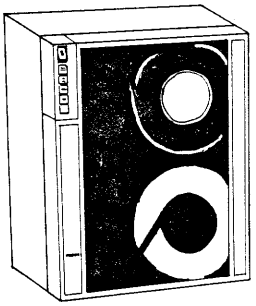
* Formatted with 512 bytes per sector and 32 sectors per track.

\+ The disk pack can be used in a 150 MB drive or a 300 MB drive.

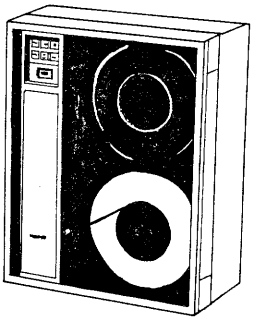
Capacity	40MB	80MB	150/300MB
Data Surfaces	5	5	19
Servo Surfaces	1	1	1
TPI	192	384	192/384

DATASHEET FOR:

TAPE STATION CR 8320-/XXX--/00



CR 8320-/002--/00



CR 8320-/001--/00

CHARACTERISTICS:

- Magnetic tape station using reels with a maximum diameter of 26.7 cm (10.5 inches) containing 731.5 cm (2400 feet) of tape. 9 tracks. ANSI and IBM compatible.
- Read after write check.
- NRZI or Phase Encoded (PE) tape format is selectable on the tape transport. (For CR8320-/001—/00 as option only).
- Interface to the Tape CTRL CR8045 through the Tape CTRL Adapter CR8085.
- Daisy chain possible.
- Power:
  220V 110V optional
  50Hz +/- 2 60Hz +/- 2 optional
- MTBF:
  CR8320-/001—/00: 8000 h
  CR8320-/002—/00: 2500 h

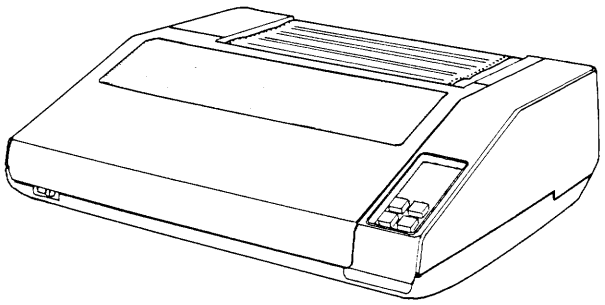
PROGRAMMERS REFERENCE:

	CR8320-/001—/00	CR8320-/002--/00	
Recording type	PE	PE	NRZI
** Capacity	~30Mbytes	~30Mbytes	~ 18Mbytes
Data density	1600 bpi	1600 bpi	800 bpi
Transport speed	Δ 25 ips	75 ips	
+ Forward move time	1155 sec.	385 sec.	
+ Rewind time	144 sec.	115 sec.	
Transfer rate	40Kbytes/ sec.	120Kbytes/ sec.	60Kbytes/ sec.
Dimensions:			
Width	48.26 cm	48.26 cm	
Depth	40.64 cm	50.80 cm	
Heigth	60.96 cm	61.00 cm	
Weight	38.60 kg	52.20 kg	
Environment:			
Temperature	2^{o}C-50^{o}C	5^{o}C - 43C	
** Relative humidity	15% - 95%	30% - 80%	
Altitude	0 - 6096 m	0 - 1219 m	
Parity and CR check	yes	yes	yes
Single error correction	yes	yes	no
POWER	360	900W	

\+ With a 731.5 m (2400 feet) tape
* Formatted with records of 2048 bytes
** Non condensing
++ 1219 - 2132 m as option
Δ Optional 45 ips

DATASHEET FOR:

MATRIX PRINTER CR8330-/200--/00



CHARACTERISTICS:

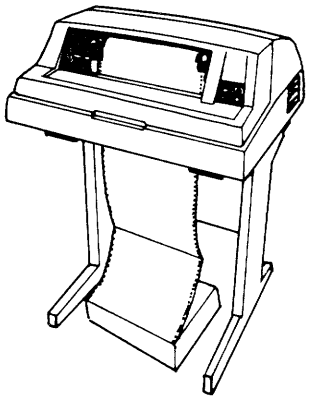
- Matrix printer.
- 132 print coloumns.
- Interface to the Dual Par. CTRL CR8046 through the Printer CTRL Adapter CR8086. Interface to the CPU-SCM CR8002 through the CSA CR8070.
  Max. cable length 15 m.
- Power:
  220V 120V optional
  50Hz 60Hz optional
  250W
- Dimensions:
  Height: 20.3 cm
  Width: 67.2 cm
  Depth: 59.5 cm
  Weight: 27.1 kg
  Delivered with pedestal
- MTBF: 4,000 hours

PROGRAMMERS REFERENCE:

Print speed:	340 cps bi-directional 200 lpm at 72 characters/line 125 lpm at 132 characters/line
Spacing:	Vertical 6 lines/inch Horizontal 10, 16.7 characters per 2.54 cm optional
Forms:	Edge-perforated 7.62 cm (3 inches) to 40.64 cm (16 inches) Forms length selects switch
Copies:	Original plus five copies
Character set:	128 characters ASCII

DATASHEET FOR:

BAND PRINTER CR8331-/XXX--/00



CHARACTERISTICS:

- Band printer with selftest.
- 132 print coloumns.
- Interface to the Dual Par. CTRL CR8046 through the Printer CTRL Adapter CR8086. Interface to the CPU + SCM CR8002 through the CSA CR8070.
  Max. cable length 15 m.
- Power:
  220V 110V optional
  50Hz 60Hz optional
- Dimensions:
  Height: 38.0 cm
  Width: 77.0 cm
  Depth: 64.0 cm
  Weight: 69.0 kg
  (With pedestal 82 kg)
  Delivered with pedestal.
- MTBF: 2,500 hours

PROGRAMMERS REFERENCE:

	CR8331-/300--/00	CR8331-/600--/00
Print speed *	300 lpm	600 lpm
Power	250W	275W

* Print speed at 132 characters/line with 64 characters ASCII character set. Other character sets are optional.

Spacing: Vertical 6/8 lines/inch
Horizontal 10 cpi

Forms: Edge-perforated 3 to 16 inches
Form length selects switch

Copies: Original plus five copies

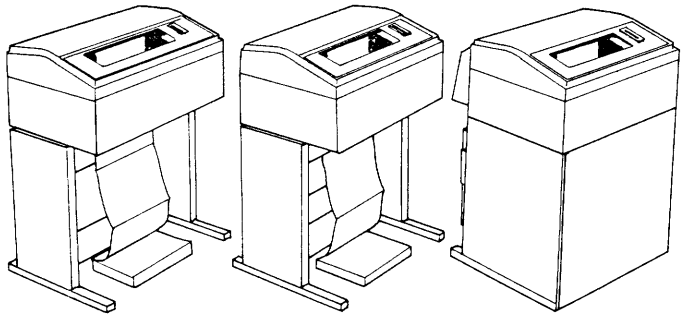
Internal switch settings with skipover = 3.

S1		
1	OFF	
2		ON
3		ON
4	OFF	
5	OFF	
6	OFF	
7		ON
8	OFF	

S2		
1	OFF	
2	OFF	} skipover
3	OFF	
4		ON
5	OFF	
6	OFF	
7	OFF	
8	OFF	

DATASHEET FOR:

DRUM PRINTER CR8332-/XXX--/00



CR8332-/300--/00
300 lpm

CR8332-/600--/00
600 lpm

CR8332-/900--/00
900 lpm

CHARACTERISTICS:

- Drum Printer.
- 136 print coloumns OCR-B font.
- Interface to the Dual Par. CTRL CR8046 through the Printer CTRL Adapter CR8086. Interface to the CPU + SCM CR8002 through the CSA CR8070. Max. cable length 15 m.
- Power:
  220V 115V optional
  50Hz 60Hz optional
- Dimensions inclusive pedestal
  Height: 115 cm
  Width: 85 cm
  Depth: 67 cm
- MTBF: 2,500 hours

PROGRAMMERS REFERENCE:

Print speed *	300 lpm	600 lpm	900 lpm
Power	525W	680W	825W
Weight	153 kg	167 kg	189 kg

* Print speed at 136 characters/line with 64 characters ASCII character set. Other character sets are optional.

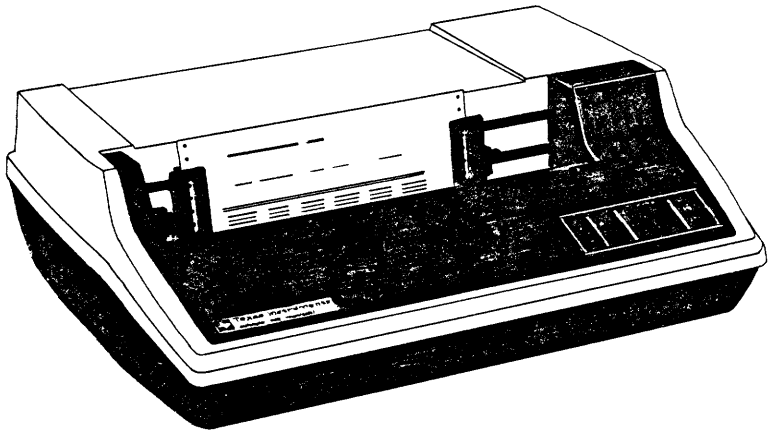
Spacing: Vertical 6/8 lines/inch
Horizontal 10 cpi

Forms: Edge-perfforated 4 to 16.75 inches.
Form length selects switch.

Copies: Original plus five copies.

DATASHEET FOR:

MATRIX PRINTER CR8333-/064--/00



CHARACTERISTICS:

- Matrix printer 64-440 LPM.
- Asynchronous interface EIA RS-232-C.
- Power:

220V	47 - 63Hz
120V	47 - 63Hz

- Operational environment:

Temperature	5°C - 40°C
Humidity	5% - 90%
non condensing	

- Dimensions:

Height:	20.3 cm
Width:	65.4 cm
Depth:	50.8 cm
Weight:	25.0 kg

- MTBF: 3,000 hours

PROGRAMMERS REFERENCE:

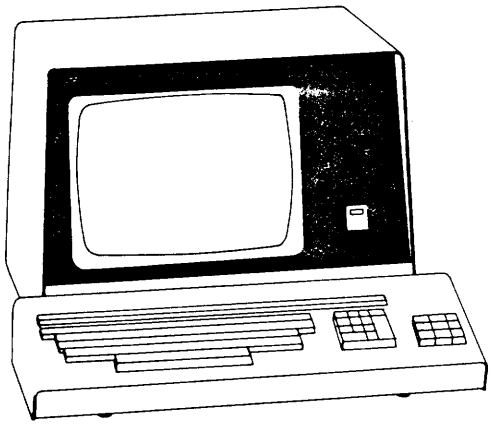
64 ASCII characters in 9 x 7 dot pattern.
Print speed 150 cps.
64 lines per minute (132 characters per line)
440 lines per minute (10 characters per line)
Bi-directional printing.

Line spacing:	6 or 8 vertical lines per 2.54 cm
Line length:	up to 132 characters at 10 characters per 2.54 cm
Forms:	Edge-perforated 7.62 cm (3 in.) to 38.1 cm (15 in.)
Multi-copy forms:	Up to 6 parts.
Baud rates:	110 - 9600 switch selectable.
Parity:	Checking for odd, even or none.
Line Buffer:	256 characters.

CR80

DATASHEET FOR:

DISPLAY TERMINAL CR8350-/XXX--/00



CHARACTERISTICS:

- Microprocessor-based alphanumeric display terminal with separate keyboard.
- Asynchronous transmission V24 (or current loop) 9600 bps.
- Power:
  220V (110V optional)
  50Hz (60Hz optional)
  120W
- Operating environment:
  Temperature: 5°C - 40C
  Relative humidity: 5% - 80%
  non condensing
- MTBF: 2,850 hours

PROGRAMMERS REFERENCE:

	CR8350-/004--/00	CR8350-/005--/00
Number of function keys	8	24

Lines:	25
Characters per line:	80
Character set:	128 character ASCII upper/lower case
Screen size:	30.5 cm (12 inches) diagonal
Character format:	9 x 9 dot matrix
Colour:	green phosphor P31
Keyboard:	cursor control cluster numeric cluster

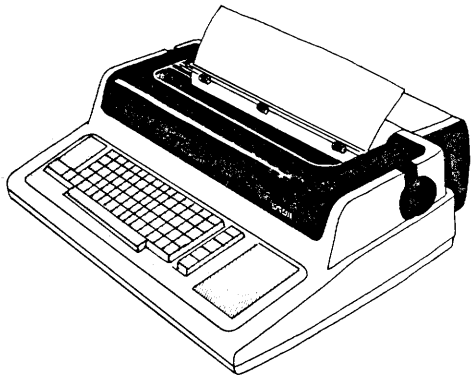
Dimentions:	In cm:	Width:	Depth:	Height:
	Display	43.2	40.6	33.0
	Kayboard	50.5	20.3	7.6

Weight:	Keyboard:	4.5 kg
	Display:	15.8 kg

CR80

DATASHEET FOR:

MATRIX PRINTER W. KEYBOARD CR 8390-/034--/00



CHARACTERISTICS:

- Terminal printer operating like typewriter.
- Asynchronous transmission V24, 300 bps. max. cable length 50 m.
- Power:
  220V 50Hz
  150V 60Hz
  optional
- Dimensions:
  Height: 18 cm
  Width: 57 cm
  Depth: 40 cm
  Weight: 11.2 kg
- MTBF: 3,300 hours

PROGRAMMERS REFERENCE:

Print speed:	30 cps, 132 print columns, 9 x 7 matrix impact printing
Spacing:	Horizontal 10 - 16.5 characters/inch Vertical 2 - 12 lines/inch
Character set:	128 character ASCII upper/lower case 96 printable characters
Baud rate:	110 or 300 baud switch-selectable.
Switch selectable parity:	Even, odd, mark or space.

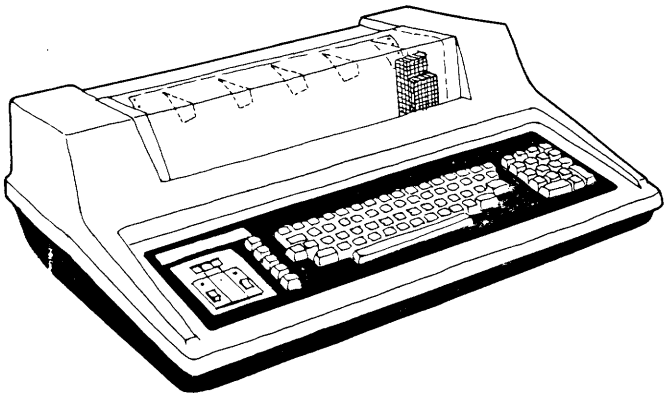
Self-test program.
Automatic printout of status message.

Tractor feed optional.

CR80

DATASHEET FOR:

MATRIX PRINTER WITH KEYBOARD CR 8391-/820--/00



CHARACTERISTICS:

- Microprocessor based Matrix printer with keyboard.
- Asynchronous transmission EIA RS-232-C compatible.
- Baud rate: 110 - 9600 baud.
- Acoustic Noise: Less than 60 dB (A weighted) measured 0.9 m directly in front under free field conditions, while printing 150 characters per second.
- Power:
  220V 47 - 63Hz
  110V 47 - 63Hz
  50 VA idle
  75 VA printing
  150 VA form feed
- Operational environment:
  Temperature: 5°C - 40°C
  Relative humidity: 5% - 90%
  non condensing.
- Dimensions:

Height:	20.95 cm
Width:	66.04 cm
Depth:	53.34 cm
Weight:	18.18 kg

- MTBF: 3300 hours

PROGRAMMERS REFERENCE:

9 x 7 matrix character font.

Print speed: 150 cps.

Characters per line: 132 max.

Character spacing: 10 per 2.54 cm

Line spacing: 6 per 2.54 cm

Forms: Edge-perforated 7.62 cm (3 in) to 37.78 cm (14.875 in)

Form length: 27.94 cm (11 in)

Copies: Original + 5 copies.

95 printable ASCII characters.

Full ASCII alphanumeric keyboard ANSI compatible typewriter layout.

Indicator for terminal status and line status. Controls for Line/Local. Last character visibility.

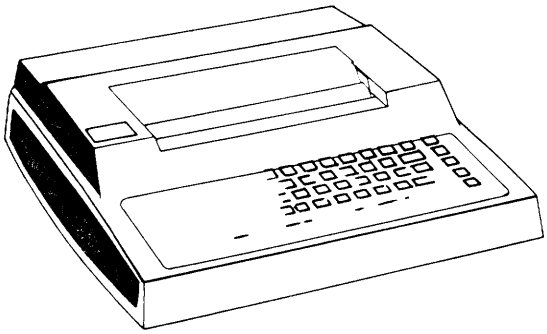
Full or half duplex operation. Half duplex with or without reverse channel.

Odd, Even, Mark, or Space parity. Buffered receiver.

CR80

DATASHEET FOR:

THERMAL PRINTER W. KEYBOARD CR 8392-/743--/00



CHARACTERISTICS:

- Terminal printer with keyboard.
- Asynchronous transmission EIA RS-232-C or DC-current loop serial.
- Power:
  220V
  110V

  Frequency 47Hz through 63Hz 75 W maximum.
- Operation environments:
  Temperature: 10°C - 40°C
  Humidity: 10% to 90%
  non condensing.
- Dimensions:

Height:	10.8 cm
Width:	37.1 cm
Depth:	38.7 cm
Weight:	5.08 kg

- MTBF: 4,000 hours

PROGRAMMERS REFERENCE:

Print speed 10 or 30 characters per second.

5 x 7 dot matrix printhead with print contrast control.

Standard typewriter - like ASCII keyboard with 69 printable characters. Option: full ASCII with 95 printable characters.

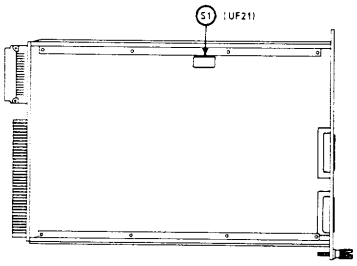
Full or half duplex operation.
110 or 300 baud.

Line length is 20.32 cm.
80 characters per line.

Even parity. Optional odd or mark.

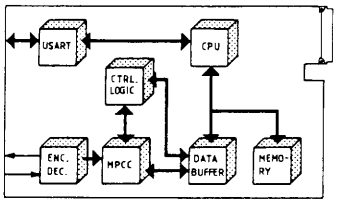
TDX CONTROLLER CR1070S/000--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- The central device of the TDX bus system
- Data on the uni directional upper bus is routed through the TDX controller to the lower bus
- Frames transmitted on the lower bus are extended with one byte which is "Mux no" (the next device address allowed to transmit on the upper bus)
- TDX signals for one or two TDX busses are connected to a 25 pin Cannon connector on the front panel
- Control lines, connected to a 25 pin Cannon connector on the front panel, are used to monitor and control the TDX controller by a watchdog
- Possibility for external clock
- Power Consumption:

+5V:	2A
+12V:	0.1A
-12V:	0.1A

- MTBF: 30,500 hours

(CONTINUED)

PROGRAMMERS REFERENCE:

Internal I/O Address:

	I/O Base Address
CT C	C0 H
USART	C8 H
STATUS REG1	D0 H
STATUS REG 2	D8 H
Counter 8253	E8 H
Command Reg	E0 H
Mux Reg	F0 H

Four sockets for memory:

	Mem. Base Address
UA20	0 H
UA17	4000 H
UC20	5000 H
UC17	6000 H

FIF0 2	F000 H (RD)
FIF0 4	F000 H (WR)

UF21 4 bit DIL switch can be read by the CPU for S/W mode selection

1	2	3	4	
D4	D5	D6	D7	"0" CLOSED
				"1" OPEN

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Strap Overview

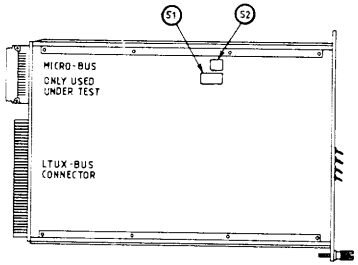
		A	B	C
SR2	PROM A20 PIN21	A11	VCC	WR
SR3	PROM/RAM A17 PIN21	A11	VCC	WR
SR4	PROM/RAM A17 PIN18	CS1	CS1	-
SR5	PROM/RAM C20 PIN23	A11	VCC	WR
SR6	PROM/RAM C20 PIN26	CS2	VCC	-
SR7	PROM/RAM C20 PIN20	CS2	CS2	-
SR8	PROM/RAM C17 PIN23	A11	VCC	WR
SR9	PROM/RAM C17 PIN26	CS2	VCC	-
SR10	PROM/RAM C17 PIN20	CS2	CS2	-
SR11	V24 R*c	internal clk.	external clk.	-
SR12	V24 T*c	internal clk.	external clk.	-
SR13	WAIT	disconnect	connect	-
SR14	4MHz	disconnect	connect	-
SR15	NMI to	normal	ground	-
SR16	TDX Clock	internal	external	-
SR17	+12V from	A/B 14	A/B 38	-
SR18	-12V from	A/B 12	A/B 39	-
SR19	Ground from	A/B 13/15	A/B 37	-

CHARACTERISTICS: (CONTINUED)

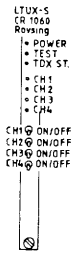
- Mechanical Dimensions:

Length:	315.5 mm
Height:	199.5 mm
Width:	17.2 mm
Weight:	0.6 kg

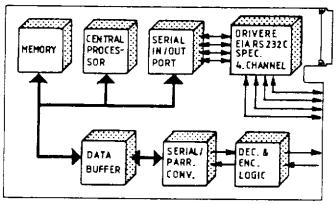
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Standard interface between the TDX-bus and communication lines, terminals, modems, etc.
- Provides up to 10 full duplex individual channels of up to 9.6Kb from the user applications to the TDX-bus
- The LTUX-S interfaces only to the LTUX-bus connector.
- When LTUX-S is used four 25 pin Cannon connectors normally will be placed on the back panel of the LTUX crate.
- Pin configuration of the Cannon connectors are according to CCITT V24 and it is possible to select DTE or DCE interface.
- Reliability:
  MTBF = 74600 H/F
- Power Consumption:

+5V:	2.5A
+12V:	0.1A *
-12V:	0.1A *

* V24 Channels not connected

(CONTINUED)

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

S1: LSB | | | | | | | MSB

PIN1

TDX DEVICE NO.

S2: | | | | |

PIN1

MODE
APPLICATION DEFINED

Switch ON ~0
Switch OFF ~1

Straps Summary:

SR1: A: NMI will be forced low (active)
B: N.C.

SR2: A: One wait state will be entered
B: N.C.

SR3: Selects memory type (UA20)
UA20/21 can be connected to 3 different signals
A: A11
B: Vcc
C: WR

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Physical Dimension:
  (315.5 x 221.5 x 17.1) mm
  Weight: 0.6 kg

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.

PROGRAMMERS REFERENCE: (CONTINUED)

SR4: Selects memory type (UA20)
UA20/18 - Chip select can be chosen active low or high
A: Active high
B: Active low

SR5: Selects memory type (UA17)
Identical with SR3.

SR6: Selects memory type (UA17)
Identical with SR4.

SR7: A: Clock (4MHz) available to the system
B: N.C.

SR8: Selects memory type (UC20)
UC20/26 can be chosen as Vcc or as chip select active high.
A: Chip select active high
B: Vcc

SR9: Selects memory type (UC17)
Identical with SR8.

SR10: Selects SIO receiver clock (channel 1)
A: Internal clock (Clk-1)
B: External clock (1-115)

SR11: Selects SIO transmitter clock (channel 1)
A: Internal clock (Clk-1)
B: External clock (1-113)

SR12: Selects SIO receiver clock (channel 2)
A: Internal clock (Clk-1)
B: External clock (2-115)

SR13: Selects SIO transmitter clock (channel 2)
A: Internal clock (Clk-1)
B: External clock (2-113)

SR14: Selects SIO receiver clock (channel 3)
A: Internal clock (Clk-2)
B: External clock (3-115)

SR15: Selects SIO transmitter clock (channel 3)
A: Internal clock (Clk-2)
B: External clock (3-113)

SR16: Selects receiver clock (channel 4)
A: Internal clock (Clk-2)
B: External clock (4-115)

SR17: Selects SIO transmitter clock (channel 2)
A: Internal clock (Clk-2)
B: External clock (4-113)

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

SR18: Clock select (channel 1)
- A: Internal clock (Clk-1) is transmitted on 1-114
- B: N.C.

SR19: Clock select (channel 2)
- A: Internal clock (Clk-1) is transmitted on 2-114
- B: N.C.

SR20: Clock select (channel 3)
- A: Internal clock (Clk-2) is transmitted on 3-114
- B: N.C.

SR21: Clock select (channel 4)
- A: Internal clock (Clk-2) is transmitted on 4-114
- B: N.C.

SR22: Selects memory type (UC20)
- A: A11
- B: WR
- C: Vcc

SR23: Selects memory type (UC20)
UC20/20 - Chip select can be chosen active low or high
- A: Active high
- B: Active low

SR24: Selects memory type (UC17)
UC17/23 can be connected to 3 different signals
- A: A11
- B: Vcc
- C: WR

SR25: Selects memory type (UC17)
UC17/20 - Chip select can be chosen active low or high
- A: Active high
- B: Active low

SR26: Selects dualized or single TDX-bus
- A: Dualized bus
- B: Single bus

As memory array the following will normally be used:

EPROM:	2K-2716
BIPOLAR PROM:	2K-825191
RAM:	2K-4816
	1K-4118
	1K-4801

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

EDGE CONNECTOR ASSIGNMENT:

The communication interfaces of the LTUX-S are specified as DTE.

Channel 1

PIN	CCITT-CIRCUIT	PIN	CCITT-CIRCUIT
A2	CT102	B2	CT102
A3		B3	
A4		B4	
A5	CT115	B5	
A6	CT109	B6	CT106
A7	CT105	B7	CT108
A8	CT103	B8	CT104

Channel 2

PIN	CCITT-CIRCUIT	PIN	CCITT-CIRCUIT
A10	CT102	B10	CT102
A11		B11	
A12		B12	
A13	CT115	V13	
A14	CT109	B14	CT106
A15	CT105	B15	CT108
A16	CT103	B16	CTY104

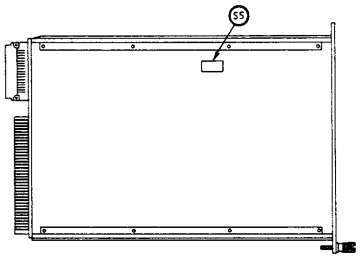
Channel 3

PIN	CCITT-CIRCUIT	PIN	CCITT-CIRCUIT
A19	CT102	B19	CT102
A20		B20	
A21		B21	
A22	CT115	B22	
A23	CT109	B23	CT106
A24	CT105	B24	CT108
A25	CT103	B25	CT104

Channel 4

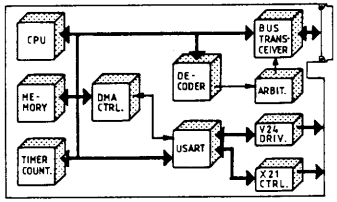
PIN	CCITT-CIRCUIT	PIN	CCITT-CIRCUIT
A27	CT102	B27	CT102
A28		B28	
A29		B29	
A30	CT115	B30	
A31	CT107	B31	CT106
A32	CT105	B32	CT108
A33	CT103	B33	CT104

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- The LTUX-M-X21 is a general V28/X21 communication interface module.
- 1 channel for X21 communication interface.
- 1 channel for CV24/V28 communication interface.
- Characteristics for line communication interface:
  Async. transmission
  Sync. transmission
  HDLC/SDLC communication protocols
  Baud rates from DC to 48K
- Interface to a general structured data, address on control bus called micro bus.
- Able to work in master/slave or multimaster configuration.
- Power Consumption:

+5V:	2.5A
-12V:	0.2A
-12V:	0.2A

- MTBF: 56,200 hours

PROGRAMMERS REFERENCE:

I/O addresses external
0-7FH

Internal I/O address space:
80-FFH

UE20	8253	80H
UD9	DMA2	90H
UD11	DMA1	A0H
UE14	CTC	B0H
UD6	8255	C0H
UH21	DIP-SW	D0H
UD3	SIO1	E0H
UD1	SIO2	F0H

Program memory area:
0H-3FFFH

(CONTINUED)

- Mechanical Dimensions:

Length:	315.5 mm
Height:	221.0mm
Width:	17.2 mm
Weight:	0.6 kg

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.

PROGRAMMERS REFERENCE: (CONTINUED)

Data memory area:

4000H - 7FFFH

External memory addresses:

8000-FFFFH

Switch S5 can be read by the CPU

1	2	3	4	5	6	7	8
D0	D1	D2	D3	D4	D5	D6	D7

"0" CLOSED
"1" OPEN

Strap Overview

			Pos. A	Pos. B	Pos. C
SR34	WAIT		enable	disable	
SR8	WR	PROM0	A11	VCC	WR
SR10	CS	0	PROM0	PROM0	RD
SR11	WR	PROM	A11	VCC	WR
SR12	CS	RAM0	MB0	MB0	
SR13	WR	PROM	A11	VCC	WR
SR20	CS	RAM1	MB1	MB1	
SR	WR	PROM	A11	VCC	WR
SR43	CS	RAM2	MB2	MB2	
SR19	A11	PROM	A11	VCC	WR
SR18	CS	RAM3	MB3	MB3	
SR22	A11	PROM/	A11	VCC	WR
SR21	CS	RAM4	MB4	MB4	
SR25	A11	PROM/	A11	VCC	WR
SR24	CS	RAM5	MB5	MB5	
SR28	A11	PROM/	A11	VCC	WR
SR27	CS	RAM6	MB6	MB6	
SR32	NMI	enable			
SR2			to u-Bus	From u-Bus	
SR7	X-clk.		Ext. clk.	4MHz/2	
SR33	BUSY		Disable	Enable	
SR35	C113 ch. 2		Enable		
SR6	T*CB		C114	Clk. 2	
SR5	R*CB		C115	Clk. 2	
SR17	CE	RAM/PROM3	MB3	MB3	
SR20	CE	RAM/PROM4	MB4	MB4	
SR23	CE	RAM/PROM5	MB5	MB5	
SR26	CE	RAM/PROM6	MB6	MB6	
SR31	IEO		I-SIO	PROUT	
SR1	RESERT		To u-Bus	From u-Bus	
SR3	Int. OSC			Enable	
SR9	RD		RD		
SR30	RD		MB0	RD	
SR29	RD	MB1	RD		

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Setting up the PPI

8255

PA0 —
PA1 —
PA2 — $\overline{\text{CLEAR}}$
PA3 — $\overline{\text{BAKEN}}$

$\overline{\text{CLEAR}}$	0 : CLEAR
	1 : NORMAL
$\overline{\text{BAKEN}}$	0 : BAK ENABLE
	1 : NORMAL

EDGE CONNECTOR ASSIGNMENT:

X21 Channel

PIN	CCITT CIRCUIT	PIN	CCITT CIRCUIT
A2	GND	B2	GND
A3	T(A)	B3	G(A)
A4	C(A)	B4	G(A)
A5	R(A)	B5	R(B)
A6	I(A)	B6	I(B)
A7	S(A)	B7	S(B)

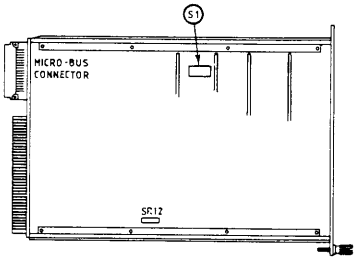
V24/V28 Channel

PIN	CCITT CIRCUIT	PIN	CCITT CIRCUIT
A10	GND	B10	GND
A11	CT111	B11	CLK
A13	CT115		
A14	CT109	B14	CT106
A15	CT105	B15	CT108
A16	CT103	B16	CT104

DATASHEET FOR:

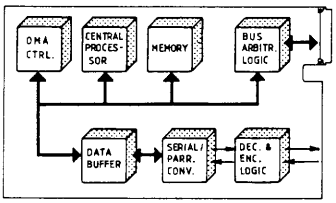
LTUX-M-FE CR1090S/000--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- The LTUX-M-FE is an intelligent slave module which interfaces one or two TDX-busses to one or several LTUX-M-CPUs.
- Bandwidth at TDX-Bus - 100 K Baud full duplex
- Reliability:
  MTBF: 134000 H/F
- Power Consumption:
  +5V: 2A
  -12V: 0.1A
- Physical Dimension:
  (315.5 x 221.5 x 17.1) mm
- BIT. A built-in test procedure is initiated upon power up and a red test LED is extenguished after app. 10 sec. if it has been carried out sucessfully.
- Weight: 0.6 kg

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

S1: LSB | | | | | | | MSB

PIN1

TDX DEVICE NO.

SWITCH ON~0
SWITCH OFF~1

Addressing:

The base address of data buffer seen from LTUX-M-CPU is 8400_H

All communication between the LTUX-M-FE and the LTUX-M-CPUs takes place in the LTUX-M-FE RAM. The RAM has a structure which divides it into a ringbuffer for outgoing data, a ringbuffer for ingoing data and a area for interchanging status and control information

Straps Summary:

SR1: A: NMI will be forced low (active)
B: N.C.

SR2: One wait state will be entered

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

SR3: Selects memory type (UA3)
UA3/21 can be connected to 3 different signals
A: A11
B: Vcc
C: WR

SR4: Selects memory type /UA3)
UA3/18 - Chip select can be chosen active low or high
A: Active high
B: Active low

SR5: Selects memory type (UA6)
Identical with SR3.

SR6: Selects memory type (UA6)
Identical with SR4.

SR7: Selects memory type (UA9)
Identical with SR3.

SR8: Selects memory type (UA9)
Identical with SR4.

SR9: Selects memory type (UA12)
UA12/23 can be chosen as A11 or WR
A: A11
B: WR

SR10: Selects memory type (UA12)
UA12/26 can be chosen as Vcc or as chip select active high
A: Chip select active high
B: Vcc

SR11: Selects memory type (UA12)
UA12/20 - Chip elect can be chosen active low or high
A: Active high
B: Active low

SR12: Selects dualized or single TDX bus
A: Dualized bus
B: Single bus

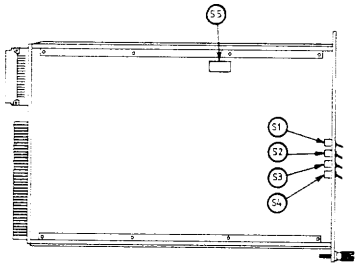
As memory array the following will normally be used:

EPROM: 2K-2716
BIPOLAR PROM: 2K-82S191
RAM: 2K-4816
1K-4118

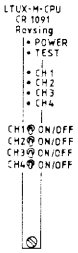
DATASHEET FOR:

LTUX-M-CPU CR 1091S/000--/00

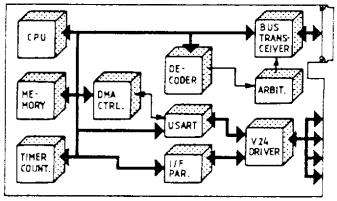
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- The LTUX-M-CPU ' is a general microcomputer board
- 4 channels for serial communications:
  Async. transmission
  Sync. transmission
  HDLC/SDLC communication protocols
  Baud rates from DC to 48K
- Interface to a general structured data, address and control bus called Micro Bus.
- Able to work in master/slave or multimaster configuration
- Power Consumption:

+5V:	2.5A
-12V:	0.1A *
-12V:	0.1A *

  * V24 Channels not connected
- Mechanical Dimensions:

Length:	315.5 mm
Height:	221.5 mm
Width:	17.2 mm
Weight:	0.6 kg

(CONTINUED)

PROGRAMMERS REFERENCE:

I/O addresses external:
0-7FH

Internal I/O address space:

	80-FFH	
UE20	8253	80H
UD9	DMA2	90H
UD11	DMA1	A0H
UE14	CTC	B0H
UD6	8255	C0H
UH21	DIP-SW	D0H
UD3	SIO1	E0H
UD1	SIO2	F0H

Program memory area:
0H-3FFFH

Data memory area:
4000H - 7FFFH

External memory addresses:
8000-FFFFH

Switch S5 can be read by the CPU

1	2	3	4	5	6	7	8		
D0	D1	D2	D3	D4	D5	D6	D7	"0"	CLOSED
								"1"	OPEN

S1 - 4 disable the serial channels.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- BIT. A built-in test procedure is initiated upon power up and a red test LED is extinguished after app. 10 sec. if it has been carried out successfully.

- MTBF: 55,200 hours

PROGRAMMERS REFERENCE: (CONTINUED)

Strap Overview

			Pos. A	Pos. B	Pos. C
SR1	WAIT		enable	disable	
SR2	WR	PROM0	A11	VCC	WR
SR3	CS	0	PROM0	PROM0	RD
SR4	WR	PROM	A11	VCC	WR
SR5	CS	RAM0	MB0	MB0	
SR6	WR	PROM	A11	VCC	WR
SR7	CS	RAM1	MB1	MB1	
SR8	WR	PROM	A11	VCC	WR
SR9	CS	RAM2	MB2	MB2	
SR10	A11	PROM	A11	VCC	WR
SR11	CS	RAM3	MB3	MB3	
SR12	A11	PROM/	A11	VCC	WR
SR13	CS	RAM4	MB4	MB4	
SR14	A11	PROM/	A11	VCC	WR
SR15	CS	RAM5	MB5	MB5	
SR16	A11	PROM/	A11	VCC	WR
SR17	CS	RAM6	MB6	MB6	
SR18	NMI	enable			
SR19			to u-Bus	From u-Bus	
SR20	X-clk.		Ext. clk.	4MHz/2	
SR21	BUSY		Disable	Enable	
SR22	C113 ch. 2		Enable		
SR23	T*CA		C114	Clk. 1	
SR24	R*CA		C115	Clk. 1	
SR25	C113 ch. 2		Enable		
SR26	T*CB		C114	Clk. 2	
SR27	R*CB		C115	Clk. 2	
SR28	C113 ch. 3		Enable		
SR29	T*CC		C114	Clk. 3	
SR30	R*CC		C115	Clk. 3	
SR31	C113 ch. 4		Enable		
SR32	T*CD		C114	Clk. 3	
SR33	R*CD		C115	Clk. 3	
SR34	CE	RAM/PROM3	MB3	MB3	
SR35	CE	RAM/PROM4	MB4	MB4	
SR36	CE	RAM/PROM5	MB5	MB5	
SR37	CE	RAM/PROM6	MB6	MB6	
SR38	IEO		I-SIO	PROUT	
SR39	RESET		To u-Bus	From u-Bus	
SR40	Int. OSC			Enable	
SR41			SYNCA to	DCDC	
SR42			SYNCA to	DCDA	
SR43	RD		RD		
SR44	RD		RD	MB0	
SR45	RD		RD	MB1	

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

EDGE CONNECTOR ASSIGNMENT:

The communication interfaces of the LTUX-M-CPU are specified as DTE.

Channel 1

PIN	CCITT-CIRCUIT	PIN	CCITT-CIRCUIT
A2	CT102	B2	CT102
A3	CT111	B3	CT114/113
A4	CT107	B4	CT116
A5	CT115	B5	CT117
A6	CT108	B6	CT106
A7	CT105	B7	CT108
A8	CT103	B8	CT104

Channel 2

PIN	CCITT-CIRCUIT	PIN	CCITT-CIRCUIT
A10	CT102	B10	CT102
A11	CT111	B11	CT114/113
A12	CT107	B12	CT116
A13	CT105	B13	CT117
A14	CT109	B14	CT106
A15	CT105	B15	CT108
A16	CT103	B16	CT104

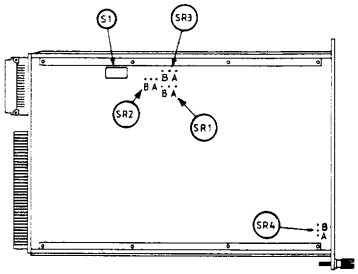
Channel 3

PIN	CCITT-CIRCUIT	PIN	CCITT-CIRCUIT
A19	CT102	B19	CT102
A20	CT111	B20	CT114/113
A21	CT107	B21	CT116
A22	CT115	B22	CT117
A23	CT109	B23	CT106
A24	CT105	B24	CT108
A25	CT103	B25	CT104

Channel 4

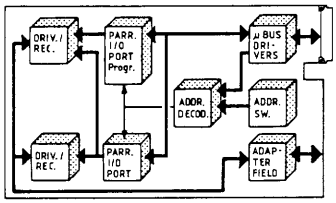
PIN	CCITT-CIRCUIT	PIN	CCITT-CIRCUIT
A27	CT102	B27	CT102
A28	CT111	B28	CT114/113
A29	CT107	B27	CT116
A30	CT 115	B30	CT117
A31	CT109	B31	CT106
A32	CT105	B32	CT108
A33	CT103	B33	CT104

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- General purpose Parallel In/Output module
- Communication via u-bus to LTUX-masters
- 32 parallel I/O lines
- Direction of all I/O lines is software controlled and selectable for groups of 8 I/O lines
- Input lines can be equipped with opto couplers
- Output lines can be equipped with Reed Relays
- Adapter field which makes it able to fulfill all interface requirements for low power controlling.
- MTBF: 250,000 hours*
- Mechanical Dimensions:
  - Length: 315.5 mm
  - Height: 221.5 mm
  - Width: 17.2 mm
  - Weight: 0.6 kg
- Power Consumption:
  - 5V 800mA*

* Without any application circuits connected.

PROGRAMMERS REFERENCE:

Module Address

Address lines A3-A7 are used.

S1

1	2	3	4	5	6
A3	A4	A5	A6	A7	(A14)

Front Panel →

OPEN ~ "1"
CLOSED ~ "0"

A0-A2 used internally for the programmable I/O ports.

By straps SR2/B and SR1/B A14 and A15 can be connected for Register Addressing Mode.

SR3/A connects $\overline{MON}$ signal.

SR3/B disconnects $\overline{MON}$ signal.

$\overline{MON}$ signal is the answer from the PIO module.

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

SR4/B connects power to the 32 RED LED, one for each I/O signal.

SR4/A disconnects power to the 32 RED LED.

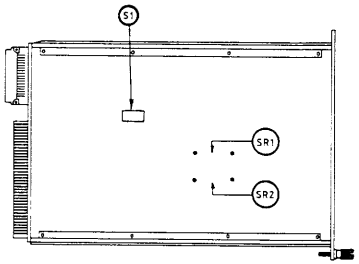
Pin assignment for the I/O area of the edge connector of the PIO-X

PIN	CIRCUIT			PIN	CIRCUIT		
A2	Signal	1	Pos.	B2	Signal	1	Neg.
A3	Signal	2	Pos.	B3	Signal	2	Neg.
A4	Signal	3	Pos.	B4	Signal	3	Neg.
A5	Signal	4	Pos.	B5	Signal	4	Neg.
A6	Signal	5	Pos.	B6	Signal	5	Neg.
A7	Signal	6	Pos.	B7	Signal	6	Neg.
A8	Signal	7	Pos.	B8	Signal	7	Neg.
A9	Signal	8	Pos.	B9	Signal	8	Neg.
A10	Signal	9	Pos.	B10	Signal	9	Neg.
A11	Signal	10	Pos.	B11	Signal	10	Neg.
A12	Signal	11	Pos.	B12	Signal	11	Neg.
A13	Signal	12	Pos.	B13	Signal	12	Neg.
A14	Signal	13	Pos.	B14	Signal	13	Neg.
A15	Signal	14	Pos.	B15	Signal	14	Neg.
A16	Signal	15	Pos.	B16	Signal	15	Neg.
A17	Signal	16	Pos.	B17	Signal	16	Neg.
A18	Signal	17	Pos.	B18	Signal	17	Neg.
A19	Signal	18	Pos.	B19	Signal	18	Neg.
A20	Signal	19	Pos.	B20	Signal	19	Neg.
A21	Signal	20	Pos.	B21	Signal	20	Neg.
A22	Signal	21	Pos.	B22	Signal	21	Neg.
A23	Signal	22	Pos.	B23	Signal	22	Neg.
A24	Signal	23	Pos.	B24	Signal	23	Neg.
A25	Signal	24	Pos.	B25	Signal	24	Neg.
A26	Signal	25	Pos.	B26	Signal	25	Neg.
A27	Signal	26	Pos.	B27	Signal	26	Neg.
A28	Signal	27	Pos.	B28	Signal	27	Neg.
A29	Signal	28	Pos.	B29	Signal	28	Neg.
A30	Signal	29	Pos.	B30	Signal	29	Neg.
A31	Signal	30	Pos.	B31	Signal	30	Neg.
A32	Signal	31	Pos.	B32	Signal	31	Neg.
A33	Signal	32	Pos.	B33	Signal	32	Neg.

DATASHEET FOR:

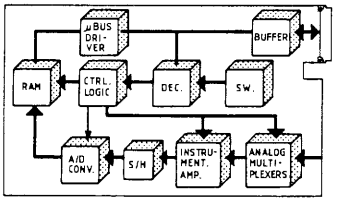
ADC-X CR1093S/000--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Analog to digital interface module between 16 analog signals and the CR micro system
- 12 bit A/D converter
- The 16 channels are multiplexed to a programable gain instrumentation amplifier before the A/D converter
- The sampling order and gain of the 16 analog siangls are programmed in a PROM
- The analog inputs are continously sampled and the digital value is updated in a 16*12 bit RAM
- Gain range is binary from 1 to 1024.
- Input range +/- 9 volt when gain is set to 1.
- Without adjustment the accuracy is 9 bit typical
- With adjustment the accuracy is 10 bit typical.
- Sampling time: typical 2 msec

(CONTINUED)

PROGRAMMERS REFERENCE:

Module Address

00 - 7F

Address lines A0-A7 are used. B2 addresses are used internally on the board, two I/O addresses for each analog channel.

Switch S1

1	2	3	4
SW	SW	SW	SW

Front Panel

CLOSED ~ "0"
OPEN ~ "1"

Switch 4 - 1	I/O address space
0	00 - 1F
1	08 - 27
2	10 - 2F
3	18 - 37
.	
.	
.	
F	78 - 7F, 00 - 17

Address line A15 can be connected by SR3 used for indirect Register Addressing.

SR1 connects 6.3V ref voltage to AN0
SR2 ground GAN0.

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Pin assignment for the I/O area of the edge connector of the ADC-X

PIN	CIRCUIT			PIN	CIRCUIT		
A2	Signal	1	AN0	B2	Signal	1	GAN0
A3	Signal	1		B3	Signal	1	
A4	Signal	2	AN1	B4	Signal	2	GAN1
A5	Signal	2		B5	Signal	2	
A6	Signal	3	AN2	B6	Signal	3	GAN2
A7	Signal	3		B7	Signal	3	
A8	Signal	4	AN3	B8	Signal	4	GAN3
A9	Signal	4		B9	Signal	4	
A10	Signal	5	AN4	B10	Signal	5	GAN4
A11	Signal	5		B11	Signal	5	
A12	Signal	6	AN5	B12	Signal	6	GAN5
A13	Signal	6		B13	Signal	6	
A14	Signal	7	AN6	B14	Signal	7	GAN6
A15	Signal	7		B15	Signal	7	
A16	Signal	8	AN7	B16	Signal	8	GAN7
A17	Signal	8		B17	Signal	8	
A18	Signal	9	AN8	B18	Signal	9	GAN8
A19	Signal	9		B19	Signal	9	
A20	Signal	10	AN9	B20	Signal	10	GAN9
A21	Signal	10		B21	Signal	10	
A22	Signal	11	AN10	B22	Signal	11	GAN10
A23	Signal	11		B23	Signal	11	
A24	Signal	12	AN11	B24	Signal	12	GAN11
A25	Signal	12		B25	Signal	12	
A26	Signal	13	AN12	B26	Signal	13	GAN12
A27	Signal	13		B27	Signal	13	
A28	Signal	14	AN13	B28	Signal	14	GAN13
A29	Signal	14		B29	Signal	14	
A30	Signal	15	AN14	B30	Signal	15	GAN14
A31	Signal	15		B31	Signal	15	
A32	Signal	16	AN15	B32	Signal	16	GAN15
A33	Signal	16		B33	Signal	16	

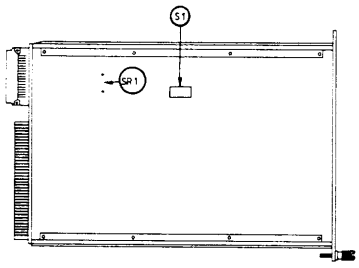
CHARACTERISTICS: (CONTINUED)

- Power Supply:
  - +5V: max 1.2A
  - +12V: max. 0.1A
  - -12V: max. 0.1A
- Dimensions: (315.5 x 221.5 x 17.1)
  Weight: 0.6 kg
- MTBF: 80,000 hours

DATASHEET FOR:

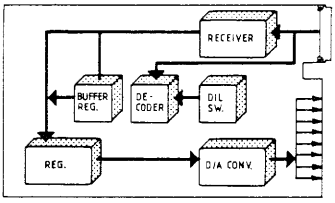
DAC-X CR 1094S/0XX--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Digital to analog interface module between the TDX system and up to 8 analog outputs.
- 12 bit D/A converters
- With adjustment 10 bit accuracy.
- Without adjustment 8 bit accuracy
- Output are short circuit protected
- Output range: +/- 8V, +/- 5mA
- Power Consumption:
  - +5V: max. 1.0A *
  - +12V: max. 0.2A *
  - -12V: max. 0.3A *
- * Without any application circuits connected.
- Mechanical Dimensions:
  - Length: 315.5 mm
  - Height: 221.5 mm
  - Width: 17.2 mm
  - Weight: 0.6 kg
- MTBF: 120,000 hours

PROGRAMMERS REFERENCE:

Module Address

Address lines A3 - A7 are used

	1	2	3	4	
S					Front Panel →
	A4	A5	A6	A7	

CLOSED ≙ "0"
OPEN ≙ "1"

The DAC-X uses 16 addresses for data and control of the channels.

By strap SR1 A15 can be connected for Register Addressing Mode.

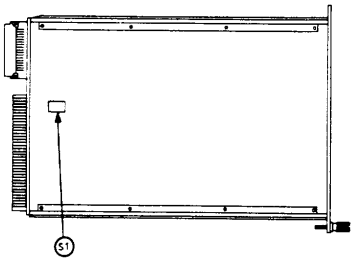
(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Pin assignment for the I/O area of the edge connector of the DAC-X

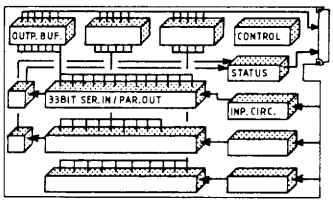
PIN	CIRCUIT			PIN	CIRCUIT		
A2	Signal	1	AN0	B2	Signal	1	GND0
A3	Signal	1	GND	B3	Signal	1	GND
A4	Signal	2	AN1	B4	Signal	2	GND1
A5	Signal	2	GND	B5	Signal	2	GND
A6	Signal	3	AN2	B6	Signal	3	GND2
A7	Signal	3	GND	B7	Signal	3	GND
A8	Signal	4	AN3	B8	Signal	4	GND3
A9	Signal	4	GND	B9	Signal	4	GND
A10	Signal	5	AN4	B10	Signal	5	GND4
A11	Signal	5	GND	B11	Signal	5	GND
A12	Signal	6	AN5	B12	Signal	6	GND5
A13	Signal	6	GND	B13	Signal	6	GND
A14	Signal	7	AN6	B14	Signal	7	GND6
A15	Signal	7	GND	B15	Signal	7	GND
A16	Signal	8	AN7	B16	Signal	8	GND7
A17	Signal	8	GND	B17	Signal	8	GND

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Interface between:
  Bosch FE Data Lines
  and CR Micro System
- Adapts to Bosch FE Data Lines Signal Specifications
- 3 channels input:
  Master Time Code
  Tape Time Code
  User Bits
- Master Time Code Input:
  High impedance
- Tape Time Code and User Bits Internal Terminated with standard 75 ohm
- Status Information from:
  Master Time Code
  Tape Time Code
- Dimensions:
  Height: 222.0 mm
  Weight: 17.1 mm
  Length: 305.0 mm
  Weight: 0.6 kg
- MBTF: 147,100 hours.

PROGRAMMERS REFERENCE:

MODULE ADDRESS:

The registers are accessed as memory locations, i.e. with read instructions.

SWITCH S1

1	2	3	4
SW	SW	SW	SW

Front Panel →

Switch 4-1	Memory address
0	Not used
1	4000
2	5000
3	6000
4	7000
5	8000
6	9000
7	A000
8	B000
9	C000
A	D000
B	E000
C	Not used
D	Not used
E	Not used
F	Not used

Registers are read only.

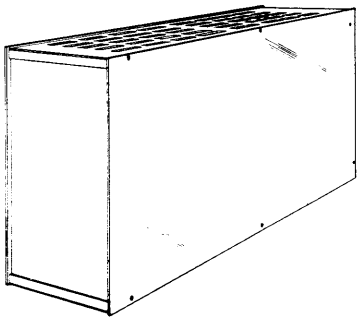
(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

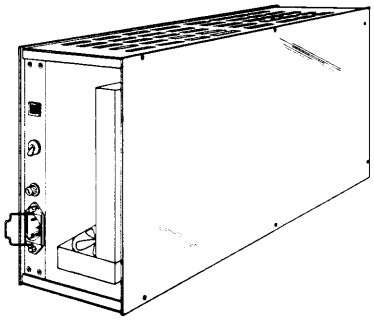
Register access is obtained on the following addresses:

Module Address		
	+ 0	MTC Frame
	+ 1	MTC Sec.
	+ 2	MTC Min.
	+ 3	MTC Hour
	+ 4	TCC Frame
	+ 5	TCC Sec.
	+ 6	TCC Min.
	+ 7	TCC Hour
	+ 8	UB LSW
	+ 9	UB
	+10	UB
	+11	UB MSW
	+12	STATUS

CHASSIS for X-NET DEVICES CR 2505-/000--/00



Front View



Rear View

CHARACTERISTICS:

- Table top chassis for LTUX modules.
- Capable of housing max. eight LTUX modules
- Mechanical Dimensions:
  Height: 230 mm
  Width: 140 mm
  Depth: 500 mm
  Weight: 8 kg (max. config.)
- Built-in blower unit
- Power supply:
  5V: 10A
  +/-12V: 1A
- Micro bus for five modules
- Mains power plug for 220V/50Hz
- Fuse, switch and light indicator for 220V
- Three CR2505 chassis' may be assembled by use of assembly kit to form a 19" crate unit for LTUX modules

INSTALLATION & CONFIGURATION INSTRUCTION:

Main Power Connector:

Voltage/Freq.: 220V/50Hz
Power: Max. 220VA

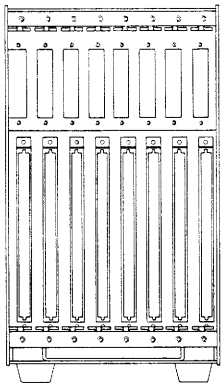
(CONTINUED)

- Slots for three back panels on the rear of the chassis.
- MTBF: 38,000 hours

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

SLOT SPECIFICATIONS

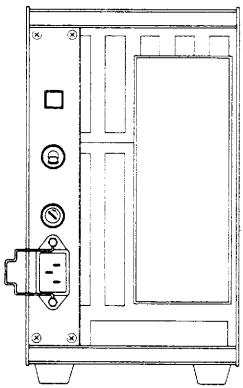
FRONT VIEW



Slot No. 1 2 3 4 5 6 7 8

Note: Micro Bus is only accessible from slot Nos. 2 - 5.

REAR VIEW



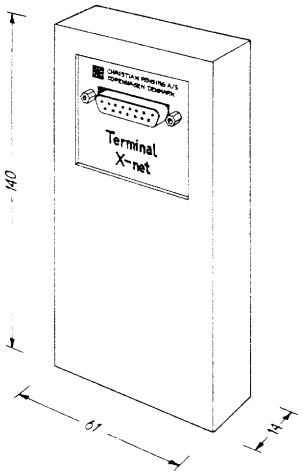
Slot No. 8 7 6 5 4 3 2 1

Slot Nos. 1 - 6: Back panels or cover panels

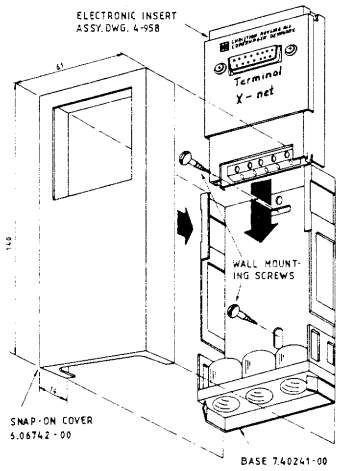
Slot Nos. 7 - 8: Mains Power Panel

DATASHEET FOR:

X-NET WALL OUTLET CR 2510-/000--/00



X-net Wall Outlet

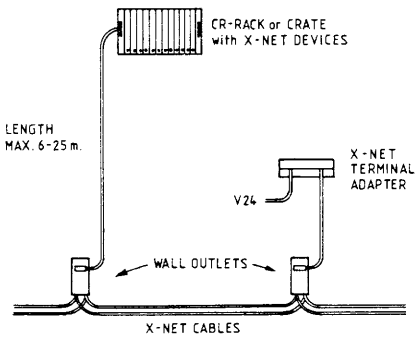


Exploded View

CHARACTERISTICS:

- Connects X-NET controller or X-NET devices with the X-NET cables.
- Provides galvanic isolation between X-NET device and X-NET cable.
- Power supplied from X-NET device (5V, 2A).
- Max. distance between X-NET device and X-NET WALL OUTLET 6 - 25 m dependent on cable used.

Locating example :



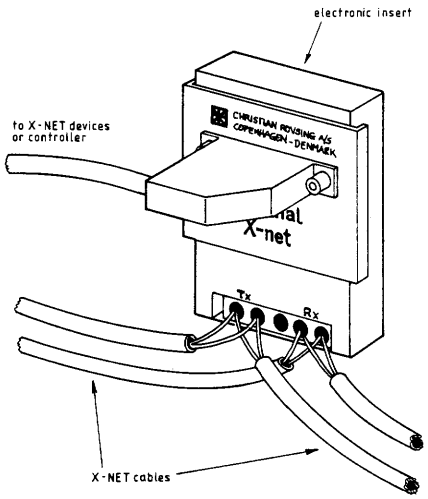
(CONTINUED)

CHARACTERISTICS: (CONTINUED)

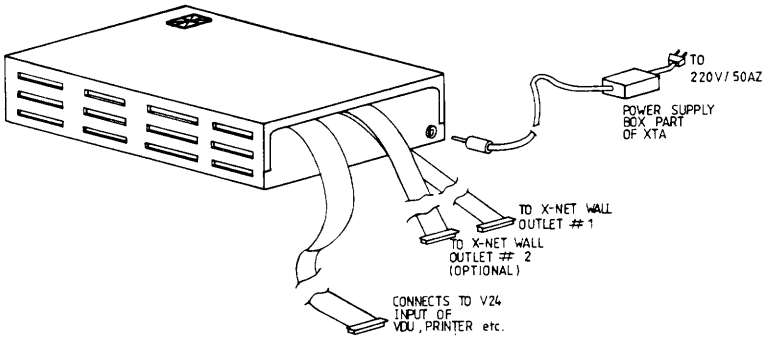
- Standard RS422 interface between outlet and X-NET device.
- X-NET cable transmitter in HI-Z state when

  1) logically disabled with DISAB signal from device
  2) X-NET device is powered down
  3) cable between outlet and X-NET device is removed

- X-NET cable transmitter outputs 13V peak-peak differential over cable impedance.
- X-NET cable receiver has input impedance of 24 Kohm and handles signals from 130mV peak-peak differential and up.
- X-NET devices are connected with outlet by means of a 15 pin standard multiconnector.



- Installation of X-NET cables is performed as ordinary installation wiring work. Outlets are attached to wall in the same way as ordinary power outlet wall sockets.
- X-NET cables are connected with outlet by means of screw-terminals



CHARACTERISTICS:

- CR2550-/010--/00 Single X-Net I/F
  CR2550-/020--/00 Dualized X-Net I/F
- Table top or wall mountable unit with flat cables to one, optional two, X-NET wall outlets and to the interfaced peripheral device (VDU, Printer, etc.).
- Interfaces to a single or optionally dualized X-NET Bus system.
- Separated Power Supply Unit, connected to the XTA Box by means of a 2 m cable.
- Mechanical Dimensions:

  XTA Box:

Length:	188 mm
Width:	110 mm
Height:	40 mm
Weight:	500 g

  Power Supply:

Length:	188 mm
Width:	110 mm
Height:	70 mm
Weight:	1.0 kg

- MTBF: 130,000 hours
- Power: Voltage/freq.
  220V/50hz. (optional 110V)
  Max. 50VA

INSTALLATION & CONFIGURATION INSTRUCTION:

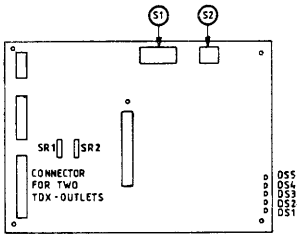
SEE FOLLOWING PAGE

- The XTA interfaces an asynchroneous terminal (V24, RS232 or Current Loop, 20mA) to the X-NET local network.
- Full or half duplex mode.
- Local or remote echo.
- 8 speeds selectable between 50 to 9600 Kbit/sec.
- Character mode:
  7 bits/char
  even parity
  1 stop bit
  (other available as customer option).
- Alternative speed and character mode down loadable via X-NET when COP function included in the X-NET.
- V24/RS232 circuits RTS (105) and CTS (106) transferred transparently through the X-NET for end to end flow control.
- Cable length:
  Flat cable(s) to X-NET WALL OUTLET(S):5 m
  Flat cable to Terminal I/F:0.5 m
  Power Cable: 2 m
  Mains Power Cable: 1 m

INSTALLATION & CONFIGURATION INSTRUCTION:

XTA Internal Switch & Jumper Options:

SET UP REFERENCE



DS1: TEST/TDX-CHANNEL SELECT LED
DS2: FUSE CONTROL LED FOR SUPPLYING TDX-OUTLET #1
DS3: " " " " " " #2
DS4: POWER LED FOR INTERNAL GENERATED "+12V"
DS5: " " " " " "-12V"

ECHO (S2):

- Switch on ("0"), local echo selected. Inputted characters are assembled into blocks of max. 16 characters and forwarded to the X-NET:

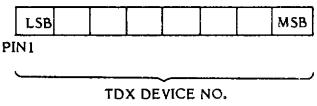
  1. when block full (16 char.), or
  2. forward timer run-out, or
  3. On Carriage Returning.

  The inputted characters are echoed locally on a character by character basis.

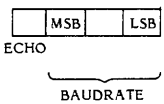
- Switch off ("1"), local echo not selected. As above except no local echo.

Switch Orientation and Definition:

S1:



S2:



SWITCH ON ~ 0
SWITCH OFF ~ 1

SR1 & SR2: Jumpers in A position, V24 Terminal I/F
Jumpers in B position, CL Terminal I/F

Baudrate (S2):			
	000	9600	bit/sec
	001	4800	bit/sec
	010	2400	bit/sec
	011	1200	bit/sec
	100	600	bit/sec
	101	300	bit/sec
	110	110	bit/sec
	111	50	bit/sec

(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

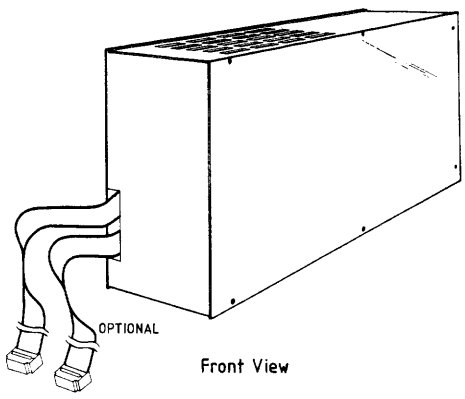
Cable to Terminal:

0.5 m flatcable terminated with 25p Canon connector (female).

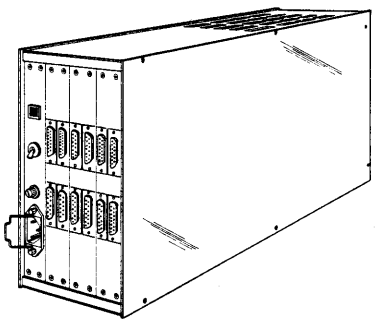
Pin Assigment:
(DCE)

Pin	CCITT No.	Name	Remarks
1	101	PROTECT.GND	Connected to 102
2	103	TXD	
3	104	RXD	
4	105	RTS	
5	106	CTS	
6	107	DSR	
7	102	SIGN.GND	
8	109	DCD	

X-NET PORT CR 2560-/0XX--/00



Front View



Rear View

CHARACTERISTICS:

CR2560-/004--/00 4 x V24 X-NET Port
CR2560-/008--/00 8 x V24 X-NET Port
CR2560-/012--/00 12 x V24 X-NET Port

- Table top X-NET unit containing all necessary modules to constitute a number of CCITT/V24 line communication interfaces (DTE).
- Mechanical Dimensions:
  Height: 230 mm
  Width: 140 mm
  Depth: 500 mm
  Weight: 8 kg (max. config.)
- Cable for X-NET terminal outlet connection (dual connection optional). Length: 4 meters
- May be connected to a single or optionally dualized X-NET Bus system.
- MTBF: 38,000 hours excluding front modules.

INSTALLATION & CONFIGURATION INSTRUCTION:

INTERFACE SPECIFICATIONS:

Back Panel Type 2

One back panel contains four 25 pin Cannon (male) connectors performing four V24 (DCE) line communication interfaces.

Pin Assignment

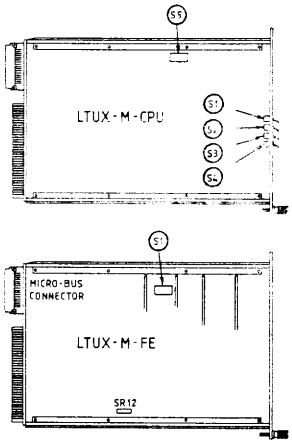
NAME	CIRCUIT	CANNON PIN.NO.	REMARKS
Pr. Gnd.	101	1	Protect Ground Strapable
Sign.Gnd	102	7	
DCD	109	8	
TxC-RxC/ TxC	114-115/ 113	15/17	Selectable by straps
RTS	108	20	
SBY-I	117/125	22/25	
S-SBY	116	24	
DSRT	111	11/23	
RTS	105	4	
CTS	106	5	
DSR	107	6	
RxD	104	3	
TxD	103	2	

(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

X-NET Port Internal Switch & Jumper Options:

SET UP REFERENCE



Please observe that the definition of S5 is the general case, variations can occur with certain type of X-NET ports.

ECHO (S5):

- Switch on ("0"), local echo selected. Inputted characters are assembled into blocks of max. 16 characters and forwarded to the X-NET:

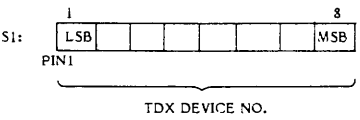
  1. when block full (16 char.), or
  2. forward timer run-out, or
  3. on Carriage Returning.

  The inputted characters are echoed locally on a character by character basis.

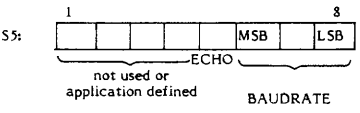
- Switch off ("1"), local echo not selected. As above except no local echo.

Switch Orientation and Definition:

LTUX-M-FE:



LTUX-M-CPU:



SWITCH ON ~ 0
SWITCH OFF ~ 1

Baudrate (S5):			
	000	9600	bit/sec
	001	4800	bit/sec
	010	2400	bit/sec
	011	1200	bit/sec
	100	600	bit/sec
	101	300	bit/sec
	110	110	bit/sec
	111	50	bit/sec

(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

Cables to X-NET Terminal Outlets

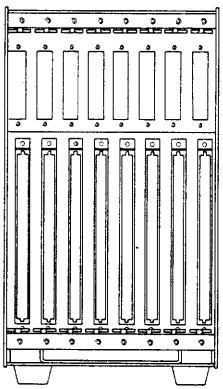
Two 15 wire flat cables (length: 4 m) terminated with 15 pin Cannon (male) connectors.

Main Power:

Voltage/Freq.: 220V/50Hz
Power: Max. 150VA

SLOT SPECIFICATIONS:

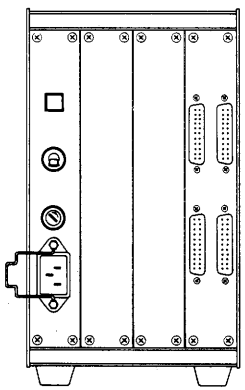
FRONT VIEW



Slots Nos. 1 2 3 4 5 6 7 8

Slot No. 1: BTM-X
Slot No. 2: LTUX-M-FE
Slot Nos. 3 - 5: LTUX-M-CPUs
Slot Nos. 6 - 8: Not used

REAR VIEW



Slot No. 8 7 6 5 4 3 2 1

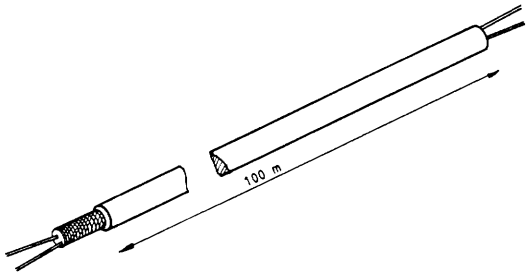
Slot Nos. 1-2, 3-4, 5-6: Back Panel Type 2
Slot Nos. 7 - 8: Main Power Panel

PROGRAMMERS REFERENCE:

For programmer's reference see the data sheet for LTUX-M-FE CR1090S, LTUX-M-CPU CR1091S.

DATASHEET FOR:

X-NET CABLE CR 2500-/000--/00

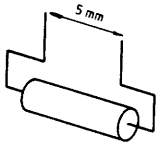


CHARACTERISTICS:

- Two core screened RF cable.
- Delivered in length of multiple 100 meters.
- Characteristic independence: 95 Ω

DATASHEET FOR:

X-NET CABLE TERMINATION CR 2501-/000--/00

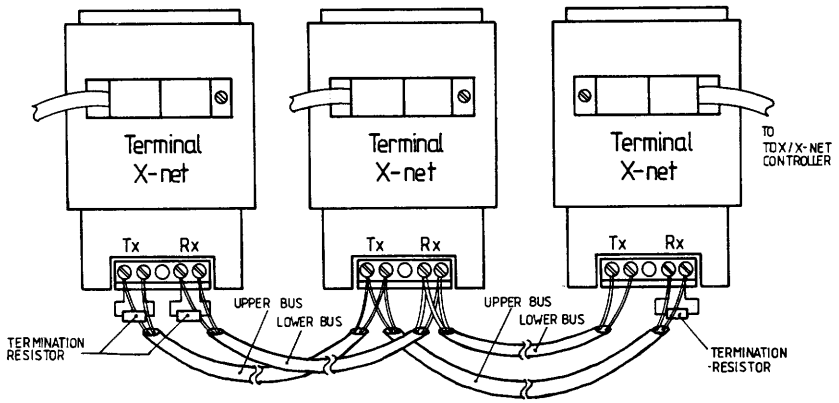


CHARACTERISTICS:

- 95Ω Terminal Resistor for TDX and X-NET cables.

Installation

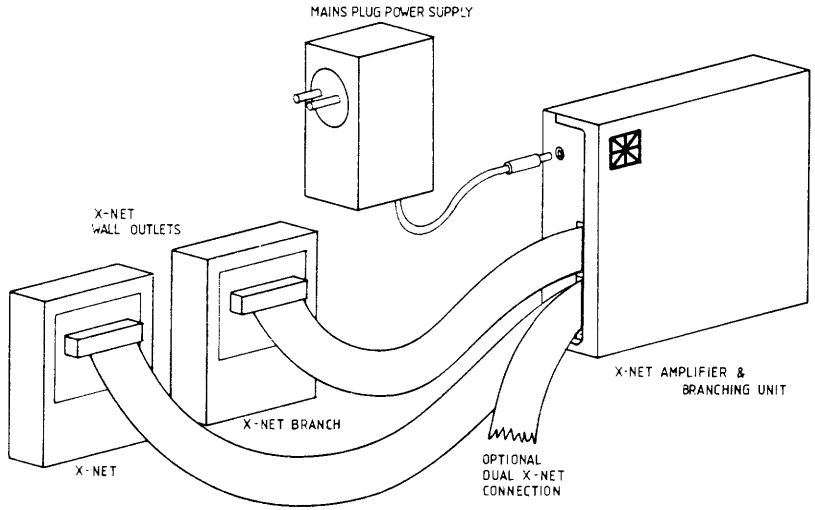
The termination resistor is mounted across the two wires of the TDX and X-NET cable on lower as well as upper bus on the farthest end of the cables seen from the TDX/X-NET Controller, and on the upper bus on the end of the cable nearest to the Controller.



NOTE: UPPER/LOWER BUS INTERCHANGED AT X-NET CONTROLLER WALL OUTLET.

DATASHEET FOR:

X-NET AMPLIFIER & BRANCHING UNIT CR2520-/0XX--/00



CHARACTERISTICS:

CR2520-/000--/00 Single X-Net/Single Branch
CR2520-/020--/00 Dual X-Net/Single Branch

- Kit consisting of a X-NET AMPLIFIER AND BRANCHING UNIT BOX, two X-NET Wall Outlets and a Power Supply Unit.
- Cable length (flat cables and power cabel): 500 mm.
- The units are to be mounted on walls, in racks, etc., and the power supply unit is directly plugged into a 220V/50Hz power plug.
- Extends the X-NET cable length to max. 2500 m, measured between the farthest point of the X-NET cable and the X-NET controller.
- Up to 2 XABs may be connected "in serial" between an X-Net station and the Controller.
- Optionally the XAB can connect a dualized X-Net to a single X-Net branch, see drawing overleaf.
- Dimensions (box unit):
  Length: 188 mm
  Width: 110 mm
  Height: 40 mm
  Weight: 1.0 kg

(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION:

INTERFACE SPECIFICATIONS

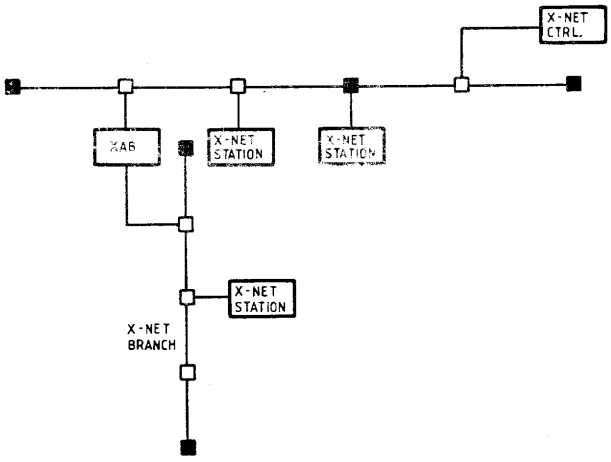
The kit interfaces to the X-NET cables as described in the data sheets for the X-NET WALL OUTLET.

- Power:
  Voltage/freq.: 220V/50Hz
  Power: Max. 50VA

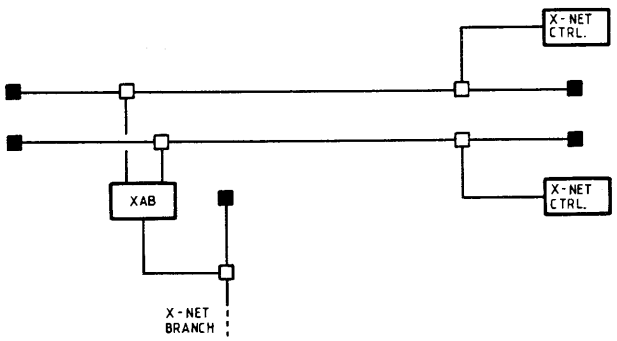
CHARACTERISTICS: (CONTINUED)

- MTBF: 181,000 hours

Single X-net to single X-net branch:

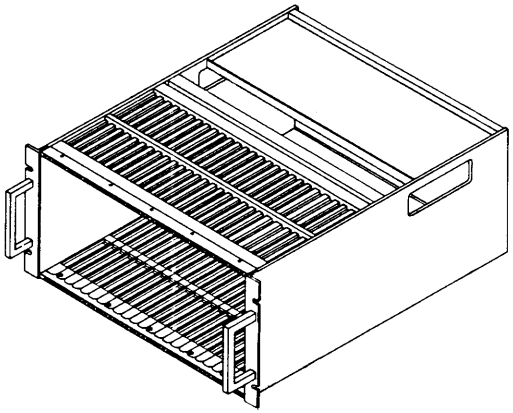


Dual X-net to single X-net branch:



DATASHEET FOR:

LTUX CRATE, 5U, CR 1081S/020--/00



CHARACTERISTICS:

- Standard CAMAC 19" crate. 5 units high (1 U = 44.45 mm). Confirms to specifications of CAMAC EUR 4100E.
- Standard housing for LTUX-modules.
- Capable of housing maximum 16 LTUX-modules (1M wide) and one 6M power supply.
- Mechanical Dimensions:
  Height: 221.5 mm
  Width: 483.0 mm
  Depth: 529.0 mm
- Weight (without modules):
  10 kg
- Equipped with four Micro Buses.
- Mains Power Panel for 220V AC INPUT.
- Fuse and switch for mains power.
- MTBF: 450,000 h

INSTALLATION & CONFIGURATION INSTRUCTION:

The crate is installed from the front of a rack and fixed with four screws. It should rest on a pair of supporting bars located just below the crate.

The protection ground of the rack is connected to the crate by means of a 6 mm^2 wire.

The inter-crate cabling supports the basic configuration except the flat cables between the connectors of the LTUX-Motherboard and the application defined Back Panels.

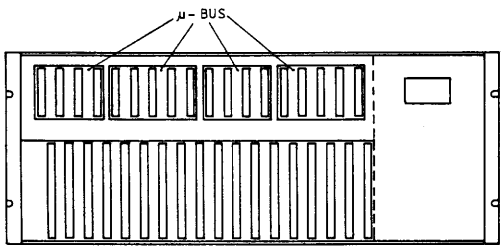
(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

Slot Specifications:

The basic crate is configured with four separate u-buses. The configuration may be changed by extending the u-buses with or without Bus Termination or by removing some or all the u-buses.

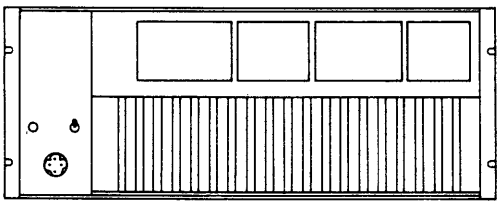
FRONT VIEW:



SLOT NO 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

Slot No.:	1:	Not used.
Slot Nos.:	2, 3, 4, 5:	May be configured with one LTUX-M-FE and one, two, or three LTUX-M-CPU, or with one LTUX-M-FE, one LTUX-M-CPU and PIO-X, ADC-X, DAC-X, DL-X, AMBE-X, etc.
Slot Nos.:	6, 7, 8, 9, 10:	
Slot Nos.:	11, 12, 13, 14, 15:	
Slot Nos.:	16, 17, 18, 19:	
Slot Nos.:	20, 21, 22, 23, 24:	Power Supply, CR8020S (Not included)

REAR VIEW:



SLOT NO 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

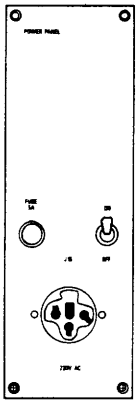
Slot No.:	1:	May be configured with BTM-X
Slot Nos.:	4, 5:	May not be configured with Back Panels Type 1,2,3 or 4.
Slot Nos.:	6, 7:	
Slot Nos.:	8, 9:	
Slot Nos.:	10, 11:	
Slot Nos.:	12, 13:	
Slot Nos.:	14, 15:	
Slot Nos.:	15, 17:	
Slot Nos.:	18, 19:	
Slot Nos.:	20, 21:	
Slot Nos.:	22, 23, 24, 25:	Mains Power Panel

(CONTINUED)

INSTALLATION & CONFIGURATION INSTRUCTION: (CONTINUED)

INTERFACE SPECIFICATIONS:

MAINS POWER PANEL:

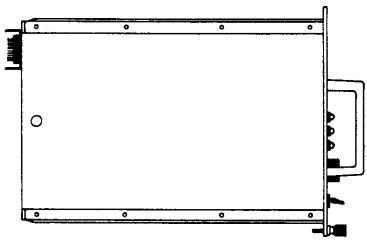


Voltage:	220V +20% -10%, 50Hz +/- 10%
Input Power:	TBS VAC
Power Plug:	Tüchel male (6.06709.01)

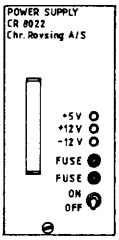
DATASHEET FOR:

POWER SUPPLY CR 8022S/000- -/00

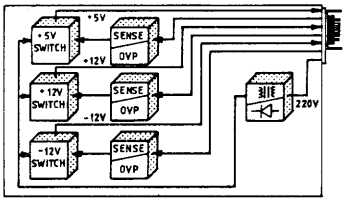
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Maximum Output:
  + 5V/32.0A
  +12V/ 2.8A
  -12V/ 1.2A
- Input Voltage:
  220 VAC +10/-15% or
  110 VAC +10/-15%
- Output will be maintained during one missing line cycle.
- Efficiency at max. load better than 75%.
- Constant current limit short circuit protection.
- OVP on all output voltages.
- Undervoltage protection.
- Parallel connection without derating possible.
- Remote Error Sensing.
- Noise and Ripple: <60mVpp.

PROGRAMMERS REFERENCE:

Main Fuses:

0.63A
3.15A

Parallel connection of power supplies possible +5V may be adjusted at the rear of the power supply. Norminal value:

$5.1V \leq V \leq 5.2V$

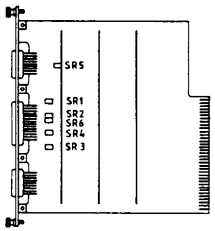
measured at +5V output with load.

- Mechanical Dimensions:
  Height: 221.5mm (5U)
  Width: 103.0mm (6M)
  Length: 305.0mm
- Weight: 4.8 kgs.

DATASHEET FOR:

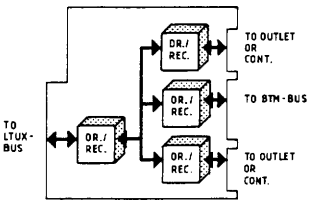
BTM-X CR 1082S/000--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- The BTM (Bus Termination Module) interfaces between the TDX outlet, CR 2510, and the BTM-Bus and the LTUX-Bus respectively.
  Several local LTUX crates can via the BTM-bus correspond to each other or at pleasure to one, or a pair of TDX outlets.
- Reliability:
  MTBF = 226000 H/F
- Power Consumption:
  +5V: 1A*
  * Without connected outlets.
- Physical Dimension:
  (184.5 x 221.5 x 17.1) mm
- Weight: 0.4 kg

PROGRAMMERS REFERENCE:

Strap Summary

SR1: Not used

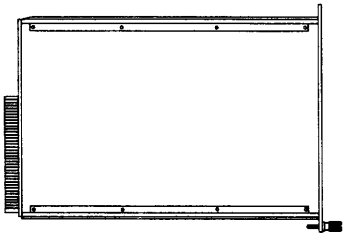
SR2: Closed if the BTM-X's must be grounded together via the BTM-bus.

SR3, SR4, SR5, SR6:
Terminates signals. Closed on those BTM-X's placed on the ends of the BTM-bus.

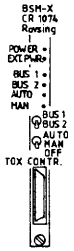
DATASHEET FOR:

BSM-X CR 1074S/000--/00

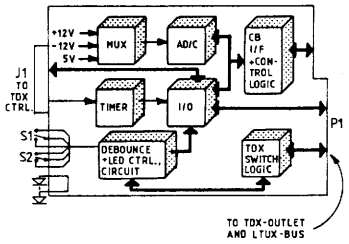
SET UP REFERENCE



FRONT PANEL



BLOCK DIAGRAM



CHARACTERISTICS:

- Configuration Bus Interface for Watchdog monitoring and control.
- Switch logic for dual separated TDX-Bus systems controlled from switches on the front panel or from the Watchdog.
- When manual mode is selected on the front panel the Watchsdog control of the TDX Bus switch is disabled.
- A TDX controller may be monitored and controlled by the Watchdog via a 25 pin common connector on the front panel.
- Two additional inputs and outputs to the Watchdog are available on P1.
- Dimensions:
  Height: 221.5 mm
  Depth: 315.5 mm
  Width: 17.1 mm
- Weight: 0.6 kg
- MTBF: 36,400 hours

PROGRAMMERS REFERENCE:

Switch Orientation and Definition

	4	3	2	1
S3:	A_3	A_2	A_1	A_0

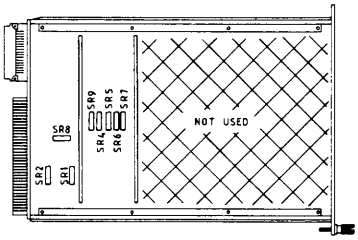
Configuration Bus Address Setting

Address determined by A0-A3;
MSB: A3
0~switch on; 1~switch off.

DATASHEET FOR:

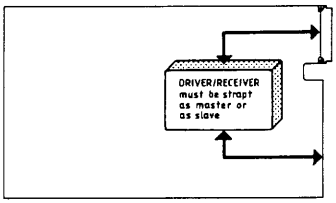
AMBE-X CR1084S/000--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Active extending of the microbus between two LTUX-crates
- Two modules are necessary
- The daisy chain interrupt facility is not available in the slave crate.
- Reliability:
  MTBF = 769230 H/F
- Power Consumption:
  +5V: 1A +/- 0.5A
- Physical Dimension:
  (305.0 x 221.5 x 17.1) mm
- Weight: 0.4 kg

PROGRAMMERS REFERENCE:

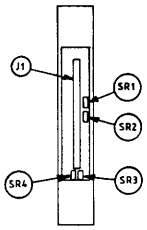
Strap Summary

	SR1	SR2	SR3	SR4	SR5	SR6	SR7	SR8	SR9
MASTER CONFIG.	A	NC	A	B	B	B	B	A	A
SLAVE CONFIG.	B	B	B	A	A	A	A	B	B

DATASHEET FOR:

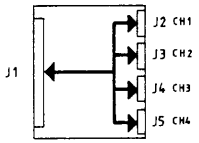
BACK PANEL TYPE 1 CR1085S/000--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Line communication interface panel for LTUX-S and LTUX-M CPU.
- Performs a DTE interface according to CCITT V24.
- 4 independent line communication interfaces.
- 4 connectors of 25 pin Cannon type (Male).
- Protection Ground (CCITT V24/101) strapable on each communication interface.

PROGRAMMERS REFERENCE:

Straps:

Protection Ground (CCITT V24/101) of the line communication channels is connected to system ground when the following straps are mounted:

SR1: Channel 1

SR2: Channel 2

SR3: Channel 3

SR4: Channel 4

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Pin assignment for the connectors of the communication interfaces:

Channel 1

Wire No.	Cannon Pin. No.	Wire No.	Cannon Pin No.
1	GND	2	GND
3	23/11	4	15
5	6	6	24
7	17	8	22
9	8	10	5
11	4	12	20
13	2	14	3
15	GND	16	GND

Channel 2

Wire No.	Cannon Pin. No.	Wire No.	Cannon Pin No.
17	GND	18	GND
19	23/11	20	15
21	6	22	24
23	17	21	22
25	8	26	5
27	4	28	20
29	2	30	3
31	GND	32	GND

Channel 3

Wire No.	Cannon Pin. No.	Wire No.	Cannon Pin No.
33	GND	34	GND
35	23/11	36	15
37	6	38	24
39	17	40	22
41	8	42	5
43	4	44	20
45	2	46	3
47	GND	48	GND

Channel 4

Wire No.	Cannon Pin. No.	Wire No.	Cannon Pin No.
49	GND	50	GND
51	23/11	52	15
53	6	54	21
55	17	56	22
57	8	58	5
59	4	60	20
61	2	62	3
63	GND	64	GND

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Specification of V24 DTE interface on Back Panel Type 1:

- Connected to LTUX-M-CPU

CPU Name	Circuit	Cannon Pin No.
Pr. GND	101	1
Sign. GND	102	7
DSRT	111	23/11
TxC	113-114	15
DSR	107	6
S-S BY	116	24
RxC	115	17
SBY-I	117	25/22
DCD	109	8
CTS	106	5
RTS	105	4
DTR	108	20
TxD	103	2
RxD	104	3

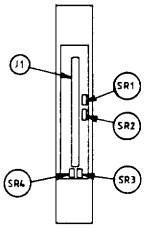
- Connected to LTUX-S

Name	Circuit	Cannon Pin No.
Pr. GND	101	1
Sign. GND	102	7
TxD	103	2
RxD	104	3
RTS	105	4
CTS	106	5
DTR	108	20
DCD	109	8
TxC	113/114	15
RxC	115	17

DATASHEET FOR:

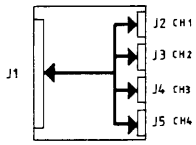
BACK PANEL TYPE 2 CR1086S/000--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Line communication interface panel for LTUX-S and LTUX-M-CPU.
- Performs a DCE interface according to CCITT V24.
- 4 independent line communication interfaces.
- 4 connectors of 25 pin Cannon type (male).
- Protection Ground (CCITT V24/101) strapable on each line communication interface.

PROGRAMMERS REFERENCE:

Straps:

Protection Ground (CCITT V24/101) of the line communication channels is connected to system ground when the following straps are mounted:

SR1: Channel 1

SR2: Channel 2

SR3: Channel 3

SR4: Channel 4

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Pin assignment for the connectors of the communication interfaces:

Channel 1

Wire No.	Cannon Pin No.	Wire No.	Cannon Pin No.
1	GND	2	GND
3	8	4	15/17
5	20	6	22/25
7	N.C.	8	24
9	11/23	10	4
11	5	12	6
13	3	14	N.C.
15	GND	16	GND

Channel 2

Wire No.	Cannon Pin No.	Wire No.	Cannon Pin No.
17	GND	18	GND
19	8	20	15/17
21	20	22	22/25
23	N.C.	21	24
25	11/23	26	4
27	5	28	6
29	3	30	N.C.
31	GND	32	GND

Channel 3

Wire No.	Cannon Pin No.	Wire No.	Cannon Pin No.
33	GND	34	GND
35	8	36	15/17
37	20	38	22/25
39	N.C.	40	24
41	11/23	42	4
43	5	44	6
45	3	46	N.C.
47	GND	48	GND

Channel 4

Wire No.	Cannon Pin No.	Wire No.	Cannon Pin No.
49	GND	50	GND
51	8	52	11/17
53	20	54	22/25
55	N.C.	56	24
57	11/23	58	4
59	5	60	6
61	3	62	N.C.
63	GND	64	GND

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

Specification of V24 DCE interface on Back Panel Type 2:

- Connected to LTUX-M-CPU

Name	Circuit	Cannon Pin No.
Pr. GND	101	1
Sign. GND	102	7
DCD	109	8
TxC-RxC/ TxC	114-115/ 113	15/17
RTS	108	20
SBY-I	117/125	22/25
S-SBY	116	24
DSRT	111	11/23
RTS	105	4
CTS	106	5
DSR	107	6
RxD	104	3
TxD	103	2

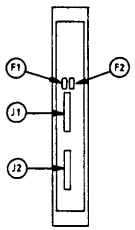
- Connected to LTUX-S

Name	Circuit	Cannon Pin No.
Pr. GND	101	1
Sign. GND	102	7
RxD	104	3
TxD	103	2
CTS	106	5
RTS	105	4
DSR	107	6
DCSD	111	(11)/23
TxC/ RxC-TxC	113/114-115	15/17

DATASHEET FOR:

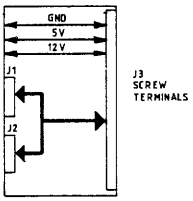
BACK PANEL TYPE 3 CR1087S/000--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Application interface panel for digital and analogue interface modules installed in LTUX-Crates.
- Two 16 pin connectors interface to one half of the I/O area of a PIO-X, ADC-X or DAC-X.
- Each pair of connectors on the screw terminals (1 and 2, 3 and 4, ...) perform a galvanic insulated or differential input/output to the interface module.
- 5V is via fuse supplied to one screw terminal.
- 12V is via fuse supplied to one screw terminal.
- Signal ground is conntected to two screw terminals.

PROGRAMMERS REFERENCE:

Fuses:

F1: Fuse for 5V terminal

F2: Fuse for 12V terminal.

(CONTINUED)

PROGRAMMERS REFERENCE: (CONTINUED)

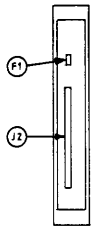
Terminal assignment of the Back Panel Type 3.

Flatcable Connector	Flatcable Wire No.	Screw Terminal No.
		1 (GND)
		2 (5V)
		3 (12V)
J1	1	4
	2	5
	3	6
	4	7
	5	8
	6	9
	7	10
	8	11
	9	12
	10	13
	11	14
	12	15
	13	16
	14	17
	15	18
	16	19
J2	1	20
	2	21
	3	22
	4	23
	5	24
	6	25
	7	26
	8	27
	9	28
	10	29
	11	30
	12	31
	13	32
	14	33
	15	34
	16	35
		36 (GND)

DATASHEET FOR:

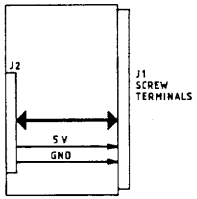
BACK PANEL TYPE 4 CR1088S/000--/00

SET UP REFERENCE



FRONT PANEL

BLOCK DIAGRAM



CHARACTERISTICS:

- Application interface panel for digital and analogue interface modules installed in LTUX-Crates.
- 64 pin connector interfaces to the I/O area of a PIO-X, ADC-X or DAC-X.
- The levels of input/output signals are referred to signal ground.
- 32 screw terminals for application interfaces.
- 5V is via fuse supplied to one screw terminal.
- Signal ground is connected to three screw terminals.

PROGRAMMERS REFERENCE:

F1: Fuse for 5V terminal.

(CONTINUED)

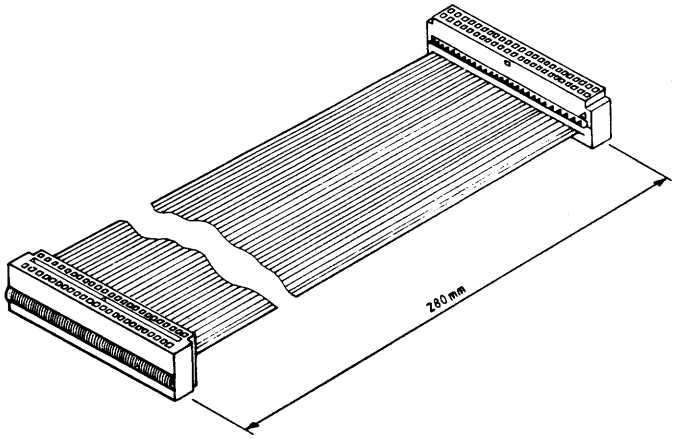
PROGRAMMERS REFERENCE: (CONTINUED)

Terminal assignment of the Back Panel Type 4:

Flatcable Wire No.	Screw Terminal No.	Flatcable Wire No.	
1	1	2	
3	2	4	
5	3	6	
7	4	8	
9	5	10	
11	6	12	
13	7	14	
15	8	16	
17	9	18	
19	10	20	
21	11	22	
23	12	24	
25	13	26	
27	14	28	Connected to
29	15	30	common
31	16	32	signal ground
33	17	34	
35	18	36	
37	19	38	
39	20	40	
41	21	42	
43	22	44	
45	23	48	
47	24	50	
49	25	52	
51	26	54	
53	27	56	
55	28	58	
57	29	60	
59	30	62	
61	31	64	
63	32		

DATASHEET FOR:

CABLE FOR BACK PANEL TYPE 1,2 or 4 CR1096-/028--/00

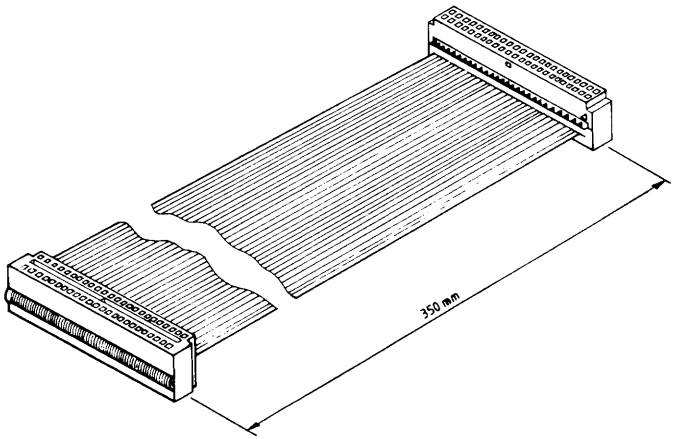


CHARACTERISTICS:

- Connects Back Panels to I/O area of the LTUX Motherboard connectors.
- 64 wires.
- 64 pin connectors (female).
- Length: 280 mm

DATASHEET FOR:

CABLE FOR BACK PANEL TYPE 1,2 or 4 CR 1096-/035/000

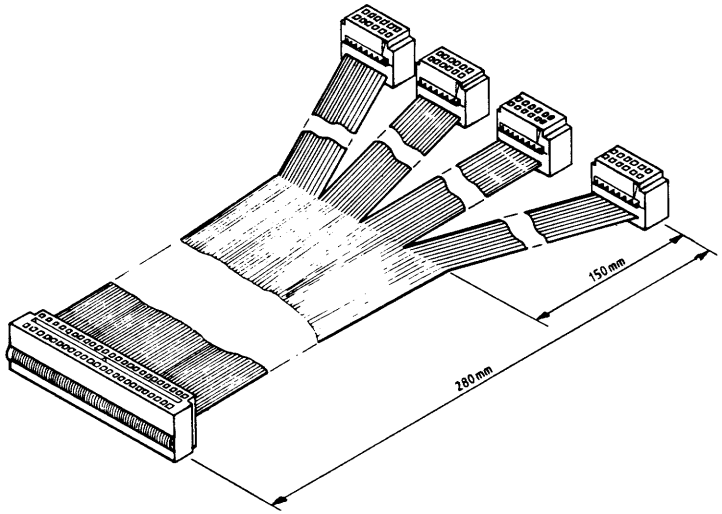


CHARACTERISTICS:

- Connects Back Panels to I/O area of the LTUX Motherboard connectors.
- 64 wires
- 64 pin connectors (female).
- Length: 350 mm

DATASHEET FOR:

CABLE FOR BACK PANEL TYPE 3 CR1097-/028--/00

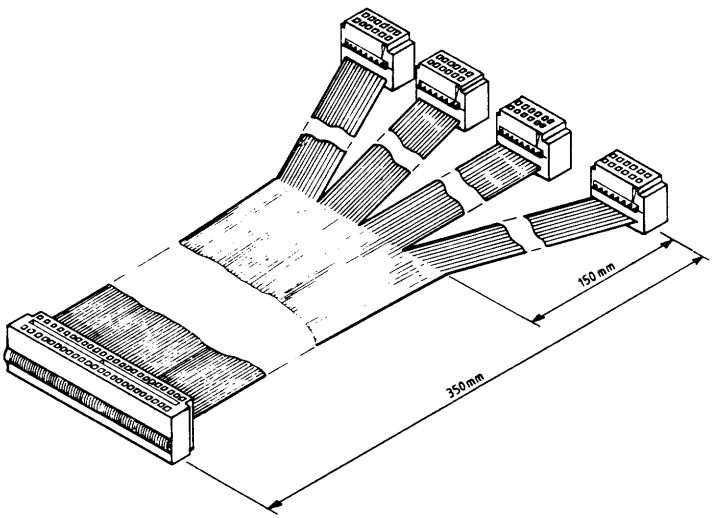


CHARACTERISTICS:

- Connects two Back Panels Type 3 to the I/O area of one LTUX Motherboard connector.
- 64 wired flat cable splitted into four parts of 16 wires each.
- One 64 pin connector (female) and four 16 pin connectors (female).
- Total length: 280 mm

DATASHEET FOR:

CABLE FOR BACK PANEL TYPE 3 CR 1097-/035--/00

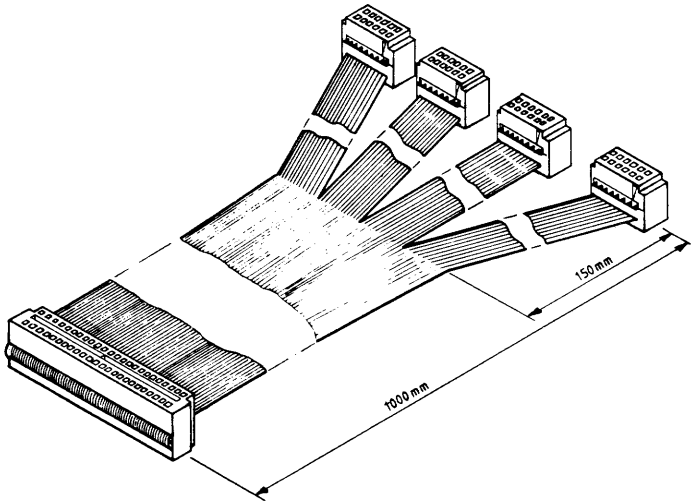


CHARACTERISTICS:

- Connects two Back Panels Type 3 to the I/O area of the LTUX Motherboard connector.
- 64 wired flat cable splitted into four parts of 16 wires each.
- One 64 pin connector (female) and four 16 pin connectors (female).
- Total length: 350 mm.

DATASHEET FOR:

CABLE FOR BACK PANEL TYPE 3 CR1097-/100--/00

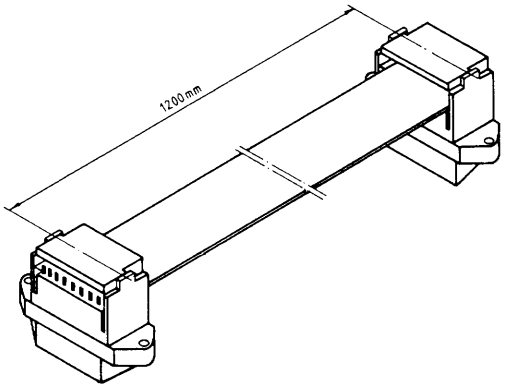


CHARACTERISTICS:

- Connects two Back Panels Type 3 to the I/O area of one LTUX Motherboard connector.
- 64 wired flat cable splitted into four parts of 16 wires each.
- One 64 pin connector (female) and four 16 pin connectors (female).
- Total length: 1000 mm

DATASHEET FOR:

CABLE CR 1069-/012--/00

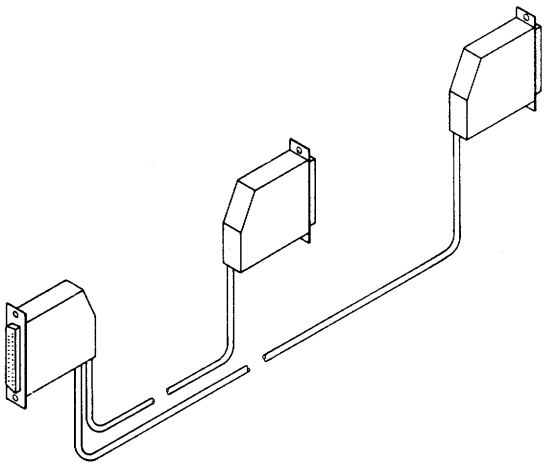


CHARACTERISTICS:

- BTM-X to Wall Outlet cable.
- 16 pol. flat cable.
- 15 pol. female connector (type 609-15S)
- 15 pol. male connector (type 609-15P)
- Length: 1200 mm

DATASHEET FOR:

CABLE CR 1071-/115--/00

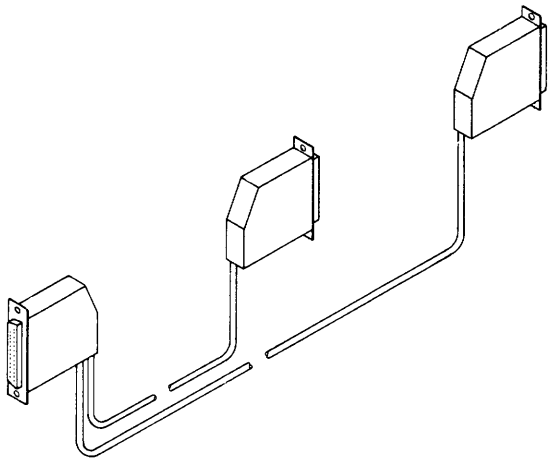


CHARACTERISTICS:

- TDX-Controller to dual Wall Outlet cable.
- Used with TDX-Controller no. 1.
- Two 15 pol. male connectors (type DAC-15P).
- One 25 pol. male connector (type DBC-25P).
- Length of both cords: 1500 mm.

DATASHEET FOR:

CABLE CR 1071-/215--/00



CHARACTERISTICS:

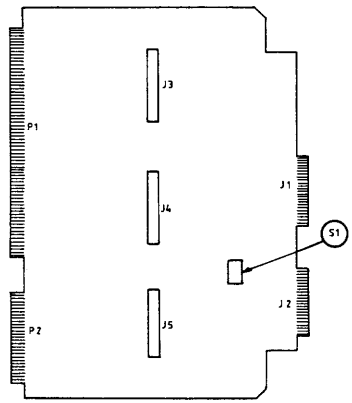
- TDX Controller to dual Wall Outlet cable.
- Used with TDX-Controller no. 2.
- Two 15 pol. male connectors (type DAC-15P).
- One 25 pol. male connector (type DBC-25P).
- Length of both cords: 1500 mm.

CR80

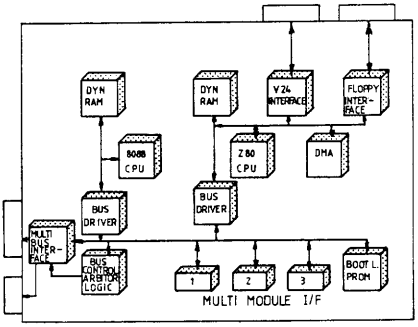
DATASHEET FOR:

MP^2 CR 2000-/0XX--/00

SET UP REFERENCE



BLOCK DIAGRAM



CHARACTERISTICS:

- Several versions available, please refer to CR80 Price List.
- Multi Purpose Multi Processor Board MP^2
- Single card dual processor micro computer system.
- Mechanically and electrically compatible with Intel's SBC boards.
- Standard Intel Multibus Interface (IEEE 796) making the board expandable with standard Multibus boards.
- On board connectors and Multimodule I/F for expansion with up to 3 standard Intel ISBX multimodule or CR Custom interface modules.
- On board RAM 128K byte, divided as 64K byte local RAM on each of the two processors.
- PROM area, standard 4 or 8K byte may be extended with on-board standard Intel or CR Custom PROM extension card.
- On-board Floppy Disc interfaces for up to four floppy disc drives.
- Two on-board serial interfaces.
- The 8088 CPU can be equipped with a arithmetic CO-processor.
- The 8088 CPU can directly address up to 1M byte memory.
- The Z80 CPU have implemented a map function, making the CPU able to address up to 1M byte memory.
- Power Consumption (max. implementation version)
  +5: 4.0A +/- 0.5A, $\pm$12V.

* MULTIBUS, MULTIMODULE trademark of INTEL CORP.

PROGRAMMERS REFERENCE:

S1 is a 4 bit switch readable by both processors from register 3.

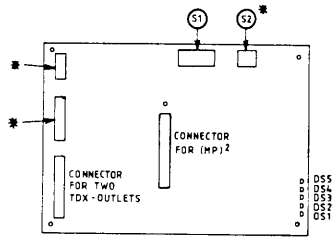
AD0	AD1	AD2	AD3
S1	S2	S3	S4

closed "0"
open "1"

DATASHEET FOR:

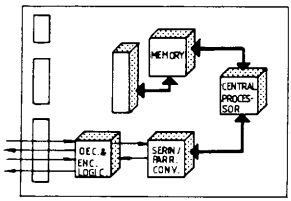
X-NET (MP)^2 ADAPTER CR 2525-/001--/00

SET UP REFERENCE



DS1: TEST/TDX-CHANNEL SELECT LED
DS2: FUSE CONTROL LED FOR SUPPLYING TDX-OUTLET #1
DS3: " " " " " " #2
DS4: POWER LED FOR INTERNAL GENERATED "+12V"
DS5: " " " " " "-12V"

BLOCK DIAGRAM



CHARACTERISTICS:

- The "X-net (MP)^2 Adapter" interfaces one or optionally dual TDX-Busses to the (MP)^2-module (CR2000) when positioned in the multimodule plug on the board.
- * Hardware design is fully prepared for connecting one serial communication channel to the TDX-Busses if supplied with 5 Volts. (See XTA CR2550-/000/--/00).
- Power Consumption: Driving two TDX-outlets

  +5V: 2,0A +/- 0.5A

- Physical Dimension:

  (160x100x22.5)mm.

Multimodule, trademark of INTEL CORP.

PROGRAMMERS REFERENCE:

Switch Orientation and Definition:

S1: | LSB | | | | | | | MSB |

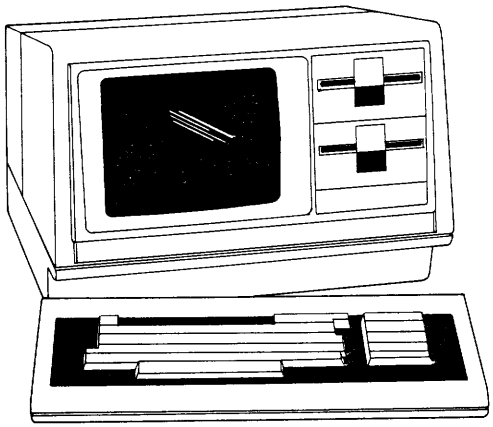
Pin1

TDX DEVICE NO.

Switch on 0
Switch off 1

S2: not used

All communication between the "X-Net (MP)^2 Adaptor" and the (MP)^2 takes place in the on-board 1K RAM of which 1/4K is shared. The RAM is divided into eight buffers of 32 byte which is used for interchanging data, status and control informations.



CHARACTERISTICS:

- CR8 is a compact desk top Integrated Work Station, which includes a versatile intelligent terminal with advanced video features, a built-in card cage which holds up to 6 MULTIBUS boards, a high capacity switching power supply and as an option two 5-1/4 inch mini floppy disk drives, or one floppy and one mini-Winchester rigid drive.

- The video terminal has a 12 inch CRT display, a number of front panel CPU control switches and LED's, plus a detachable 82 keyboard. The terminal's RS232 interface communicates with the MP^2 cards at speeds from 50 to 9600 bps.

- The versatile display is 80 characters by 25 rows, with split screen (effectively provides two "virtual" terminals) and separate scrolling regions. The systems initializes to 24 rows for screen 0 and 1 row for screen 1. Video attributes include normal and faint intensity, blink, underline, insert/delete character or line, smooth scrolling and a number of erase commands.

- The optional MP^2 (Multipurpose Multiprocesser) CR2000 which performs as the Work Stations central processor has several important features: Multibus interface, dual CPU's, optional interface to X-net (local network for terminals, process Control equipment etc.) etc. Formore detailed information please refer to datasheet for CR2000 MP^2.

- The Work Station has additional card slots available for expansions and optional cards as Disk Controller, Memory, Peripherals interfaces etc.

' MULTIBUS is trademark of Intel Corporation.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Dimensions:

Height:	370 mm
Width:	520 mm
Depth:	640 mm (w/keyboard)
Weight:	22.7 kg

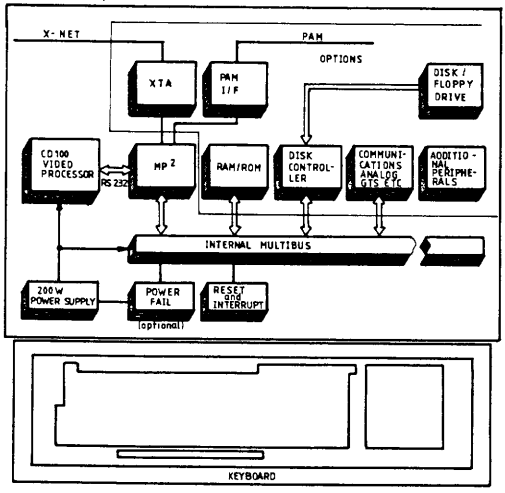
- Mains Power:

220V, 50Hz (110V, 60Hz optional)
460 W

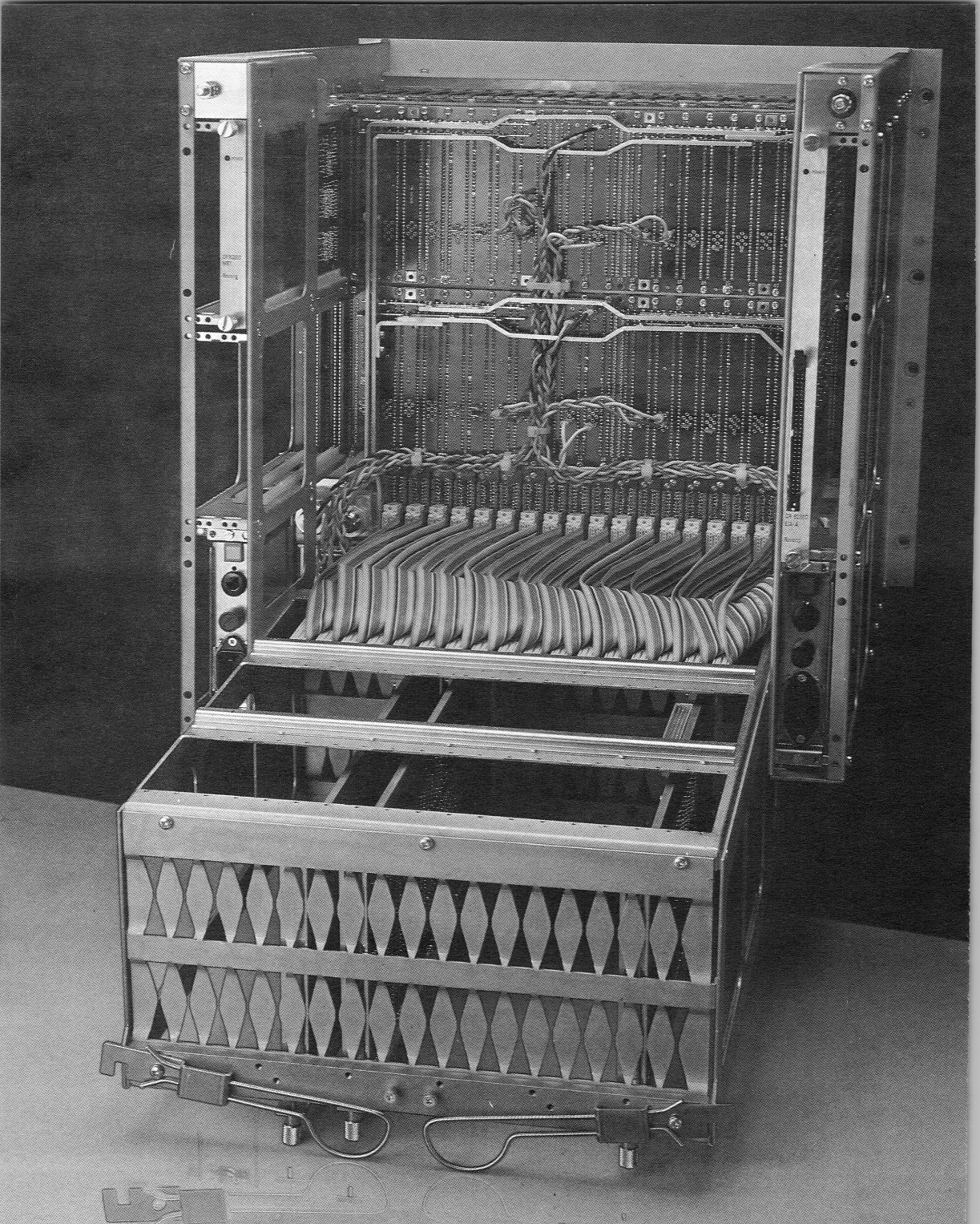
- Multibus DC Power:

+5V:	20A
+/-12V:	4A
-5V:	4A

CR 8 with optional MP² board :



he CR80 MINI (Model 801) standard computer system for commercial EDP
pplications.

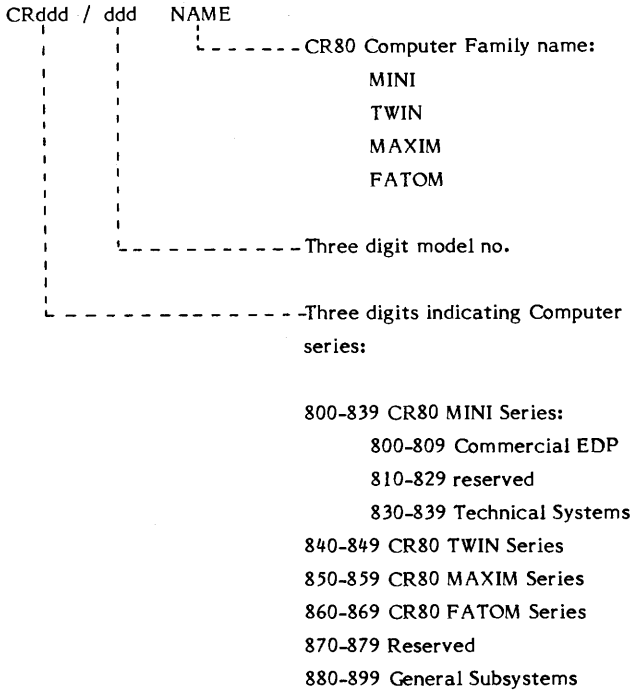


The CR80 M-Crate seen from the rear with the Adapter crate pivoted down fo access to internal buses and power distribution. Also shown are the flatcable connecting Front Modules with their associated Adapter Modules.

11. CR80 MINI,TWIN,MAXIM AND FATOM STANDARD COMPUTERS

This section contains the detailed datasheets on the CR80 Standard Computer models.

11.1 Nomenclature of CR80 Standard Computers



11.2 CR80 MINI Series

Type	Description	Ref	Page no.
CR801/331	Minicrate based commercial EDP mini with 128K RAM and supporting multidropped terminals on 4 lines and up to 4x80 Mb disk, 2 line printers, and operators console. (excl. Syst. S/W)	801.0	11-7
CR801/361	Minicrate based commercial EDP mini with 128K RAM and supporting multidropped terminals on 4 lines and up to 4x80 Mb disk, 2 line printers, Mag.tape drive, and operators console. (excl. Syst. S/W)	801.1	11-7
CR801/631	Minicrate based commercial EDP mini with 224K RAM and supporting multidropped terminals on 4 lines and up to 4x80 Mb disk, 2 line printers, and operators console. (excl. Syst. S/W)	801.2	11-7
CR801/661	Minicrate based commercial EDP mini with 224K RAM and supporting multidropped terminals on 4 lines and up to 4x80 Mb disk, 2 line printers, Mag.tape drive, and operators console. (excl. Syst. S/W)	801.3	11-7
CR831/XXX	Minicrate based Technical or Communications oriented mini with up to 224K RAM. Founded on the CR801 models, the use of the remaining positions in the MINIcrate are free to the user to specify from the range of CR80 standard modules (non-mapped).	831.0	11-7

Type	Description	Ref	Page no.
CR835/001 MINI	Single-rack computer constitued of Processor Unit crate and Channel Unit crate CPU-SCM, 128K RAM and accessories included	835.0	11-10
CR836/001 MINI	Single-rack computer constitued of Processor Unit crate including space for peripheral modules. CPU-SCM, 128K RAM and accessories included.	836.0	11-14

11.3 CR80 TWIN Series

Type	Description	Ref	Page no.
CR840/001 TWIN	Two-rack dualized computer with each rack containing a Processor Unit Crate and a dual bus Channel Unit Crate. Two CPU-SCM's, 2x128K RAM and accessories included.	840.0	11-17
CR841/001 TWIN	Single-rack dualized computer including two Processor Units in one crate anda dual bus Channel Unit Crate in another. MAP, CPU-Cache, 256K RAM, and accessories included.	841.0	11-22

11.4 CR80 MAXIM Series

Type	Description	Ref	Page no.
CR850/001 MAXIM	Single-Rack virtual memory computer including dual bus Processor Unit Crate and a Channel Unit crate in another. MAP, CPU-Cache, 256K RAM, and accessories included.	850.0	11-26
CR851/001 MAXIM	Single-Rack virtual memory computer including dual bus Processor Unit Crate and a Channel Unit in a single crate. MAP, CPU-Cache, 256K RAM, and accessories included.	851.0	11-30

11.5 CR80 FATOM Series

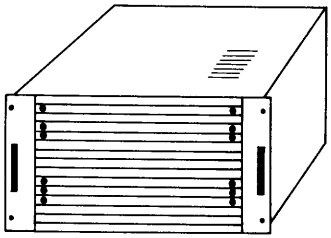
Type	Description	Ref	Page no.
CR860/0XX FATOM	Fault tolerant virtual memory computer constituted of from 2 to 16 racks each containing a dual bus'ed Processor Unit crate and a dual bus'ed Channel Unit Crate. Each Processor Unit includes MAP, CPU-Cache, I/F to Supra Bus, and 256K RAM. XX denotes the number of racks.	860.0	11-34
CR861/00X FATOM	Fault tolerant virtual memory computer constituted of from 1 to 8 racks each containing two dual bus'ed Processor Units in a single crate and a dual bus'ed Channel Unit crate. Each Processor Unit includes MAP, CPU-Cache, I/F to Supra Bus, and 256K RAM. X denotes the number of racks.	861.0	11-39

11.6 CR80 General Subsystems

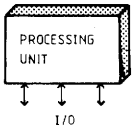
Type	Description	Ref	Page no.
CR880/OXY	Extension element with one or two channel units contained in a single rack. X = 3: Extension to MINI X = 4: Extension to TWIN X = 5: Extension to MAXIM X = 6: Extension to FATOM Y = 1: One channel Unit Y = 2: Two channel Units	880.0	11-44
CR890/0XX	Maintenance and Configuration Subsystem consisting of WPU, WCA, Configuration Bus, and up to 32 CCA'S. XX denotes the number of CCA'S, i.e. the number of Processor and Channel units under configuration control.	890.0	11-48
CR890/1XX	Maintenance and Configuration Subsystem consisting of WPU, WCA, Configuration Bus, and up to 32 CCA'S. The WPU and the WCA are house in a Watchdog crate. XX denotes the number of CCA'S, i.e. the number of Processor and Channel units under configuration control.	890.1	11-48
CR891/0XX	Maintenance and Configuration Subsystem consisting of WPU, WCA, WPC, Watchdog Panel, Configuration Bus and up to 32 CCA'S. XX denotes the number of CCA'S,i.e. the number of Processor and channel Units under configuration control.	891.0	11-49

Type	Description	Ref	Page no.
CR891/0XX	Maintenance and Configuration Subsystem consisting of WPU, WCA, WPC, Watchdog Panel, Configuration Bus and up to 32 CCA'S. WPU, WCA, WPC, and the Watchdog Panel are housed in a Watchdog crate. XX denotes the number of CCA'S, i.e. the number of Processor and channel Units under configuration control.	891.0	11-49

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- The CR801-MINI is a compact, general purpose computer, which is ideal for a range of commercial applications with limited input/output requirements.
- The CR831-MINI is the technical or communications oriented versions of the CR801, housing more customer specified modules than given in the below standard system nomenclature for the CR801.
- The system is housed in a self-contained Minicrate with integrated Power Supply and Fan Unit.
- The system consists of a Processor Unit with space available for peripheral controllers.
- System nomenclature

CR801/d_1 d_2 d_3 SS/XX

d_1: (memory)	3	128K RAM
	4	160K RAM
	5	192K RAM
	6	224K RAM
d_2: (peripheral options)	3	DISC
	6	DISC + TAPE
d_3: (communication interface)	1	1 LTU

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Module survey

Central Processing Unit and System Controller (CPU/SCM + CSA)

is a general purpose processor with 16 bits word length and a standard instruction set of basic arithmetics, logics, transfer, and special instructions including bit-, byte-, word- and multiple word manipulations. The adapter (CSA) provides a terminal interface and two line printer interfaces.

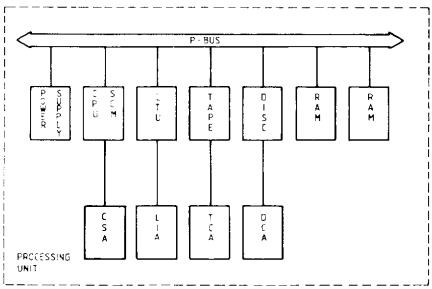
Working Storage (RAM) The memory area is constituted by memory modules of up to 128K words and memory resident on the peripheral controllers.

DISC CTRL (+DCA) Controller capable of handling 1 to 4 disc drives.

TAPE CTRL (+TCA) Controller capable of handling 1 to 8 magnetic tape transports.

Communications Interface (LTU+LIA) Protocol processor with interface to 4 serial channels.

Module Diagram



- Dimensions:

Height:	270 mm	(6U)
Width:	485 mm	(19")
Deepth:	615 mm	
Weight:	37 kg	(Maximum configuration)

- Mains Power:

Voltage 220V +/- 20% or 110V +/- 20%
Frequency 45HZ - 440HZ
Power Consumption: 450W

- Environmental Conditions:

Temperature	15°C to 32°C
Humidity	20% RH - 80% RH (non condensing).

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

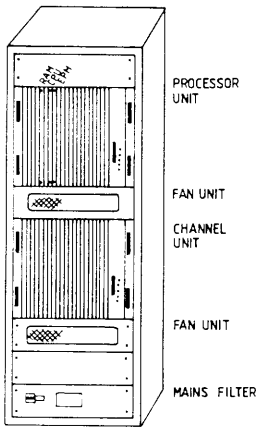
- Equipment List:

The items constituting the CR801 MINI are listed below:

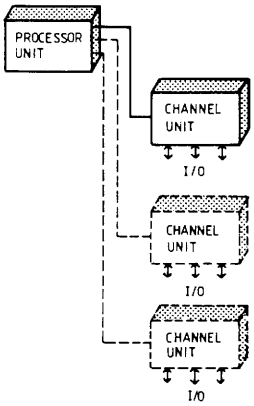
			QTY according to d_1 d_2 d_3							
Ref .	Type No.	Description	331	361	431	461	531	561	631	661
115.0	CR8115M/012P-/00	Minicrate	1	1	1	1	1	1	1	1
201.0	CR8201M/015--/00	Power Cable	1	1	1	1	1	1	1	1
2.0	CR8002M/010P-/00	CPU-SCM	1	1	1	1	1	1	1	1
70.0	CR8070M/010--/00	CSA	1	1	1	1	1	1	1	1
16.4	CR8016M/128P-/00	128K RAM	1	1	2	2	2	2	2	2
44.2	CR8044M/011A-/00	DISC CTRL	1	1	1	1	1	1	1	1
84.0	CR8084M/010A-/00	DSA	1	1	1	1	1	1	1	1
45.1	CR8045M/010A-/00	TAPE CTRL	0	1	0	1	0	1	0	1
85.0	CR8085M/010--/00	TCA	0	1	0	1	0	1	0	1
66.0	CR8066M/030A-/00	LTU	1	1	1	1	1	1	1	1
82.0	CR8082M/010--/00	LIA-N	1	1	1	1	1	1	1	1

CR835/001 MINI COMPUTER

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- The CR835 MINI Computer allows the system designer to specify single computer systems with a processing power from 0.7 mega instructions in the basic 1 CPU system to 2 mega instructions by adding CPU modules. The CR835 MINI's working storage can be expanded from 128K words to 256K words and the Input/Output connectivity can be as high as 400 peripheral controllers.

- The system consists of a Processor Unit and a Channel Unit both contained in one rack:

Processing Unit

Central Processing Unit and System Controller (CPU-SCM) is a general purpose processor with 16 bits word length and a standard instruction set of basic arithmetics, logics, transfer, and special instructions including bit-, byte-, word- and multiple word manipulations. The controller section makes it possible to install more CPU's for obtaining more processing capacity.

128K Working Storage (128K RAM) included in the basic Processing Unit can be expanded by adding extra 128K words to the configuration.

Power Supply One module is included, but an auxiliary module working in parallel can be installed, if expansion of the Processing Unit makes it necessary.

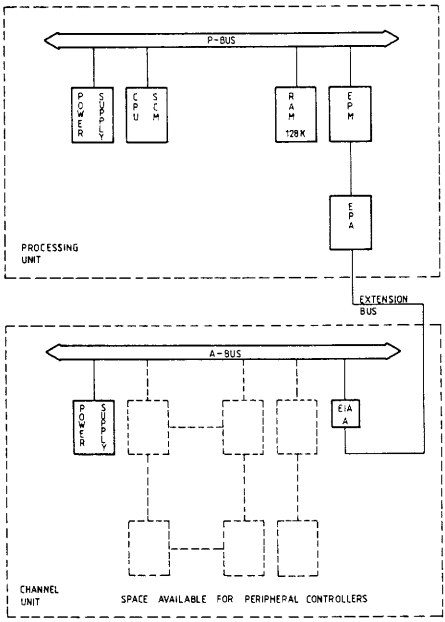
Channel Unit

The Channel Unit is delivered with Power Supply, Transfer Bus and Extension Bus Interface (EIA-A). No peripheral controllers are included but the unit is prepared for installation of a great variety of single bus peripheral controllers from the CR80 product line as DISC CTRL, TAPE CTRL, LINE PRINTER CTRL, COMMUNICATION LINE CTRL, etc.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

Module Diagram



- Dimensions (basic configuration)

Height:	1800 mm
Width:	600 mm
Deepth:	730 mm
Weight:	225 kg

- Environmental Conditions:

The CR835 MINI is designed to operate within the following limits:

Ambient temperature	15°C to 32°C
Humidity	20% RH to 80% RH (non-condensing)

- Mains Power:

Voltage 220V	+20% -10%	(optional 110V)
Frequency 50HZ +/- 5HZ		(optional 60HZ)
Power Consumption: 900W		for the basic CR840 element

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Processor Unit Expandability:

The basic Processing Unit contains module positions for expansion by adding modules (CPU, RAM etc.) to the configuration. The module positions are specified in the CR8125M/125P-/00 PU-crate datasheet.

Power available from the basic Power Supply allows for the following expansion

a) 128K words of memory
b) 1 CPU-CACHE + 128K words of memory
c) 2 CPU-CACHE + 128K words of memory

By adding one Power Supply module, further expansion is possible

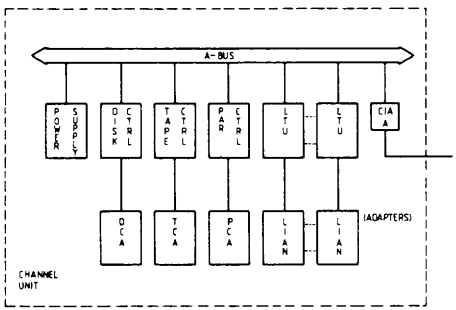
a) 3 CPU-CACHE + 128K words of memory
b) 4 CPU-CACHE + 128K words of memory

The Processing Unit can also contain I/O modules.

- Input/Output Connectivity:

The Channel Unit is equipped with cables, Power Supply, and interfaces for connecting single bus'ed peripheral controllers and adapters from the CR80 product range. When furnishing the Channel Unit, the available power (5V:55A) and the available module slots (17) have to be considered as specified in the CR8125M/325A-/00 CU-crate datasheet.

As an example is it calculated how many communication interfaces (LTU/LIA-pairs) can be installed besides a typical selection of basic peripherals:



(CONTINUED)

CHARACTERISTICS: (CONTINUED)

	POWER CONSUMPTION +5V	+12V	-12V	
1 x Disc CTRL	7.0	0.3	0.1	A
1 x DCA	1.0	0	0	A
1 x Tape CTRL	6.5	0.3	0.1	A
1 x TCA	1.0	0	0	A
1 x Parallel CTRL	6.5	0.3	0.1	A
1 x PCA	1.0	0	0	A
Subtotal	23	0.9	0.3	A
Total available	55	4	4	A
Available for LTU/LIA	32	3.1	3.7	A

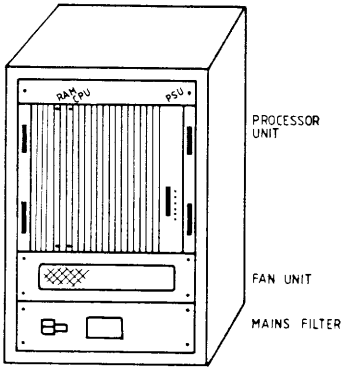
As a LTU consumes 3.5A on 5V, 9 can be installed. If the input/output connectivity of the CR835 is not sufficient, further Channel Units can be added by using the CR880 extension element.

- Equipment List:

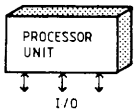
The items constituting the basic CR835 MINI computer are listed below:

Ref .	Type No.	Description	Qty
101.1	CR8101-/036--/00	36U RACK	1
125.1	CR8125M/125P-/00	PU-CRATE	1
125.3	CR8125M/325A-/00	CU-CRATE	1
105.0	CR8105M/020--/00	FAN UNIT	2
106.0	CR8106-/220--/00	MAINS FILTER	1
50.0	CR8050M/010--/00	POWER SUPPLY	2
2.0	CR8002M/010P-/00	CPU SCM	1
70.0	CR8070M/010--/00	CSA	1
16.4	CR8016M/128P-/00	RAM 128K	1
9.0	CR8009M/010P-/00	EPM	1
74.0	CR8074M/010--/00	EPA	1
88.0	CR8088M/010A-/00	EIA-A	1
212.0	CR8212M/015--/00	CABLE EXTENSION BUS	1
201.0	CR8201M/015--/00	CABLE RACK POWER	6

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- CR836 MINI computer is a compact system with a capacity sufficient to cover all requirements for small computer systems.

- The basic CR836 MINI is delivered equipped with one Central Processor Unit (CPU), but the system can be upgraded to a multiprocessor by adding more CPU modules. The working storage can be increased from 128K Words to 256K Words, and upto 10 peripheral Controller modules can be installed.

- The system consists of a Processor Unit with space available for peripheral controllers. Three basic modules are included:

  Central Processing Unit and System Controller (CPU-SCM) is a general purpose processor with 16 bits word length and a standard instruction set of basic arithmetics, logics, transfer, and special instructions including bit-, byte-, word-, and multiple word manipulations. The controller section makes it possible to install more CPU's for obtaining more processing capacity.

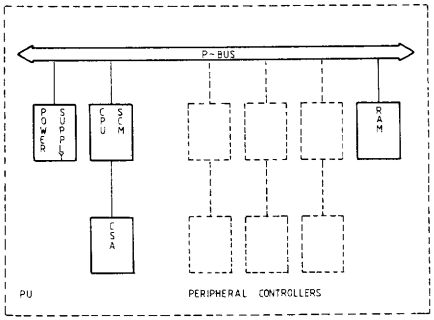
  128K Working Storage (128K RAM) included in the basic Processor Unit can be expanded by adding a module of 128K to the configuration.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

Power Supply One unit is included but an auxiliary unit working in parallel can be installed if the power requirements are increased.

Module diagram



- Dimensions (basic configuration)

Height:	1000 mm
Width:	600 mm
Deepth:	730 mm
Weight:	120 kg

- Environmental Conditions:

The CR836 MINI is designed to operate within the following limits:

Ambient temperature	15°C to 32°C
Humidity	20% RH to 80% RH (non-condensing)

- Mains Power:

Voltage 220V	+20% -10%	(optional 110V)
Frequency 50HZ +/- 5HZ		(optional 60HZ)
Power Consumption: 500W		(basic configuration)

- Additional Modules:

The available power and the available module slots must be considered when furnishing the Processor Unit. The CR8125M/125P-/00 PU-Crate datasheet specifies the slots and the following examples illustrate the power calculation:

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

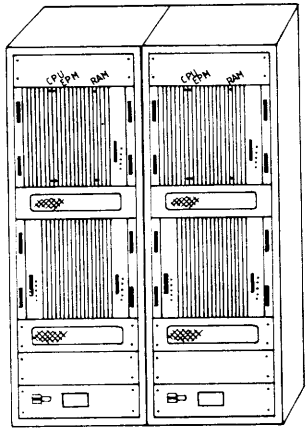
Additional modules:		AVAILABLE POWER +5V 44	+12V 3	-12V 3.5	A
a)	128K RAM	3.5	1.0	0.5	A
	DISC CTRL	7.0	0.3	0.1	A
	DCA	1.0	0	0	A
	TAPE CTRL	6.5	0.3	0.1	A
	TCA	1.0	0	0	A
	PAR CTRL	6.5	0.3	0	A
	PCA	1.0	0	0	A
	3xLTU/L1A	10.5	0.9	0.3	A
	Total	37	2.8	1.0	A
b)	CPU-CACHE	18	0	0	A
	128K RAM	3.5	1.0	0.5	A
	TAPE CTRL	6.5	0.3	0.1	A
	TCA	1.0	0	0	A
	4xLTU/L1A	14	1.2	0.4	A
	Total	43	2.5	1.0	A

- Equipment List

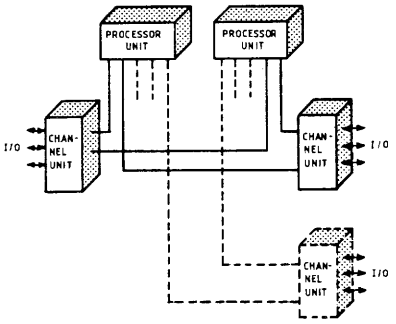
The items consituting the basic CR836 MINI computer are listed below:

Ref .	Type No.	Description	Qty
102.0	CR8102-/000--/00	18U RACK	1
125.1	CR8125M/125P-/00	PU-CRATE	1
105.0	CR8105M/020--/00	FAN UNIT	1
106.0	CR8106-/220--/00	MAINS FILTER	1
50.0	CR8050M/010--/00	POWER SUPPLY	1
2.0	CR8002M/010P-/00	CPU SCM	1
70.0	CR8070M/010--/00	CSA	1
16.4	CR8016M/128P-/00	128K RAM	1
201.0	CR8201M/015--/00	CABLE POWER	3

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- CR840 TWIN computer allows the system designer to specify standard dualized computers which can be expanded by increasing the number of Central Processor Units (CPU) up to three, the amount of working storage and the number of peripheral Controllers to meet systems demands for processing power and Input/Output connectivity.

- For monitoring, control and maintenance of the dualized CR840 TWIN computer, please refer to datasheets for Supervision and Configuration Subsystem CR890 and CR891.

- CR840 TWIN contains two Processor Units and two Channel Units housed in two 19" standard racks, one Processor Unit and one Channel Unit in each.

- The performance characteristics of each rack (Processor Unit and Channel Unit) are defined below and shown in module diagram overleaf:

  Processor Unit

  Central Processing Unit and System Controller (CPU-SCM) is a general purpose processor with 16 bits word length and a standard instruction set of basic arithmetics, logics, transfer, and special instructions including bit-, byte-, word-, and multiple word manipulations. The controller section makes it possible to install more CPU's for obtaining more processing capacity.

  128K Working Storage (128K RAM) included in the basic Processor Unit can be expanded by adding a module of 128K to the configuration.

  Extension Bus Interface (EPM+EPA) The Extension Bus is a twisted pair flatcable which extends the Processor Bus to a Channel Unit. More Channel Units can be connected by adding one EPA for each Channel Unit. (One EPM can handle 8 EPA's).

  Channel Unit

(CONTINUED)

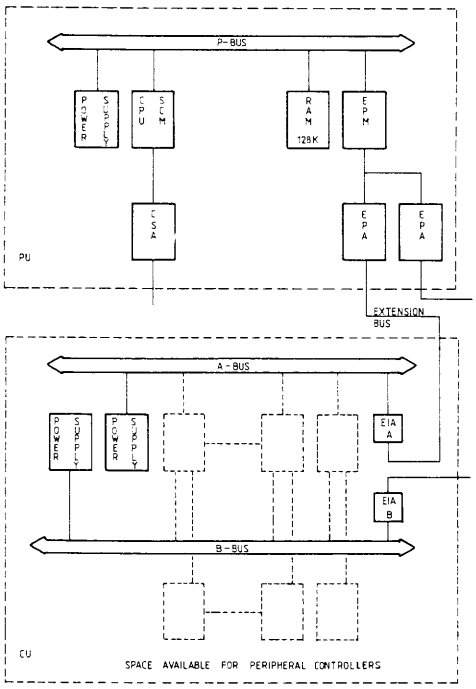
CHARACTERISTICS: (CONTINUED)

Extension Bus Interface (EIA-A & EIA-B) connects the Channel Unit to two Extension Buses.

Power Supply Two units are included and the load is shared by an OR-circuit on each power-consuming module.

Peripheral Controllers are not included but the unit is prepared for installation of a great varity of dual bus'ed peripheral modules from the CR80 product line as DISK CTRL, TAPE CTRL, LINE PRINTER CTRL, Communication Line CTRL etc.

Module Diagram (one basic rack)



(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Dimensions of CR840 TWIN basic computer:

Height:	1800 mm
Width:	1200 mm
Deepth:	730 mm
Weight:	430 kg

- Environmental Conditions:

The CR840 TWIN is designed to operate within the following limits:

Ambient temperature	15°C to 32°C
Humidity	20% RH to 80% RH (non-condensing)

For more detailes please ref. CR80 Handbook.

- Mains Power:

Voltage 220V	+20% -10%	(optional 110V)
Frequency 50HZ +/- 5HZ		(optional 60HZ)
Power Consumption: 2000W		for the basic CR840

- Processor Unit Expandability:

The basic Processing Unit contains module positions for extension by adding modules (CPU, RAM and other) to the configuration. The module positions are specified in CR8125M/125P-/00 PU-Crate datasheet.

Power available from the basic Power Supply allows for the following expansions:

128K words of memory + 2 CPU

When additional modules are required e.g. Mainframe interfaces, an extra Power Supply operating in parallel with the basic module can be installed.

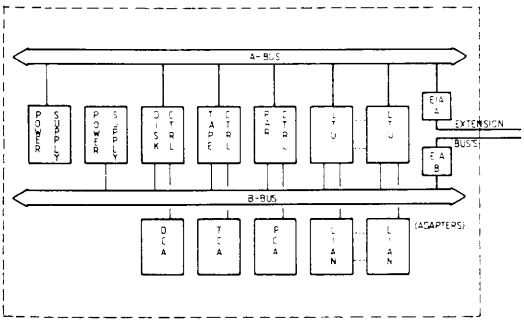
(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Input/Output Connectivity

The Channel Unit is equipped with cables, Power Supply interfaces for connecting dual bus'ed peripheral controllers and adapters from the CR80 product range. When furnishing the Channel Unit the available power (5V: 55A) and the available module slots (17) have to be considered as specified in the CR8125M/425AB/00 CU-Crate datasheet.

As an example the power consumption of the configuration below is calculated.



	POWER CONSUMPTION			
	+5V	+12V	-12V	
1 x Disc CTRL	7.0	0.3	0.1	A
1 x DCA	1.0	0	0	A
1 x Tape CTRL	6.5	0.3	0.1	A
1 x TCA	1.0	0	0	A
1 x Parallel CTRL	6.5	0.3	0.1	A
1 x PCA	1.0	0	0	A
Subtotal	23	0.9	0.3	A
Total available	50	4	4	A
Available for LTU/LIA-N	27	3.1	3.7	A
Number of LTU/LIA-N	27/3.5(+5V)		=	7 LTU/LIA-N

If the Input/Output connectivity of CR840 is not sufficient for the system, then the extension element CR880 can be added to the configuration. Up to 3 x CR880 with 6 CU's can be connected by adding EPA's to the CR840 PU's.

(CONTINUED)

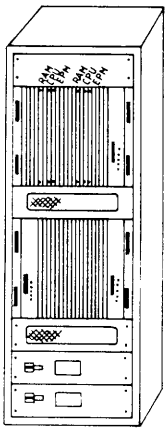
CHARACTERISTICS: (CONTINUED)

- Equipment List

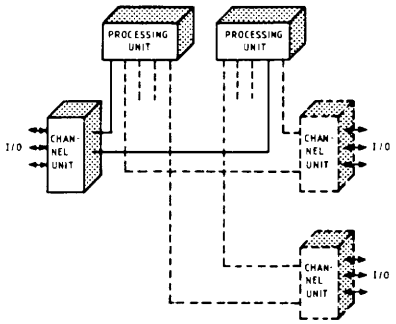
The items constituting the CR840 TWIN basic computer are defined below:

Ref .	Type No.	Description	Qty
101.1	CR8101-/036--/00	RACK	2
125.1	CR8125M/125P-/00	PU-CRATE	2
125.2	CR8125M/425AB/00	CU-CRATE	2
105.0	CR8105M/020--/00	FAN UNIT	4
106.0	CR8106-/220--/00	MAINS FILTER	2
50.0	CR8050M/010--/00	POWER SUPPLY	6
107.0	CR8107-/010--/00	POWER DISTRIBUTION PANEL	4
2.0	CR8002M/010P-/00	CPU SCM	2
9.0	CR8009M/010P-/00	EPM	2
74.0	CR8074M/010--/00	EPA	4
16.4	CR8016M/128P-/00	RAM 128K	2
70.0	CR8070M/010--/00	CSA	2
212.0	CR8212M/015--/00	CABLE EXTENSION BUS	4
201.0	CR8201M/015--/00	CABLE RACK POWER	16
88.0	CR8088M/010A-/00	EIA-A	2
88.1	CR8088M/010-B/00	EIA-B	2
55.1	CR8055M/020--/00	MBT	8

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- CR841 TWIN computer allows the system designer to specify standard dualized computers which can be expanded by increasing the number of Central Processor Units (CPU) upto three, the amount of working storage and the number of Input/Output Controllers to meet systems demands for processing power and Input/Output connectivity.

- For monitoring, control and maintenance of the dualized CR840 TWIN computer please refer to datasheets for Supervision and Configuration Subsystem CR890 and CR891.

- CR841 TWIN contains two Processing Units and one Channel Unit housed in a standard 19" rack.

- The system consists of a double Processing Unit and a Channel Unit with characteristics as defined below and shown in module diagram overleaf:

Processing Unit:

The two Processing Units are identical and contains the modules defined below:

Central Processing Unit and System Controller (CPU-SCM) is a general purpose processor with 16 bits word length and a standard instruction set of basic arithmetics, logics, transfer, and special instructions including bit-, byte-, word-, and multiple word manipulations. The controller section makes it possible to install more CPU's for obtaining more processing capacity.

128K Working Storage (128K RAM) included in the basic Processing Unit can be expanded by adding extra 128K to the configuration.

Extension Bus Interface (EPM+EPA) The Extension Bus is a twisted pair flatcable which extends the Processor Bus to a Channel Unit. More Channel Units can be connected by adding one EPA for each Channel Unit. (One EPM can handle 8 EPA's).

Power Supply with sufficient power for expansion of the unit.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

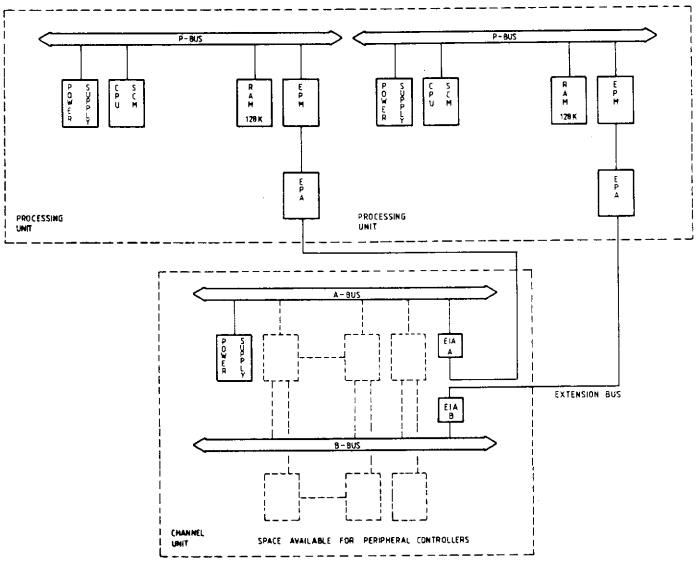
Channel Unit:

Extension Bus interface (EIA-A & EIA-B) connects the Channel Unit to two Extension Buses.

Power Supply Two units are included and the load is shared by an OR-circuit on each power consuming module.

Input/Output Controllers are not included but the unit is prepared for installation of a great varity of dual bus'ed controllers from the CR80 product line as DISC CTRL, TAPE CTRL, LINEPRINTER CTRL, Communication line CTRL etc.

Module Diagram (One basic rack)



- Dimensions of CR841 TWIN basic computer:

Height:	1800 mm
Width:	600 mm
Deepth:	730 mm
Weight:	250 kg

- Environmental Conditions:

The CR840 TWIN is designed to operate within the following limits:

Ambient temperature	15°C to 32°C
Humidity	20% RH to 80% RH (non-condensing)

For more detailes please ref. CR80 Handbook.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Mains Power:

Voltage 220V	+20% -10%	(optional 110V)
Frequency 50HZ +/- 5HZ		(optional 60HZ)
Power Consumption: 1200W		for the basic CR841 element

- Processor Unit Expandability:

The basic Processing Unit contains module positions for expansion by adding modules (CPU, RAM and other) to the configuration. The module positions are specified in CR8112M/212PC/00 PU-Crate datasheet.

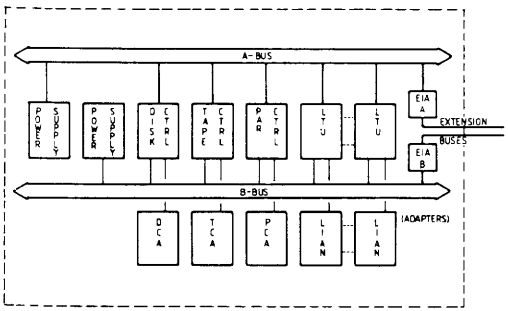
Power available from the basic Power Supply allows for the following expansions:

128K words of memory + 2 CPU

- Input/Output Connectivity

The Channel Unit is equipped with cables, Power Supply interfaces for connecting dual bus'ed peripheral controllers and adapters from the CR80 product range. When furnishing the Channel Unit the available power (5V: 55A) and the available module slots (17) have to be considered as specified in the CR8125M/425AB/00 CU-Crate datasheet.

As an example the power consumption of the configuration below is calculated.



(CONTINUED)

CHARACTERISTICS: (CONTINUED)

	POWER CONSUMPTION +5V	+12V	-12V	
1 x Disc CTRL	7.0	0.3	0.1	
1 x DCA	1.0	0	0	
1 x Tape CTRL	6.5	0.3	0.1	
1 x TCA	1.0	0	0	
1 x Parallel CTRL	6.5	0.3	0.1	
1 x PCA	1.0	0	0	
Subtotal	23	0.9	0.3	
Total available	50	4	4	
Available for LTU/LIA	27	3.1	3.7	
Number of LTU/LIA-N	27/3.5(+5V)	=		7 LTU/LIA

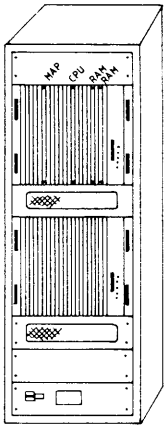
If the Input/Output connectivity of CR841 isn't sufficient for the system, then the extension element CR880 can be added to the configuration. Up to 3 x CR880 with 6 CU's can be connected by adding corresponding EPA's to the CR841 PU's.

- Equipment List

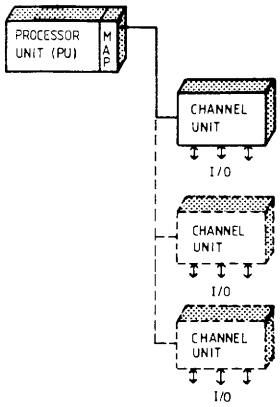
The items constituting the CR841 TWIN basic computer are defined below:

Ref .	Type No.	Description	Qty
101.1	CR8101-/036--/00	RACK	1
112.0	CR8112M/212PC/00	PU-CRATE	1
125.2	CR8125M/425AB/00	CU-CRATE	1
105.0	CR8105M/020--/00	FAN UNIT	2
106.0	CR8106-/220--/00	MAINS FILTER	2
50.0	CR8050M/010--/00	POWER SUPPLY	4
107.0	CR8107-/010--/00	POWER DISTRIBUTION PANEL	2
2.0	CR8002M/010P-/00	CPU SCM	2
9.0	CR8009M/010P-/00	EPM	2
74.0	CR8074M/010--/00	EPA	2
16.4	CR8016M/128P-/00	RAM 128K	2
70.0	CR8070M/010--/00	CSA	2
212.0	CR8212M/015--/00	CABLE EXTENSION BUS	2
201.0	CR8201M/015--/00	CABLE RACK POWER	8
88.0	CR8088M/010A-/00	EIA-A	1
88.1	CR8088M/010-B/00	EIA-B	1
55.1	CR8055M/020--/00	MBT	6

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- The CR850 MAXIM Computer allows the system designer to specify a standard computer which can be expanded from one Central Processor Unit (CPU) up to five CPU's to meet a variery of single systems demands for throughput and performance.

- The CR850 MAXIM is equipped with a Processor Unit and a Channel Unit as defined below and shown in the module diagram overleaf.

Processor Unit

Memory Mapping Module (MAP) provides addressing of up to 16M words of virtual memory, demand paging and a protection mechanism to prevent access from unauthorized users.

Central Processor Unit (CPU) is a general purpose processor with 16 bits word length and a standard instruction set of basic arithmetics, logic, transfer and special instruction including bit, byte, word and multiple word manipulations. The CACHE memory included on the module minimizes the bus load and thereby ensures that the number of CPUs can be increased.

256K words Working Storage (256K RAM) included in the basic Processor Unit (PU) can be expanded by adding modules of 128K words to the configuration.

Data Channel Interface (MIA) interfaces the Processor Unit via the flatcable Data Channel to the Channel Unit. A maximum of 15 Channel Units can be connected to the Data Channel.

Power Supply one module is included but if required due to expansion of the Processor Unit, an additional Power Supply can be installed.

Channel Unit

The Channel Unit is delivered with power Supply, Transfer Bus and Data Channel Interface (CIA-A). No peripheral controllers are included, but the unit is prepared for installation of a great variety of

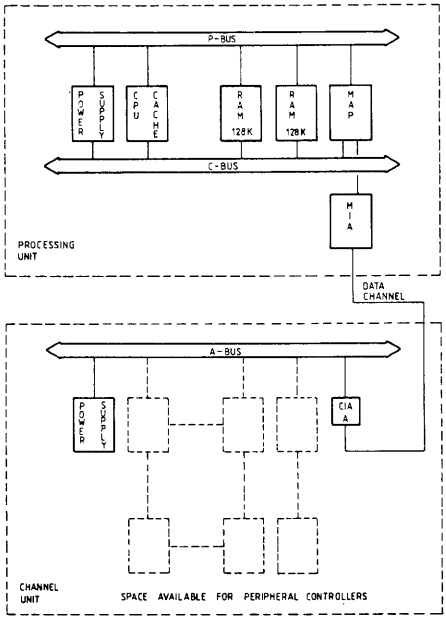
(CONTINUED)

CHARACTERISTICS: (CONTINUED)

single bus peripheral controllers from the CR80 product line as DISC CTRL, TAPE CTRL, LINE PRINTER CTRL, Communication Line CTRL, etc.

Module Diagram

The diagram shows CR850's module configuration.



- Dimensions:

The dimensions of the CR850 MAXIM basic configuration are specified below:

Height:	1800 mm
Width:	600 mm
Deepth:	730 mm
Weight:	225 kg

- Environmental Conditions:

The CR851 MAXIM is designed to operate within the following limits:

Ambient temperature	15°C to 32°C
Humidity	20% RH to 80% RH (non-condensing)

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Mains Power:

Voltage 220V	+20% -10%	(optional 110V)
Frequency 50HZ	+/- 5HZ	(optional 60HZ)
Consumption:	500 W	

- Processor Unit Expandability:

The basic Processing Unit contains module positions for expansion by adding modules (CPU, RAM and other) to the configuration. The module positions are specified in the CR8125M/225PC/00 PU-Crate datasheet.

Power available from the basic Power Supply allows for the following expansions:

128K words of memory.

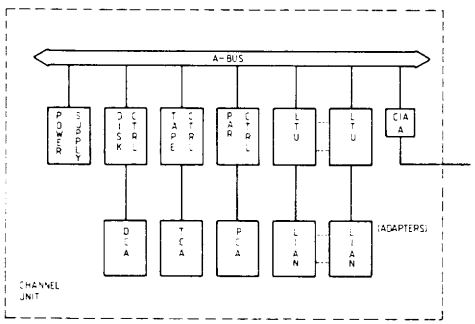
By adding one Power Supply module following expansions are possible.

a. 768K words of memory
b. 768K words of memory +1 CPU
c. 768K words of memory +2 CPU
d. 256K words of memory +3 CPU

- Input/Output Connectivity

The Channel Unit is equipped with cables, Power Supply interfaces for connecting single bus'ed peripheral controllers and adapters from the CR80 product range. When furnishing the Channel Unit the available power (5V: 55A) and the available module slots (17) have to be considered as specified in the CR8125M/425A-/00 CU-crate datasheet.

Configuration Example:



(CONTINUED)

CHARACTERISTICS: (CONTINUED)

	POWER CONSUMPTION +5V	+12V	-12V	
1 x Disc CTRL	7.0	0.3	0.1	A
1 x DCA	1.0	0	0	A
1 x Tape CTRL	6.5	0.3	0.1	A
1 x TCA	1.0	0	0	A
1 x Parallel CTRL	6.5	0.3	0.1	A
1 x PCA	1.0	0	0	A
Subtotal	23	0.9	0.3	A
Total available	55	4	4	A
Available for LTU/LIA	32	3.1	3.7	A
Number of LTU/LIA	32/3.5		=	9 LTU/LIA

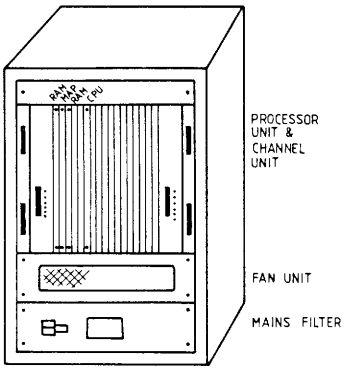
If the Input/Output connectivity of the CR860 is not sufficient, further Channel Units can be added by using the CR880 extension element.

- Equipment List.

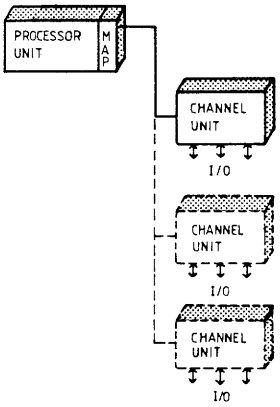
The items forming the basic CR850 MAXIM computer are listed below:

Ref .	Type No.	Description	Qty
101.1	CR8101-/036--/00	RACK	1
125.0	CR8125M/225PC/00	PU-CRATE	1
125.3	CR8125M/425A-/00	CU-CRATE	1
105.0	CR8105M/020--/00	FAN UNIT	2
106.0	CR8106-/220--/00	MAINS FILTER	1
50.0	CR8050M/010--/00	POWER SUPPLY	2
107.0	CR8107-/010--/00	POWER DISTRIBUTION PANEL	1
20.0	CR8020M/000PC/00	MAP	1
71.0	CR8071M/010--/00	MIA	1
3.1	CR8030M/040PC/00	CPU CACHE	1
16.8	CR8016M/128PC/00	RAM 128K	2
217.5	CR8211M/738--/00	DATA CHANNEL TERMINATION	1
211.1	CR8211M/015--/00	CABLE DATA CHANNEL	1
201.0	CR8201M/015--/00	CABLE RACK POWER	6
55.1	CR8055M/020--/00	MBT	5

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- The CR851 MAXIM Computer allows the system designer to specify a standard memory mapped computer for small systems applications which can be expanded from one Central Processor Unit (CPU) up to three to meet most small systems demands for throughput and performance.

- The CR851 MAXIM is equipped with a Processor Unit and a Channel Unit as defined below and shown in module diagram overleaf.

Processor Unit

Memory Mapping Module (MAP) provides addressing of up to 16M words of virtual memory demand paging and a protection mechanism to prevent access from unauthorized users.

Central Processor Unit (CPU) is a general purpose processor with 16 bits word length and a standard instruction set of basic arithmetics, logic, transfer and special instruction including bit, byte, word and multiple word manipulations. The CACHE memory included on the module minimizes the bus load and thereby increases effectivity.

256K words Working Storage (256K RAM) included in the basic Processing Unit (PU) can be expanded by adding modules of 128K words to the configuration.

Data Channel Interface (MIA) interfaces the Processor Unit via the flatcable Data Channel to the channel unit. A maximum of 15 Channel Units can be connected to the Data Channel.

Channel Unit

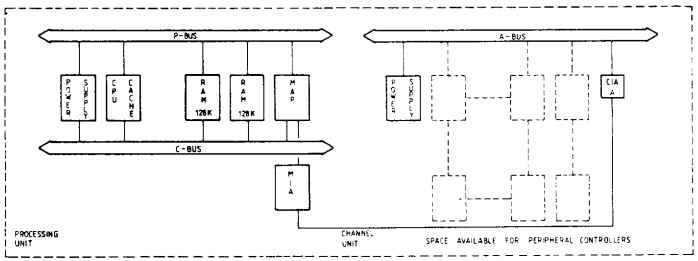
The Channel Unit is delivered with power Supply, Transfer Bus and Data Channel Interface (CIA-A). No peripheral controllers are included, but the unit is prepared for installation of a great variety of single bus peripheral from the CR80 product line as DISC CTRL, TAPE CTRL, LINE PRINTER CTRL, Communication Line CTRL, etc.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

Module Diagram

The diagram shows CR851's module configuration.



- Dimensions:

The dimensions of the CR851 MAXIM basic configuration are specified below:

Height:	1000 mm
Width:	600 mm
Deepth:	730 mm
Weight:	135 kg

- Environmental Conditions:

The CR851 MAXIM is designed to operate within the following limits:

Ambient temperature	15°C to 32°C
Humidity	20% RH to 80% RH (non-condensing)

- Mains Power:

Voltage 220V	+20% -10%	(optional 110V)
Frequency 50HZ	+/- 5HZ	(optional 60HZ)
Consumption:	500 W	

- Processor Unit Expandability:

the basic Processing Unit contains module positions for extension. The module positions are specified in CR8112M/112PA/00 PU-CRATE datasheet.

Power and slots available allows for the following expansion:

512K words of memory.

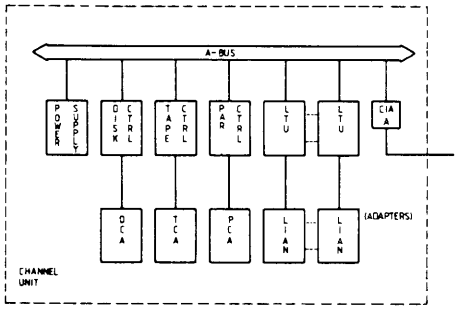
(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Input/Output Connectivity

The Channel Unit is equipped with cables, Power Supply interfaces for connecting single bus'ed peripheral controllers and adapters from the CR80 product range. When furnishing the Channel Unit the available power (5V: 55A) and the available module slots (8) have to be considered as specified in the CR8112M/112PA/00.
COMBI CRATE datasheet:

- Configuration Example



	POWER CONSUMPTION			
	+5V	+12V	-12V	
1 x Disc CTRL	7.0	0.3	0.1	A
1 x DCA	1.0	0	0	A
1 x Tape CTRL	6.5	0.3	0.1	A
1 x TCA	1.0	0	0	A
1 x Parallel CTRL	6.5	0.3	0.1	A
1 x PCA	1.0	0	0	A
Subtotal	23	0.9	0.3	A
Total available	55	4	4	A
Available for LTU/LIA	32	3.1	3.7	A

As the 5V- consumption of a LTU/LIA- pair is 3,5A, the limiting factor is the number of slots, and 5 LTU's can therefore be installed.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

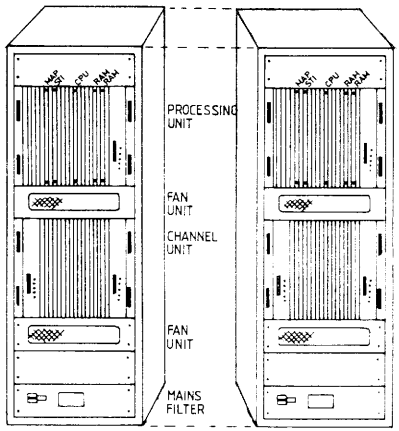
- Equipment List.

The items forming the basic CR851 MAXIM computer are listed below:

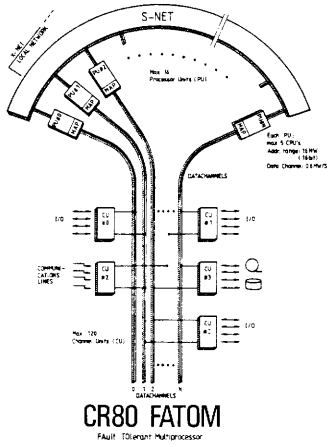
Ref .	Type No.	Description	Qty
	CR8102-/000--/00	18U Rack	1
112.1	CR8112M/112PA/00	Combi Crate	1
20.0	CR8021M/000PC700	MAP	1
3.1	CR8003M/040PC/00	CPU-CACHE	1
16.8	CR8016M/128PC/00	128K RAM	2
71.0	CR8017M/010--/00	MIA	1
50.0	CR8050M/010--/00	Power Supply	2
105.0	CR8105M/020--/00	Fan Unit	1
106.0	CR8106-/220--/00	Mains Filter	1
81.0	CR8081M/010A-/00	CIA-A	1
55.1	CR8055M/020--/00	MBT	3
211.1	CR8211M/015--/00	Cable Data Channel	1
211.5	CR8211M/738--/00	Data Channel Termination	1
201.0	CR8201M/015--/00	Rack Power Cable	4

CR860 FATOM COMPUTER

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- The CR860 FATOM Computer allows the system designer to specify standard fault tolerant computers with up to 16 Processor Units to meet virtually any demands to system throughput and performance.

- CR860 is available in following models ranging from a 2 PU (Processor Unit), 2 CU (Channel Unit) computer up to the maximum 16 PU, 16 CU, with each PU, CU forming the basic element housed in a standard 19" rack:

  CR860/002- CR860/016; last three digits indicating number of elements (PU, CU).

- All elements in the CR860 are equipped identically with a Processing Unit and a Channel Unit as defined below and shown in module diagram overleaf.

  Processor Unit

  Memory Mapping Module (MAP) provides addressing of up to 16M words of virtual memory, demand paging and a protection mechanism to prevent access from unauthorized users.

  Central Processor Unit (CPU) is a general purpose processor with 16 bits word length and a standard instruction set of basic arithmetics, logic, transfer and special instruction including bit, byte, word and multiple word manipulations. The CACHE memory included on the module minimizes the bus load and increase the processing speed.

  256K words Working Storage (256K RAM) included in the basic Processing Unit (PU) can be expanded by adding modules of 128K words to the configuration.

  Supra Net (STI and 2 x SBA) with redundant supra buses support checkpointing between active processes and standby processes resident in different PU's. Throughput of the Supra Net can be increased by adding Supra Bus Adapters (SBA) to the configuration.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

Data Channel Interface (MIA) interfaces the Processing Unit via the flatcable Data Channel to the channel unit. A maximum of 15 Channel Units can be connected to the Data Channel.

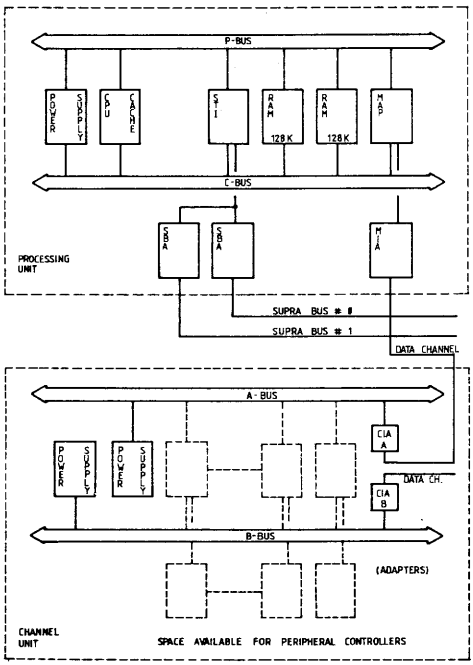
Power Supply. One module is included but if required due to expansion of the Processing Unit, an additional Power Supply can be installed.

Channel Unit

The Channel Unit is delivered with redundant Power Supplies, Transfer Buses and Data Channel Interfaces (CIA-A, CIA-B). No peripheral controllers are included, but the unit is prepared for installation of a great variety of dual bus peripheral modules from the CR80 product line as DISC CTRL, TAPE CTRL, LINE PRINTER CTRL, Communication Line CTRL, etc.

Module Diagram

The diagram shows the module configuration of the basic CR860 computer element.



(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Dimensions:

The dimensions of the CR860 FATOM computer depends of the selected model (number of basic elements). Figures given below are for one element including Processor Unit, Channel Unit, 19" rack, etc.

Height:	1800 mm
Width:	600 mm
Deepth:	730 mm
Weight:	230 kg

- Environmental Conditions:

The CR860 FATOM is designed to operate within the following limits:

Ambient temperature	15°C to 32°C
Humidity	20% RH to 80% RH (non-condensing)

For more detailes please ref. CR80 Handbook.

- Mains Power:

Voltage 220V	+20% -10%	(optional 110V)
Frequency 50HZ +/- 5HZ		(optional 60HZ)
Power Consumption: 1000W		for the basic CR860 element

- Processor Unit Expandability:

The basic Processing Unit contains module positions for extension by adding modules (CPU, RAM and other) to the configuration. The module positions are specified in CR8125M/225PC/00 PU-Crate datasheet.

Power available from the basic Power Supply allows for the following expansions:

128K words of memory.

By adding one Power Supply module following expansions are possible.

a. 768K words of memory
b. 768K words of memory +1 CPU
c. 768K words of memory +2 CPU
d. 256K words of memory +3 CPU

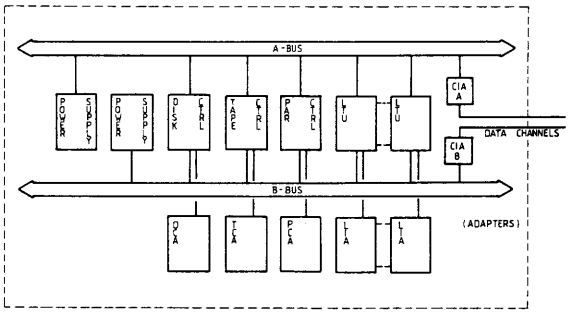
(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Input/Output Connectivity

The Channel Unit is equipped with cables, redundant Power Supplies, interfaces for connecting dual bus'ed peripheral controllers and adapters from the CR80 product range. When furnishing the Channel Unit the available power (5V: 55A) and the available module slots (17) have to be considered as specified in the CR8125M/425AB/00 CU-Crate datasheet.

As an example the configuration below is calculated.



	POWER CONSUMPTION			
	+5V	+12V	-12V	
1 x Disc CTRL	7.0	0.3	0.1	
1 x DCA	1.0	0	0	
1 x Tape CTRL	6.5	0.3	0.1	
1 x TCA	1.0	0	0	
1 x Parallel CTRL	6.5	0.3	0.1	
1 x PCA	1.0	0	0	
Subtotal	23	0.9	0.3	
Total available	50	4	4	
Available for LTU/LIA	27	3.1	3.7	
Number of LTU/LIA-N	27/3.5(+5V)	=		7 LTU/LIA

If the Input/Output connectivity of the CR860 is not sufficient for the system, extension element CR880 can be added to the configuration as required.

- Supervision and configuration subsystem (CR890 and CR891 for monitoring, control and maintenance of the CR861 FATOM computer are available as defined in datasheets for CR890 and CR891.

(CONTINUED)

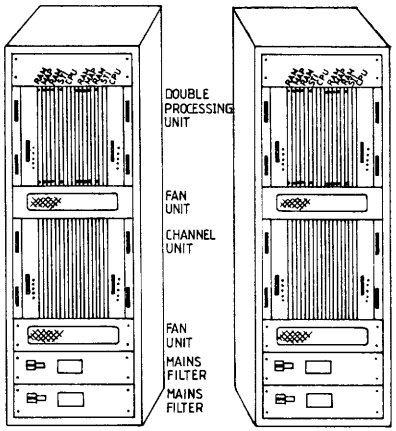
CHARACTERISTICS: (CONTINUED)

- Equipment List.

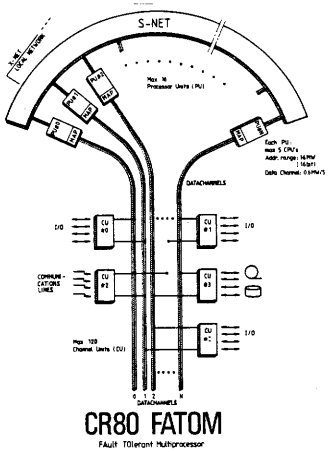
The items constituting the CR860 FATOM basic computer are defined in equipment list below:

Ref .	Type No.	Description	Qty
101.1	CR8101-/036--/00	RACK	1
125.0	CR8125M/225PC/00	PU-CRATE	1
125.2	CR8125M/425AB/00	CU-CRATE	1
105.0	CR8105M/020--/00	FAN UNIT	2
106.0	CR8106-/220--/00	MAINS FILTER	1
50.0	CR8050M/010--/00	POWER SUPPLY	3
107.0	CR8107-/010--/00	POWER DISTRIBUTION PANEL	1
20.0	CR8020M/000PC/00	MAP	1
71.0	CR8071M/010--/00	MIA	1
3.1	CR8030M/040PC/00	CPU CACHE	1
16.8	CR8016M/128PC/00	RAM 128K	2
21.1	CR8021M/010-C/01	STI(1)	1
72.0	CR8072M/010--/01	SBA	2
81.0	CR8081M/010A-/00	CIA-A	1
81.1	CR8081M/010-B/00	CIA-B	1
215.5	CR8215M/778--/00	SUPRABUS TERMINATION	4
211.5	CR8211M/738--/00	DATA CHANNEL TERMINATION	2
215.1	CR8215M/015--/00	CABLE SUPRABUS	2
211.1	CR8211M/015--/00	CABLE DATA CHANNEL	2
201.0	CR8201M/015--/00	CABLE RACK POWER	8
55.1	CR8055M/020--/00	MBT	6

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- The CR861 FATOM Computer allows the system designer to specify standard fault tolerant computers with up to 16 Processor Units to meet most medium systems demands to throughput and performance.

- CR861 is available in the following models ranging from a 2 PU (Processor Unit), 1 CU (Channel Unit) computer up to the maximum 16 PU, 8 CU, with each 2 PU, CU forming the basic element housed in a standard 19" rack:

  CR861/001- CR861/008; last three digits indicating number of elements (2 PU, CU).

- All elements in the CR861 are equipped identically with two Processing Units and a Channel Unit as defined below and shown in module diagram overleaf.

  Processor Unit

  The two PU's are equipped identically as follows:

  Memory Mapping Module (MAP) provides addressing of up to 16M words of virtual memory, demand paging and a protection mechanism to prevent access from unauthorized users.

  Central Processor Unit (CPU) is a general purpose processor with 16 bits word length and a standard instruction set of basic arithmetics, logic, transfer and special instructions including bit, byte, word and multiple word manipulations. The CACHE memory included on the module minimizes the bus load and increases the processing speed.

  256K words Working Storage (256K RAM) included in the basic Processing Unit (PU) can be expanded by adding modules of 128K words to the configuration.

  Supra Net (STI and 2 x SBA) with redundant supra buses support checkpointing between active processes and standby processes resident in different PU's. Throughput of the Supra Net can be increased by adding Supra Bus Adapters (SBA) to the configuration.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

Data Channel Interface (MIA) interfaces the Processing Unit via the flatcable Data Channel to the channel unit. A maximum of 15 Channel Units can be connected to the Data Channel.

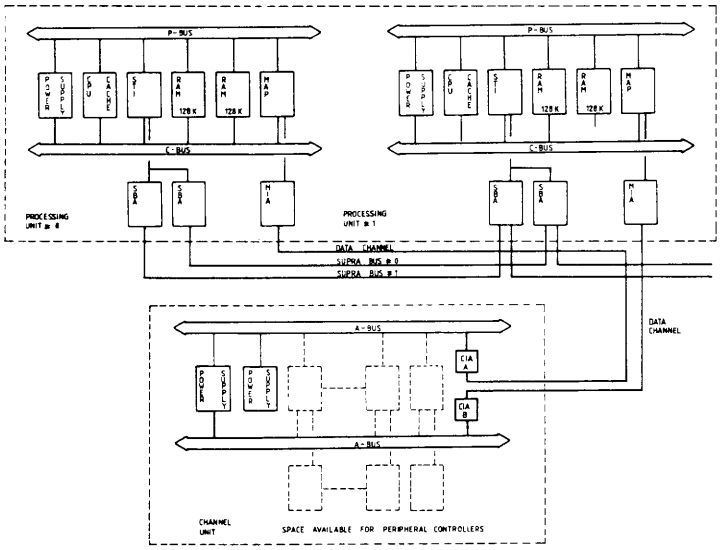
Power Supply installed ensures that power is available for expansion of the Processing Units working storage.

Channel Unit

The Channel Unit is delivered with redundant Power Supplies, Transfer Buses and Data Channel Interfaces (CIA-A, CIA-B). No peripheral controllers are included, but the unit is prepared for installation of a great variety of dual bus peripheral modules from the CR80 product line as DISC CTRL, TAPE CTRL, LINE PRINTER CTRL, Communication Line CTRL, etc.

Module Diagram

The diagram shows the module configuration of the basic CR861 computer element.



(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Dimensions:

The dimensions of the CR861 FATOM computer depends of the selected model (number of basic elements). Figures given below are for one element including Processor Unit, Channel Unit, 19" rack, etc.

Height:	1800 mm
Width:	600 mm
Deepth:	730 mm
Weight:	250 kg

- Environmental Conditions:

The CR861 FATOM is designed to operate within the following limits:

Ambient temperature	15^{o}C to 32^{o}C
Humidity	20% RH to 80% RH (non-condensing)

For more detailes please ref. CR80 Handbook.

- Mains Power:

Voltage 220V	+20% -10%	(optional 110V)
Frequency 50HZ +/- 5HZ		(optional 60HZ)
Power Consumption: 1600W		for the basic CR860 element

- Processor Unit Expandability:

The basic Processing Unit contains module positions for extension by adding modules (CPU, RAM and other) to the configuration. The module positions are specified in CR8112M/212PC/00 PU-Crate datasheet.

Power available from the basic Power Supply allows for the following expansions:

128K words of memory.

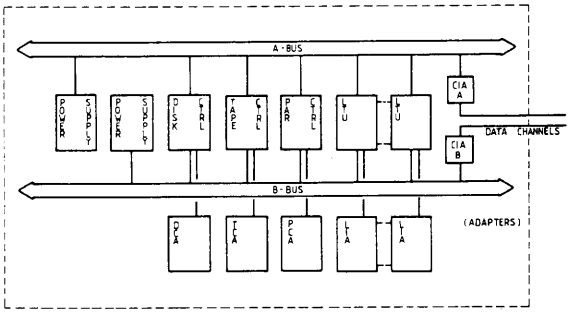
(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Input/Output Connectivity

The Channel Unit is equipped with cables, Power Supply interfaces for connecting dual bus'ed peripheral controllers and adapters from the CR80 product range. When furnishing the Channel Unit the available power (5V: 55A) and the available module slots (17) have to be considered as specified in the CR8125M/425AB/00 CU-Crate datasheet.

As an example the power consumption of the configuration is calculated below.



	POWER CONSUMPTION			
	+5V	+12V	-12V	
1 x Disc CTRL	7.0	0.3	0.1	
1 x DCA	1.0	0	0	
1 x Tape CTRL	6.5	0.3	0.1	
1 x TCA	1.0	0	0	
1 x Parallel CTRL	6.5	0.3	0.1	
1 x PCA	1.0	0	0	
Subtotal	23	0.9	0.3	
Total available	50	4	4	
Available for LTU/LIA	27	3.1	3.7	
Number of LTU/LIA-N	27/3.5(+5V)		=	7 LTU/LIA

If the Input/Output connectivity of the CR861 is not sufficient for the system, extension element CR880 can be added to the configuration as required.

- Supervision and configuration subsystem (CR890 and CR891) for monitoring, control and maintenance of the CR861 FATOM computer are available as defined in datasheets for CR890 and CR891.

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

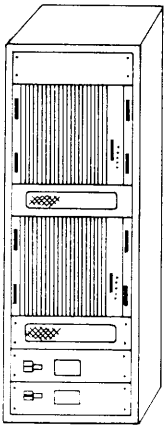
- Equipment List.

The items constituting the CR861 FATOM basic computer are defined in the equipment list below:

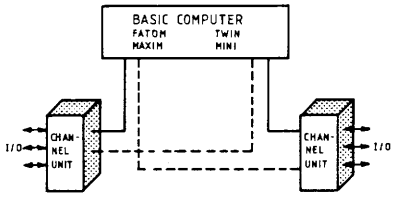
Ref .	Type No.	Description	Qty
101.1	CR8101-/036--/00	RACK	1
125.0	CR8125M/225PC/00	PU-CRATE	1
125.2	CR8125M/425AB/00	CU-CRATE	1
105.0	CR8105M/020--/00	FAN UNIT	2
106.0	CR8106-/220--/00	MAINS FILTER	2
50.0	CR8050M/010--/00	POWER SUPPLY	4
107.0	CR8107-/010--/00	POWER DISTRIBUTION PANEL	2
20.0	CR8020M/000PC/00	MAP	2
71.0	CR8071M/010--/00	MIA	2
3.1	CR8030M/040PC/00	CPU CACHE	2
16.8	CR8016M/128PC/00	RAM 128K	4
21.1	CR8021M/010-C/01	STI(1)	2
72.0	CR8072M/010--/01	SBA	4
81.0	CR8081M/010A-/00	CIA-A	1
81.1	CR8081M/010-B/00	CIA-B	1
215.5	CR8215M/778--/00	SUPRABUS TERMINATION	8
211.5	CR8211M/738--/00	DATA CHANNEL TERMINATION	1
215.1	CR8215M/015--/00	CABLE SUPRABUS	4
211.1	CR8211M/015--/00	CABLE DATA CHANNEL	2
201.0	CR8201M/015--/00	CABLE RACK POWER	8
55.1	CR8055M/020--/00	MBT	6

CR 880 EXTENSION ELEMENT

SYSTEM LAYOUT



SYSTEM STRUCTURE



CHARACTERISTICS:

- The CR880 Extension Element allows the system designer to expand the CR80 computers (FATOM, MAXIM, TWIN and MINI) to meet almost any requirement for Input/Output connectivity.

- CR880 contains 1 or 2 Channel Units with Cables, Power Supplies, Interface Modules and Transfer Buses housed in a standard 19" rack.

- CR880 is available in the following models each equippped in accordance with the computer type (redundant/non-redundant, mapped/unmapped) as defined in module diagram overleaf:

CR880/031,-/032; MINI extension	(with 1 CU: 031, with 2 CU's: 032)
CR880/041,-/042; TWIN extension	(with 1 CU: 041, with 2 CU's: 042)
CR880/051,-/052; MAXIM extension	(with I CU: 051, with 2 CU's: 052)
CR880/061,-/062; FATOM extension	(with 1 CU: 062, with 2 CU's: 062).

- Input/Output Connectivity

The CR880 Extension Element is equipped with Cables, Power Supplies and Interfaces for connecting dual bussed peripheral controllers (TWIN and FATOM) or single bussed peripheral controllers (MINI and MAXIM) and adapters from the CR80 product range. When equipping the Channel Units the available power (50 A) and available module slots have to be considered as specified in:

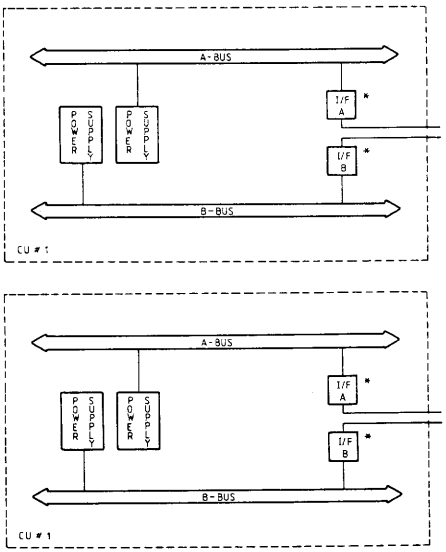
CR8125M/425AB/00	CU Crate datasheet (TWIN and FATOM)
CR8125M/325A-/00	CU Crate datasheet (MINI and MAXIM)

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Module Diagram:

The diagram shows the module configuration of the CR880 Extension Elements, and defines the different models.



* In MINI and TWIN extension models (CR880/03X and CR880/04X) I/F modules correspond to EIA modules . In MAXIM and FATOM extension models (CR880/05X and CR880/06X) I/F modules correspond to CIA modules.

In single CU extension elements, indicated by "1" in last digit in model number, only one CU is installed.

In non-redundand systems (CR880/03X MINI and CR880/05X MAXIM) the B-BUS and corresponding Power Supply and I/F are not installed.

- Dimensions

Height:	1800 mm
Width:	600 mm
Deepth:	730 mm
Weight:	175 kg (basic configuration)

- Mains Power:

Voltage 220V	+20% -10%	(optional 110V)
Frequency 50HZ +/- 5HZ		(optional 60HZ)
Power Consumption: 500W		(basic configuration)

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

- Equipment List

Below equipment lists for CR880 Extension models are found:

Model CR880/031 and CR880/032: "MINI"

Ref .	Type No.	Description	Qty /032	Qty /031
101.1	CR8101-/036-/00	RACK	1	1
125.3	CR8125M/325A-/00	CU-CRATE	2	1
105.0	CR8105M/020-/00	FAN UNIT	2	1
106.0	CR8106-/220-/00	MAINS FILTER	1	1
50.0	CR8050M/010-/00	POWER SUPPLY	2	1
107.0	CR8107-/010-/00	POWER DISTRIBUTION PANEL	2	1
55.1	CR8055M/020-/00	MBT	2	1
88.0	CR8088M/010A-/00	EIA-A	2	1
212.1	CR8212M/030-/00	CABLE EXTENSION BUS	2	1
201.0	CR8201M/015-/00	CABLE RACK POWER	6	3

Model CR880/041 and CR880/042: "TWIN"

Ref .	Type No.	Description	Qty /042	Qty /041
101.1	CR8101-/036-/00	RACK	1	1
125.2	CR8125M/425AB/00	CU-CRATE	2	1
105.0	CR8105M/020-/00	FAN UNIT	2	1
106.0	CR8106-/220-/00	MAINS FILTER	2	2
50.0	CR8050M/010-/00	POWER SUPPLY	4	2
107.0	CR8107-/010-/00	POWER DISTRIBUTION PANEL	2	2
55.1	CR8055M/020-/00	MBT	4	2
88.0	CR8088M/010A-/00	EIA-A	2	1
88.1	CR8088M/010-B/00	EIA-B	2	1
212.1	CR8212M/030-/00	CABLE EXTENSION BUS	4	2
201.0	CR8201M/015-/00	CABLE RACK POWER	8	4

Model CR880/051 and CR880/052: "MAXIM"

Ref .	Type No.	Description	Qty /052	Qty /051
101.1	CR8101-/036-/00	RACK	1	1
125.3	CR8125M/325A-/00	CU-CRATE	2	1
105.0	CR8105M/020-/00	FAN UNIT	2	1
106.0	CR8106-/220-/00	MAINS FILTER	1	1
50.0	CR8050M/010-/00	POWER SUPPLY	2	1
107.0	CR8107-/010-/00	POWER DISTRIBUTION PANEL	2	1
55.1	CR8055M/020-/00	MBT	2	1
81.0	CR8081M/010A-/00	CIA-A	2	1
211.2	CR8211M/030-/00	CABLE DATA CHANNEL	2	1
201.0	CR8201M/015-/00	CABLE RACK POWER	6	3

(CONTINUED)

CHARACTERISTICS: (CONTINUED)

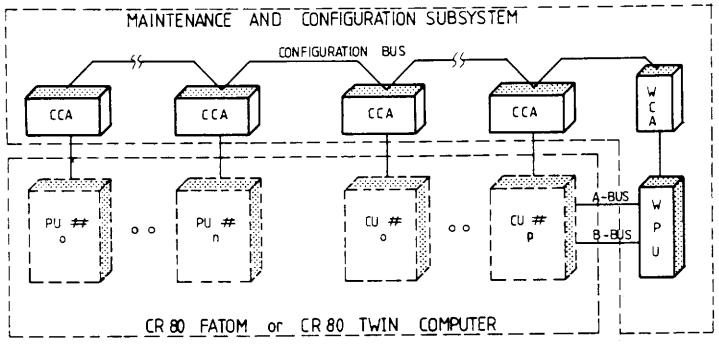
Model CR880/061 and CR880/062: "FATOM"

Ref .	Type No.	Description	Qty /062	Qty /061
101.1	CR8101-/036—/00	RACK	1	1
125.2	CR8125M/425AB/00	CU-CRATE	2	1
105.0	CR8105M/020—/00	FAN UNIT	2	1
106.0	CR8106-/220—/00	MAINS FILTER	2	2
50.0	CR8050M/010—/00	POWER SUPPLY	4	2
107.0	CR8107-/010—/00	POWER DISTRIBUTION PANEL	2	1
55.1	CR8055M/020—/00	MBT	4	2
81.0	CR8081M/010A-/00	CIA-A	2	1
81.1	CR8081M/010-B/00	CIA-B	2	1
211.2	CR8211M/030—/00	CABLE DATA CHANNEL	4	2
201.0	CR8201M/015—/00	CABLE RACK POWER	8	4

DATASHEET FOR:

CR 890 MAINTENANCE and CONFIGURATION SUBSYSTEM

SYSTEM STRUCTURE



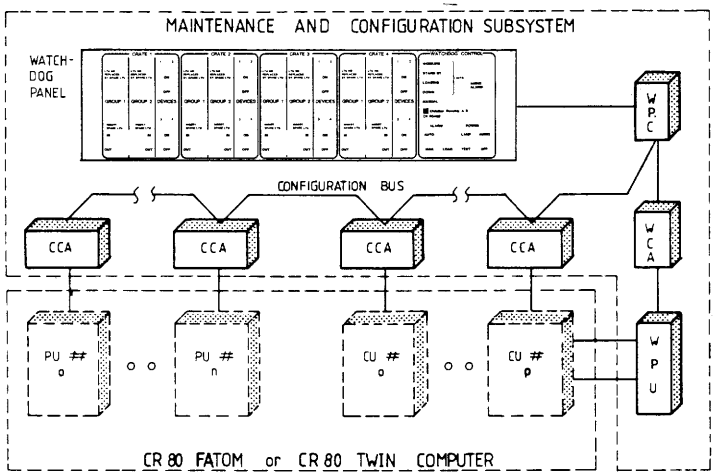
CHARACTERISTICS:

- The CR890 Maintenance and Configuration Subsystem (MCS) is designed for automatic monitoring and configuration control of the CR80 computer family (FATOM & TWIN).
- CR890 MCS contains modules of the same dimensions as those in the CR80 computer. As the CR80 is prepared for insertion of the MCS modules, the installation of the subsystem is performed by adding the modules and connecting the configuration bus.
- CR890 MCS is a micro computer based subsystem, with the CPU located in the WPU module and firmware programs resident in PROM located on the WCA.
- Monitoring and control, as PU and CU crate DC voltages monitoring, switching of dual ported adapter (LIA-S) enabling/disabling of PU external interfaces etc, is performed from the Crate Configuration Adapter modules (CCA).
- The CR890 processor section (WCA and WPU) can be delivered as a mechanical self-contained unit housed in a Watchdog Crate or it can be delivered as modules for insertion in one of the CR80 computers CU-Crates.
- CR890 MCS is available in following models:

CR890/001-CR890/032; Processor section (WCA and WPU) delivered as modules. Last two digits indicating number of CCA modules (PU + CU crates).

CR890/101-CR890/132; Processor section (WCA and WPU) delivered housed in a Watchdog Crate. Last two digits indicating number of CCA modules (PU + CU crates).

SYSTEM STRUCTURE



CHARACTERISTICS:

- The CR891 Maintenance and Configuration Subsystem (MCS) is designed for automatic and/or manual monitoring and configuration control of the CR80 computer family (FATOM & TWIN).

- CR891 MCS contains modules of the same dimensions as these in the CR80 computer. As the CR80 is prepared for insertion of the MCS modules, the installation of the subsystem is performed by adding the modules and connecting the configuration bus.

- CR891 MCS is a micro computer based subsystem, with the CPU located in the WPU module and firmware programs resident in PROM located on the WCA. Also, the WPC contains a micro processor to allow for manual monitoring and configuration control from the Watchdog Panel.

- CR891 MCS is failure tolerant.
  The configuration Bus contains two redundant transfer buses. During automatic operation the configuration bus is supervised from the WCA to allow for switch to manual operation when failures are recognized.

- Monitoring and control, as PU and CU crate DC voltages monitoring, switching of dual ported adapter (LIA-S), enabling/disabling of PU external interfaces etc, is performed by the Crate Configuration Adapter modules (CCA).

- The CR891 processor section (WCA, WPU, WPC and Panel) can be delivered as a mechanical self-contained unit housed in a Watchdog Crate or it can be delivered as modules for insertion in one of the CR80 computers CU-Crates.

- Optional panels (Slave Panels) one per CU-Crate can be installed for manual monitoring and control of each crate.

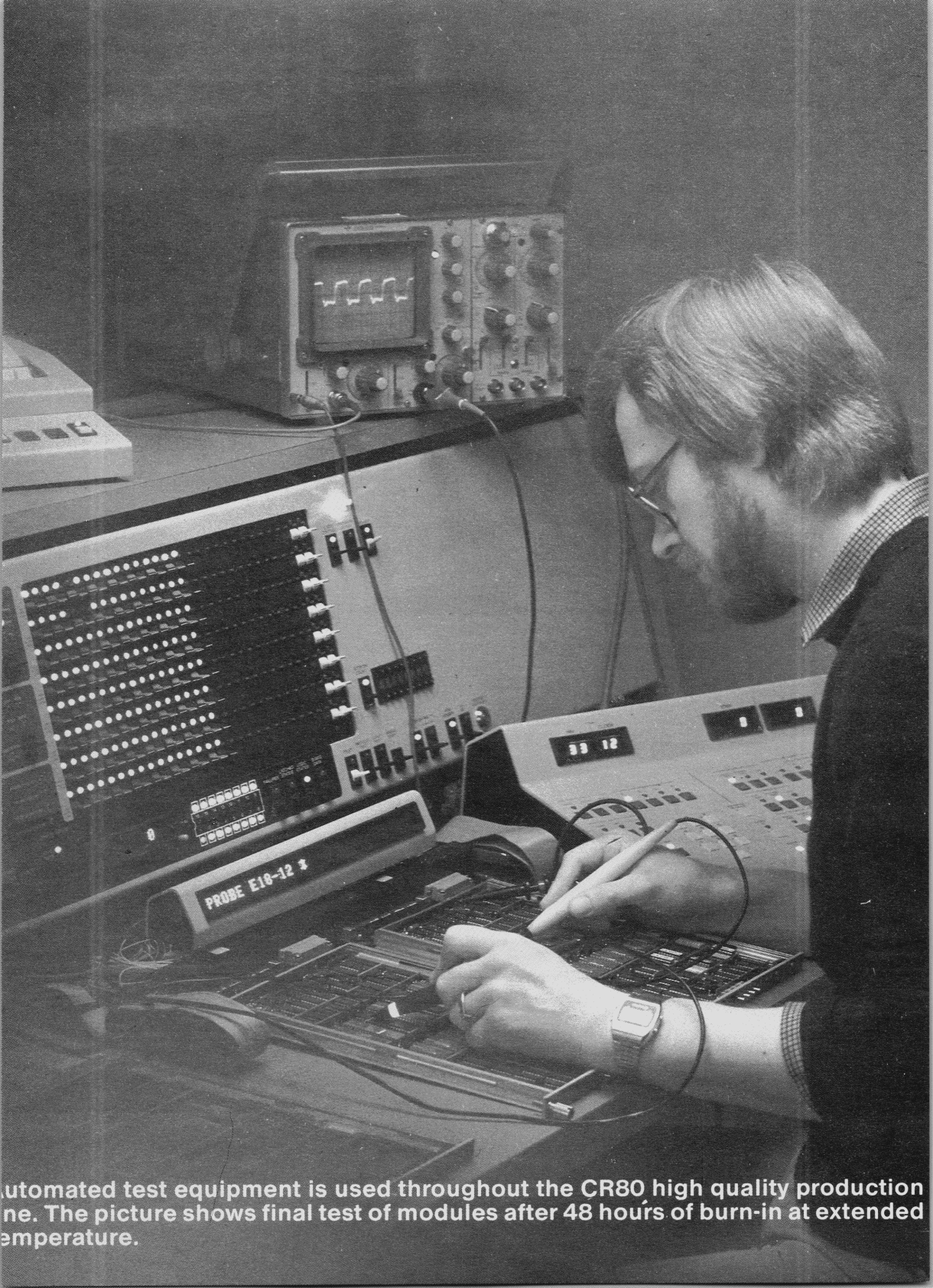
(CONTINUED)

CHARACTERISTICS: (CONTINUED)

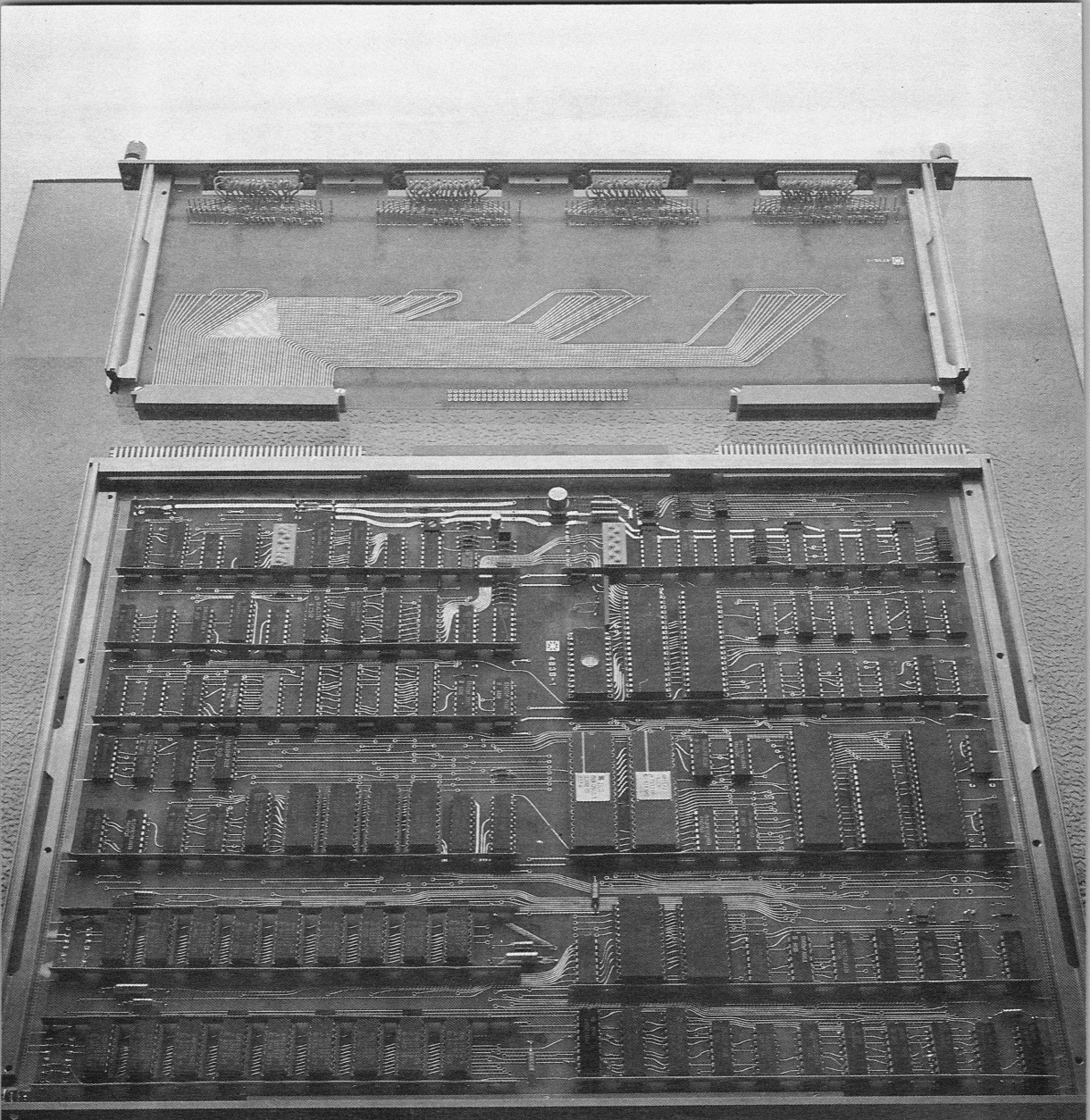
CR891 MCS is available in following modules:

CR891/001-CR891/032; Processor section (WCA, WPU, WPC and Watchdog Panel) delivered as modules. Last two digits indicating number of CCA modules (PU + CU crates)

CR891/101-CR891/132; Processor section (WCA, WPU, WPC and Watchdog Panel) delivered housed in a Watchdog Crate. Last two digits indicating number of CCA modules (PU + CU crates).



utomated test equipment is used throughout the CR80 high quality production ne. The picture shows final test of modules after 48 hours of burn-in at extended emperature.



The LTU is the CR80 standard microcomputer based Communication Interface here shown with its associated Line Interface Adapter (LIA-N). The on-board microcomputer, based on its downloaded program, off-loads the CR80 Computer and adapts to all line oriented protocol processing.

12. RELIABILITY, MAINTAINABILITY AND AVAILABILITY

12.1 Introduction

Reliability and maintainability engineering was a significant factor in guiding the development of the CR80.

The CR80 architecture is designed with a capability to achieve a highly available computer system in a cost-effective way. It will provide a reliable set of services to the users of the system, because it is customized to the actual availability requirements. The CR80 fault tolerant computers are designed to avoid single point errors of all critical system elements by provision of redundancy of data paths, processor capabilities and power supplies.

The architecture reflects the fact that the reliability of peripheral devices is lower than that of the associated CR80 Peripheral Modules. This does equally well apply to communication lines where modems are used as part of the transmission media. Thus, the peripheral devices, modems, communication lines, etc. impact the system availability much more than the corresponding Peripheral Modules. To achieve a high system availability it is therefore necessary to provide redundancy of peripheral devices as well as of Peripheral Modules.

In connection with communication lines the CR80 CU's utilizes the unique concept of n out of n+1 LTU's and n LIA-S to achieve the highest possible equipment availability. In this way the functions of a failed LTU automatically can be taken over by the spare LTU, electronically controlled by enabling the appropriate LIA-S.

In section 12.5.3 RMA calculations, are shown the high availability obtained by these methods. A typical FATOM or TWIN system with 32 LTUs + 2 spare LTU's will have a system Mean Time Between Failure (MTBF) in excess of 8 years. That is, once per 8 year not all 32 LTUs will be functioning, but only 31 out of the 32 LTUs.

To assure this very highly reliable product, several criterias were also introduced on the module level:

- An extensive use of hi-rel, mil-spec components, ICs are tested to the requirements of MIL-STD 883 level B or similar.

- All hardware is designed in accordance with the general CR80 H/W design principles. These include derating specifications, which greatly enhance the reliability and reduce the sensibility to parameter variations.

- Critical modules feature a Built-In test (BIT) capability as well as a display of the main states of the internal process by Light Emitting Diodes on the module front plate. This greatly improves module maintainability, as it provides debug and trouble shooting methods, which reduce the repair time.

- A high quality production line, which includes high quality soldering, inspection, burn-in and an extensive automatic functional test.

12.2 Reliability

The reliability of a system is reflected by the period over which it is able to provide the required functional services. The system reliability is dependant on the reliability as well as the redundancy of the modules and units which constitute the system. It is determined as part of a Reliability, Maintainability and Availability (RMA) analysis (refer to section 12.5).

The reliability of a module concerns the ability of survival in a specific environment. The reliability for electronic equipment can be expressed in a fixed value: Mean Time Between Failures (MTBF).

MTBF is calculated by modelling the electrical/mechanical structure of the module in accordance with MIL-HDBK 217.

This model describes how the elements functionally are connected and how the different parameters affect the failure of every single component.

12.2.1 Typical reliability model and failure rate calculations:

A CR80 reliability models is normally a single series configuration of all components because the module's performance relies on every single elements.

When calculating these reliability models the MTBF is not used, but the inverse of MTBF, the failure rate.

Failure rate per million hours: LAMBDA = 10^6/MTBF.

The module failure rate is simply the sum of the individual component failure rates.

$$LAMBDA_M = LAMBDA_a + LAMBDA_b + ...+ LAMBDA_i + ...$$

$$MTBF_M = 10^6/LAMBDA_M$$

As an example of predicting the failure rate $LAMBDA_p$ of a specific component is taken the IC 54LS112(MIL-M-38510 type 30103).

MIL-HDBK 217C prescribes the model:

$$LAMBDA_P = PI_Q (C_1 * PI_T + (C_2 + C_3) PI_E) PI_L$$

PI_Q is the quality factor. 54LS112 is procured to screening requirements of MIL-STD-883, Method 5005 Class B giving quality level B-1 and PI_Q = 3.

PI_E is the application environmental factor. All failure rates are predicted providing a ground fixed environment. This gives a PI_E = 2.5.

PI_L is the device learning factor 54LS112 has been produced for over six months giving a PI_L = 1.

PI_T is the temperature acceleration factor. Below listed parameters are used to determine this:

- Technology temperature factor ("A)
- Empiric typical power dissipation
- Thermal resistances ("Q_{jc}, Q_{ca}")
- Forced Air Cooling correction factor ("D")
- Ambient temperature ("Ta").

For LSTTL technology is A = 5794, 54LS112 power dissipation is 30mWatt, Q_{jc} = 26°C/W and Q_{ca} = 59°C/W.

The maximum component ambient temperature, in MTBF calculations, is 43°C based on normal room ambient on 28°C and a maximum temperature rise through the equipment at 15°C, thus giving Ta = 43°C. Efficient forced air cooling enables an assumption on D = 0.1.

This gives us a T junction = 44°C and following the formula in MIL-HDBK 217C we derive PI_T = 0.32.

C_1, C_2 and C_3 are complexity factors based on chip gate count, sealing and No. of pins. From the MIL-HDBK we find:

C_1 = 0.0046 (16 gates)
C_2 = 0.0006 (16 gates)
C_3 = 0.005 (Hermetic glass seal, 16 pins)

Now we can calculate the failure rate:

$LAMBDA_p$ = 3 (0.0046 * 0.32 + (0.0006 + 0.005)) * 1 = 0.05 FPMH

12.3 Equipment Maintainability

The CR80 is designed for ease of maintenance. The system is built of modules, each comprising a complete well-defined function. For example, the CPU is only one module which in case of error is changed in one operation.

In case the correct function of the sytem is extremely critical, the CR80 will have built-in on-line diagnostic programs. Even though the CR80 is highly reliable, failures can occur, and the below described diagnostics will minimize the downtime for a system.

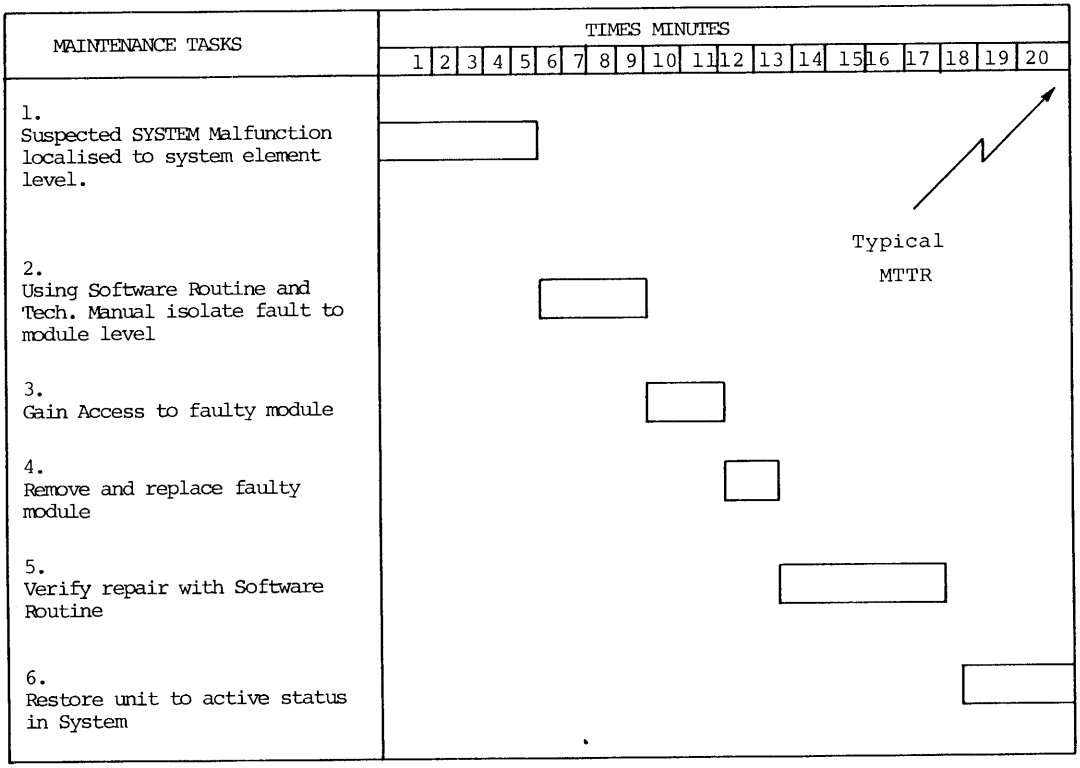
An off-line diagnostics software package is employed to ease the diagnostics in case of error. Normally, this software package is stored on disk. Then after initiation, the program will test all modules forming the system and print the name and address of the erroneous module on the operator's console. Having replaced the erroneous module, the CR80 is ready for operation again. The operator might if necessary run the off-line diagnostics program once more to verify that the system is now working without errors.

Having performed time-line analysis of the tasks associated with fault detection, isolation, repair, and verification, the Mean Time to Repair (MTTR) is found to be less than 30 minutes for the modules and 60 - 90 minutes for peripherals.

Figure 12.3-1 shows the typical fault isolation and replacement sequence.

12.4 Equipment MTBF, MTTR Values

A list containing the CR80 modules MTBF and MTTR values are shown for easy reference in table 12.4a (MTBF value is also given for each CR80 Module on its Datasheet, which in case of conflict is the authoritative value). They are derived from reliability predictions, and the R & M figures for peripherals are based on data supplied by the manufacturers.



TYPICAL FAULT ISOLATION AND REPLACEMENT SEQUENCE

FIGURE 12.3-1

Module CR No.	Item Description	MTBF (hrs.)	FPMH	MTTR (minutes)
8002	CPU, SCM	36500	27.4	30
8003	CPU, CACHE	26100	38.3	30
8009	EPM	172400	5.8	30
8013	EPROM	91700	10.9	30
8016	RAM 128K/64K	17000/29600	58.8/33.8	30
8020	MAP	19400	51.6	30
8021	STI	32800	30.5	30
8037	UNIVAC I/F	33200	30.1	30
8039	IBM CH. I/F	32400	30.9	30
8044	DISC CTRL DUAL/SINGLE	30200/39400	33.1/25.4	30
8045	TAPE CTRL 16K	35700	28.0	30
8046	DUAL PAR. CTRL	35700	28.0	30
8047	ST. FD. CTRL DUAL/SINGLE	59500/84700	16.8/11.5	30
8050	POWER SUPPLY	26800	37.3	30
8055	MBT	285700	3.5	30
8059	MBE	10000000	0.1	30
8066	LTU DUAL/SINGLE	27600/45000	36.9/22.2	30
8070	CSA	769000	1.3	30
8071	MIA	85500	11.7	30
8072	SBA	90100	11.1	30
8073	TIA	117600	8.5	30
8074	EPA	256000	3.9	30
8078	IBA	21600	46.2	30
8079	UIA	15600	64.0	30
8081	CIA A & B	71400	14.0	30
8082	LIA-N	10000000	0.1	30
8083	LIA-S (Switch + Common)	534759/3571428	1.87+0.28	30
8084	DCA	46900	21.3	30
8085	TCA	128200	7.8	30

R & M VALUES FOR MODULES AND PERIPHERALS

Table 12.4a (continues)

Module CR No.	Item Description	MTBF (hrs.)	FPMH	MTTR (minutes)
8086	PCA	185200	5.4	30
8087	SFA	10000000	0.1	30
8088	EIA A & B	113600	8.8	30
8106	MAINS FILTER DISTRIBUTION	625.000	1.6	30
8115	Minicrate	26300	38	60
8125/PC	PU-CRATE	200000	5.0	60
8124/AB	CU-CRATE	703630	1.4	60

Peripheral No.	Item Description	MTBF (hrs)	FPMH	MTTR (minutes)
8300/---	DISC DRIVE, SMD, 40-300MB	4000	250.0	90
8301/---	DISC DRIVE, CMD (16-48)+16MB	4000	250.0	90
8302/---	DISC DRIVE, MMD, 12-80MB	8000	125.0	60
8307/---	FLOPPY DRIVE, dual/single sided	8000	125.0	30
8320/001	TAPE STATION, Pertec FT 8000	8000	125.0	60
8320/002	TAPE STATION, Pertec FT 9000	2500	400.0	60

R & M Values for Modules and Peripherals
Table 12.4a

12.5 RMA Analysis

This section provides information with respect to RMA analysis of a system. It includes the detailed formulas which apply as part of the RMA calculations and is followed by supporting examples.

The RMA analysis of a system provides information on how much of the time the system provides a given set of required functional capabilities, i.e. provides operative availability. It shows how many times the system is not operative during a given period and for how long. A system may be operative even with one or more elements of the total system down or taken off-line for the purpose of repairing and/or replacement of defect modules/units. Note that this is operative as seen by a user of the functional capabilities, not as seen by maintenance personnel.

The basis for determining the system level availability is an RMA model of serial and parallel system elements. Each of these elements define a specific subset of the total system with a well defined state: either functioning or not.
Serial elements refer to elements all of which have to be available for that set to be available.
Parallel elements describes those sets where not all elements need to be available, the number determined by the required service level or the redundancy provided.
The subsequent sections introduces the basic RMA building stones.

12.5.1 Series Element

The mean time between failures of a series of n different RMA elements is made up as follows:

$$MTBF_S = \frac{10^6}{LAMBDA_S}$$

where the series failure rate is determined by the sum of the failure rates of the elements:

$$LAMBDA_s = LAMBDA_1 + LAMBDA_2 + \ldots\ldots + LAMBDA_i + \ldots\ldots + LAMBDA_n$$

where $LAMBDA_i$ denote the failure rate of the i'th element.

The availability of a system of n different serial RMA elements is determined by

$$A = A_1 * A_2 * \ldots\ldots * A_i * \ldots\ldots * A_n$$

In most cases A_i is very close to 1, and it is therefore much easier during the calculations to use the unavailability B_i, which is defined as follows:

$$B_i = 1 - A_i$$

The availability of a system is thus:

$$A_s = (1-B_1)(1-B_2)\ldots(1-B_i)\ldots(1-B_n)$$

For $B_i << 1$ the following approximation is valid:

$$A_s = 1 - (B_1 + B_2 + \ldots + B_i + \ldots + B_n)$$

This gives:

$$B_s = (B_1 + B_2 + \ldots\ldots + B_n) \text{ for } B_i << 1$$

The ration between the sum of unavailabilities and the sum of failure rates determines the mean repair rate of the system. Thus, the mean time to repair a system of series elements is:

$$MTTR_S = \frac{B_S}{LAMBDA_S} \quad \text{if } MTTR_S << MTBF_S$$

(Note, this actually corresponds to the weighted mean because the sum of B_i = sum of ($LAMBDA_i * MTTR_i$).

12.5.2 Parallel Elements

When RMA elements are in parallel, it is required that one or more of the parallel units are operative simultaneously to obtain the required system performance. The actual number of parallel units required are dependant of the actual models. Assuming operational redundancy and negligle recovery time, the calculation rules are:

a. Mean Time Between Failure

When the parallel elements have defined MTBF and MTTR values the following rules apply:

1 of 2 equal parallel elements

$$\text{Element } MTBF_E = \frac{2 * MTBF * MTTR + MTBF^2}{2 \times MTTR}, \text{ or}$$

$$MTBF_E = \frac{MTBF^2}{2 \times MTTR}, \text{ Provided MTTR} << \text{MTBF}$$

The corresponding element failure rate are determined by

$LAMBDA_E = 2 * LAMBDA^2 * MTTR/10^6$, provided MTTR << MTBF.

n of n+1 Equal Parallel Elements

$$\text{Element MTBF}_E = \frac{(n+1)*MTBF*MTTR + MTBF^2}{n(n+1)MTTR}, \text{ or}$$

$$MTBF_E = \frac{MTBF^2}{n(n+1)MTTR}, \quad \text{provided } (n+1)*MTTR << MTBF$$

The corresponding element failure rate is
$LAMBDA_E = n*(n+1)*LAMBDA^2*MTTR/10^6$, provided (n+1) MTTR << MTBF.

1 of 2 Un-equal Parallel Elements

$$MTBF_E = \frac{MTBF_1 * MTBF_2}{MTTR_1 + MTTR_2}, \quad \text{provided MTTR} << \text{MTBF}$$

The corresponding element failure rate is
$LAMBDA_E = LAMBDA_1*LAMBDA_2 (MTTR_1 + MTTR_2)/10^6$,
provided MTTR << MTBF

b. Mean Time To Repair

The element mean time to repair, $MTTR_E$, corresponds to the period where more than n out of the n+1 units are not available i.e. the element not fully operative.

1 of 2 Parallel Elements

$$MTTR_E = \frac{MTTR}{2}$$

n of n+1 Parallel Elements

$$MTTR_E = \frac{MTTR}{2}$$

c. Availability

The availability corresponds to the ratio between the MTBF and the total operative time, which is equal to the sum of MTBF and MTTR for the element, thus:

$$A_E = \frac{MTBF_E}{MTBF_E + MTTR_E}$$

1 of 2 equal parallel elements

$$A_E = \frac{MTBF^2 + 2*MTTR*MTBF}{(MTTR + MTBF)^2}$$

$$A_E = 1 - \frac{MTTR^2}{MTBF^2}, \text{ provided MTTR} << \text{MTBF}$$

or

$$A_E = 1 - LAMBDA^2 * MTTR^2/10^{12} \text{ provided MTTR} << \text{MTBF}$$

n of n+1 Equal Parallel Elements

$$A_E = 1 - \binom{n+1}{2} * \frac{MTTR^2}{MTBF^2}, \text{ provided MTTR} << \text{MTBF}$$

or

$$A_E = 1 - \binom{n+1}{2} * LAMBDA^2 * MTTR^2/10^{12}, \text{ provided MTTR} << \text{MTBF}.$$

Certain complicated RMA models can with advantage be broken down to groups of series/parallel elements, which are combine using the general formulas for Availability (A) and Unavailability (B) where B = 1-A.

- Series connection of two elements 1 and 2.

  $A = A_1 * A_2$

  $A = 1-(B_1 * B_2)+ B_1 * B_2$

- Parallel connection of two elements 1 and 2.

  $A = A_1 + A_2 - A_1 * A_2$

  $A = 1-B_1 * B_2$

- Parallel connection of two elements 1 and 2, and they are in series with element 3.

  $A = A_1 * A_3 + A_2 * A_3 - A_1 * A_2 * A_3$

  $A = 1 - B_3 - B_1 * B_2 + B_1 * B_2 * B_3$

12.5.3 RMA Calculations

In this chapter will be shown some examples of RMA calculations. First, as an in depth example, the n of n+1 redundancy concept with LTU's and switching LIA-S will be analyzed and discussed. Next, the results will be used in a simplified RMA analysis of a CR80 FATOM communication system with 3 PU's and 2 CU's with 2 disc drives. A CR80 TWIN system with 2 PU's and 2 CU's will also be calculated. The results are summarized below:

MODEL	LAMBDA	MTTR	B	AVAILABILITY
CU, 8 LTU + 1 SPARE	2.3 FPMH	0.5 hour	$1.2/10^6$	99.99988%
CU, 16 LTU + 1 SPARE	4.9 FPMH	0.5 hour	$2.5/10^6$	99.99975%
CR80 FATOM,2 CU's EACH 16 LTU + 1 SPARE	13.3 FPMH	0.6 hour	$8/10^6$	99.9992%
CR80 TWIN, 2 CU's EACH 16 LTU + 1 SPARE	12.8 FPMH	0.6 hour	$7.9/10^6$	99.99921 %

A prerequisite for designing a system RMA model is an exact definition of the required service level of that particular system i.e. the minimum number of active functional elements necessary to fullfill the operational requirements.

Another important factor to be considered is the total maintenance philosophy because of the great impact on the resultant system reliability. To achieve the optimum system reliability very short repair times like MTTR = 0.5 hour are necessary. Items with low failure rates and forming part of the redundant concept, like LIA-S switch, can be repaired once per day and thus avoiding taking a functioning group of channels out of service during normal operation. The MTTR for LIA-S switch failure, are in this case therefore 12 hours.

Preventive maintenance on the unused redundant elements in the system has to be included in the considerations. During normal system operation the unused redundant elements might fail without affecting normal operation. When a failure in the operating part of the system then occurs it will not be able to switch to the redundant circuitry and thus maintain full service level. The redundant elements must therefore be functionally tested at regular intervals to maintain overall system availability . If they are tested too often it might degrade operational availability because of excessive testing, likevise if they are tested too seldom the probability of multiple hardware errors will increase and thus decrease the reliability.

12.5.3.1 LTU/LIA-S Communication Lines

To achieve the high reliability required in advanced communication systems different kinds of redundancy are required. The fault tolerant CR80 FATOM and TWIN uses the unique n out of n+1 redundancy concept in connection with communication lines. This method means that the functions of a failed LTU automatically can be taken over by the spare LTU, electronically controlled by enabling the appropriate LIA-S, see figure 12.5.3.1-1.

The resultant RMA model of the n out n+1 LTU/LIA-S concept is a simple series connection of the n LIA-S common circuits with slightly increasing values of FPMH and B dependant of n.

The equations are:

MTTR = 0.5 hour

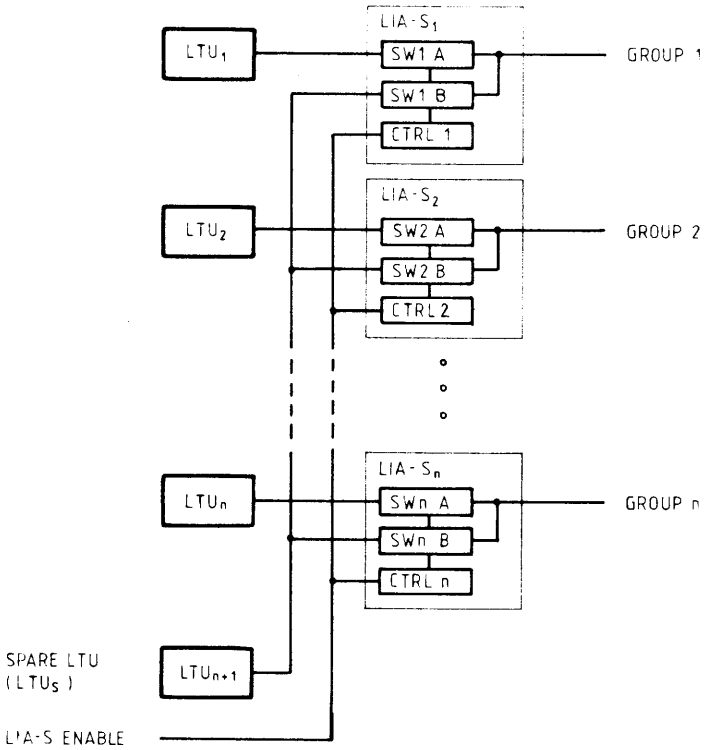
LAMBDA = n(0.28+(n+1)0.0014) FPMH

B = n(0.14+(n+1)0.0008) 10^6

where n can be a maximum of 16.

This results in the following values:

LTU Model	LAMBDA	B	MTTR	RELIABILITY
8+1 spare	2.3 FPMH	$1.2/10^6$	0.5 hour	99.99988%
16+1 spare	4.9 FPMH	$2.5/10^6$	0.5 hour	99.99975%



N out of N+1 LTU's
ELECTRICAL BLOCKDIAGRAM
FIGURE 12.5.3.1-1

The LIA-S is a simple and thus highly reliable element. In essence it consists of a multipole single throw FET switch and a small control circuit. The FET switch switches all the communication lines connected to a LTU at the same time. Either the communication lines are connected through switch section A of the LIA-S to the normal LTU I/O circuitry or through switch section B to the spare LTU I/O circuitry.

Seen from a RMA point of view the LIA-S model looks somewhat different. The LIA-S control circuit and some other common circuitry like I/O connectors and solder joints must always function properly. If the group is connected to the normal LTU, switch A must not fail open and switch B must not fail shorted to maintain the function. Likewise when the group is connected to the spare LTU switch A must not fail shorted and switch B must not fail open. The failure rate of one half open or the other half shorted equals the failure rate of one switch. The failure rate of the LIA-S is:

$$LAMBDA_{LIA\text{-}S} = LAMBDA_{COMMON} + LAMBDA_{SWITCH}$$

$$LAMBDA_{LIA\text{-}S} = 0.28 + 1.87 \text{ FPMH.}$$

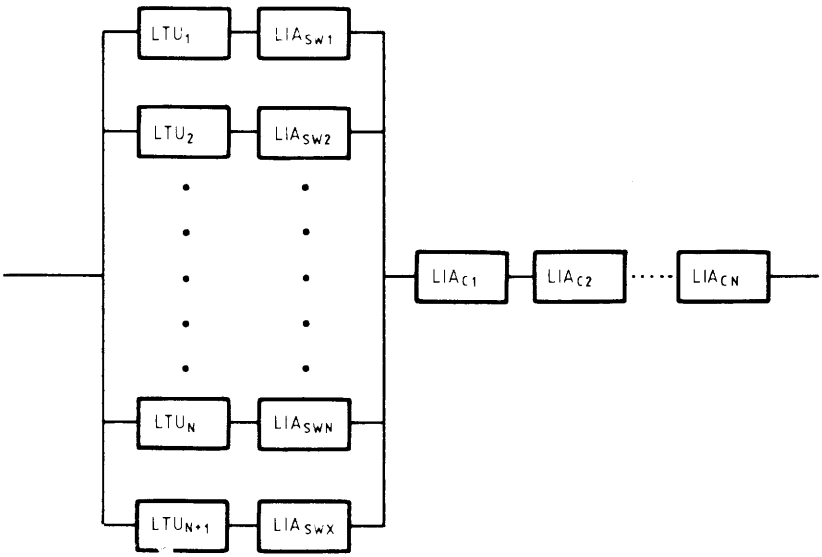
In the RMA model each LTU is in series with the corresponding LIA-S switch section. The spare LTU is in series with the other seciton of the LIA-S switch. All these branches are in parallel.
The LIA-S common circuits are then in series with this parallel group.

The resultant LTU/LIA-S RMA model is shown in figure 12.5.3.1-2.

The operational conditions for this LTU/LIA-S RMA example is that all n groups of lines shall be functioning. This means that n out of n+1 LTU's shall be operable.
Furthermore all the n LIA-S common circuitry shall be operable.

The MTTR values used in the example are:

$MTTR_A$ = 0.5 h, LTU and LIA-S common.
$MTTR_B$ = 12 h , LIA-S switch.



N out N+1 LTU's
RMA BLOCK DIAGRAM
Figure 12.5.3.1-2

The different branches of the RMA model are calculated individually and then combined to the final RMA model. This is done by using the different calculation rules involving LAMBDA, MTTR, Availability (A), and Unavailability (B).

1. LTU in series with LIA-S switch

$$LAMBDA_1 = LAMBDA_{LTU} + LAMBDA_{SW}$$

$$B_1 = LAMBDA_{LTU} * MTTR_{LTU} + LAMBDA_{SW} * MTTR_{SW}$$

$$MTTR_1 = \frac{B_1}{LAMBDA_1} = \frac{LAMBDA_{LTU}}{LAMBDA_1} * MTTR_{LTU} + \frac{LAMBDA_{SW}}{LAMBDA_1} * MTTR_{sw}$$

Inserting the actual values:

$LAMBDA_{LTU} =$	36.9 FPMH
$LAMBDA_{SW} =$	1.87 FPMH
$LAMBDA_C =$	0.28 FPMH
$MTTR_{LTU} = MTTR_A =$	0.5 hour
$MTTR_{SW} = MTTR_B =$	12 hours

results in:

$LAMBDA_1 =$	38.8 FPMH
$MTTR_1 =$	0.9 hour
$B_1 =$	$40.9/10^6$

2. n out of (n+1) LTU/LIA-S

The n+1 LTU/LIA-S series elements are in operational parallel. Assuming negligle recovery time when switching between the elements the normal equation for n out of (n+1) operational redundancy is used:

$$LAMBDA_2 = n(n+1)(LAMBDA_1)^2 * MTTR_1 * 10^{-6} \text{ FPMH}$$

$$B_2 = \binom{n+1}{2} * B_1{}^2 = \frac{n(n+1)}{2} * (B_1)^2$$

$$MTTR_2 = \frac{B_2}{LAMBDA_2} = \frac{MTTR_1}{2}$$

3. LIAS-S Common circuits

The n LIA-S common circuits are in series.

$$LAMBDA_3 = n * LAMBDA_C = n * 0.28 \text{ FPMH}$$

$$MTTR_3 = MTTR_A = 0.5 \text{ hour}$$

$$B_3 = LAMBDA_3 * MTTR_3 = n * B_C = n * 0.14/10^6$$

4. Resultant LTU/LIA-S model

The two branches calculated in 2 and 3 above are in series giving til final result:

$$LAMBDA = LAMBDA_2 + LAMBDA_3$$

$$LAMBDA = n(LAMBDA_C + (n+1)(LAMBDA_1)^2 * MTTR_1 * 10^{-6})$$

$$LAMBDA = n(0.28 + (n+1)\ 0.0014) \text{ FPMH}$$

$$B = B_2 + B_3$$

$$B = n\ (B_C + \frac{n+1}{2} (B_1)^2)$$

$$B = n(0.14 + (n+1)0.0008)/10^6$$

---

note: n can maximum be 16.

$$MTTR = \frac{B}{LAMBDA}$$

$$MTTR = 0.5 \text{ hour}$$

The above 3 equations are used to calculate the RMA figures of an CU-Crate with n out of n+1 redundancy using 8 LTU's + 1 spare and 16 LTU's + 1 spare respectively. The results are summarized below:

LTU/LIA-S MODEL	LAMBDA	B	MTTR	AVAILABILITY
8+1 spare	2.3 FPMH	$1.2/10^6$	0.5 hours	99.99988%
16 + 1 spare	4.9 FPMH	$2.5/10^6$	0.5 hour	99.99975%

The system availability and failure rate will be slightly degraded because of the small probability of having an undetected failure in the unused redundant elements of the system during the regular preventive maintenance periods, and at the same time have a failure in the rest of the items.

The added unavailability because of multiple failures equals:

$$B = LAMBDA_T * MTTR_A * LAMBDA_R * MTTR_C$$

where

$LAMBDA_T$ equals the sum of all elements exclusive of the redundant elements.

$MTTR_A$ equals the element MTTR

$LAMBDA_R$ equals the sum of all redundant elements

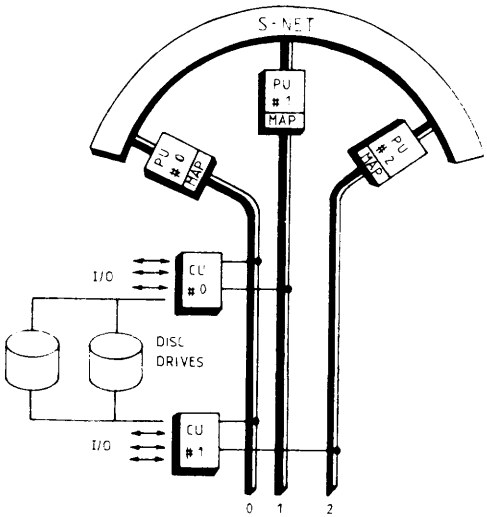
$MTTR_C$ equals the preventive maintenance period, for instance one week = 168 hours.

The MTTR will not be changed, and the failure rate is therefore:

$$LAMBDA = \frac{B}{\overline{MTTR}_A}$$

12.5.3.2 CR80 FATOM

This example demonstrates the RMA calculations on a simplified CR80 FATOM system with 3 PU's, 2 CU's, and 2 Disc Drives. This example assumes negligible switching and recovery time between the redundant elements. Furthermore, the Maintenance and Configuration Processor (MCP) are not made part of the model. The CR80 FATOM configuration is shown below.



DATACHANNEL
CR80 FATOM

The calculations demonstrates a system MTBF in excess of 8 years.

The PU and CU configurations are shown in figures 12.5.3.2-1 and 12.5.3.2-2. The PU consists of 3 x CPU/CACHE and 384K RAM. The CU contains the required number of LTU/LIA-S and a DISC CTRL/DCA. Two CU configurations are used:

8 + 1 spare LTU/LIA-S groups, table 12.5.3.2a
16 + 1 spare LTU/LIA-S groups, table 12.5.3.2b.

The RMA values are calculated using the form described in table 12.5.3.4a of section 12.5.3.4 and the values from table 12.4a section 12.4.

ITEM NO.	ITEM DESCRIPTION	M of N Req'd	MTBF (hours) each	λ (fpm) each	MTTR (hour)	λ equiv (fpm)	$\bar{A}$ equiv x 10^{-6} M of N	U_S M of N (hours)	D_S M of N (hours)	AVAILABILITY (A) (Equiv)	NOTES
1	MFD	2of3		1.6	0.5	-	-				NEGLIGIBLE
2a	CPU-CACHE	3		39	0.5						
2b	RAM	3		58.8	0.5						
2c	MAP	1		52	0.5						
2d	MIA	1		12	0.5						
2e	STI	1		41	0.5						
2f	SBA	1of2		12	0.5	-	-				NEGLIGIBLE
2g	PS	1		36	0.5						
2h	MBT	4		3.5	0.5						
2i	CIA	2		14	0.5						LOCATED IN CU-CRATES
2j	PU-CRATE	1		5	1.0						
2	PROCESSOR UNIT	2of3		481	0.5	0.7	0.17				
3	DISC DRIVE	1of2		125	1.5	0.05	0.04				
4a	DISC CTRL	1		33	0.5						
4b	DCA	1		21.3	0.5						
4	DISC INTERFACE	1of2		54.3	0.5	0.003	0.001				

TABLE 12.5.3.2a
SHEET 1 of 2

CR80 FATOM RMA EXAMPLE
TWO CU'S EACH 8+1 Spare LTU/L1A-S

300-157

ITEM NO.	ITEM DESCRIPTION	M of N Req'd	MTBF (hours) each	λ (fpm) each	MTTR (hour)	λ equiv (fpm)	$\bar{A}$ equiv x 10^{-6} M of N	U_S M of N (hours)	D_S M of N (hours)	AVAILABILITY (A) (Equiv)	NOTES
5a	MBT	1of2		3.5	0.5	-	-				
5b	PS	1of2		36	0.5	0.001	-				
5c	CU CRATE, Basic	1		1.4	1.0						
5	CU CRATE	2		1.4	1.0	2.8	2.8				
6	LTU/LIA-S. 2 CU's										SEE SECTION 12.5.3.1
	EACH 8+1 Spare	2		2.3	0.5	4.6	2.4				
	SYSTEM				0.7	8.1	5.4			99.99946%	

TABLE 12.5.3.2a CR80 FATOM RMA EXAMPLE
SHEET 2 of 2 TWO CU'S EACH 8+1 Spare LTU/LIA-S

300-157

ITEM NO.	ITEM DESCRIPTION	M of N Req'd	MTBF (hours) each	λ (fpm) each	MTTR (hour)	λ equiv (fpm)	$\bar{A}$ equiv x 10^{-6} M of N	U_S M of N (hours)	D_S M of N (hours)	AVAILABILITY (A) (Equiv)	NOTES
1	MFD	2of3		1.6	0.5	-	-				NEGLIGIBLE
2a	CPU-CACHE	3		39	0.5						
2b	RAM	3		58.8	0.5						
2c	MAP	1		52	0.5						
2d	MIA	1		12	0.5						
2e	STI	1		41	0.5						
2f	SBA	1of2		12	0.5	-	-				NEGLIGIBLE
2g	PS	1		36	0.5						
2h	MBT	4		3.5	0.5						
2i	CIA	2		14	0.5						LOCATED IN CU-CRATES
2j	PU-CRATE	1		5	1.0						
2	PROCESSOR UNIT	2of3		481	0.5	0.7	0.17				
3	DISC DRIVE	1of2		125	1.5	0.05	0.04				
4a	DISC CTRL	1		33	0.5						
4b	DCA	1		21.3	0.5						
4	DISC INTERFACE	1of2		54.3	0.5	0.003	0.001				

TABLE 12.5.3.2b
SHEET 1 of 2

CR80 FATOM RMA EXAMPLE
TWO CU'S EACH 16+1 Spare LTU/L1A-S

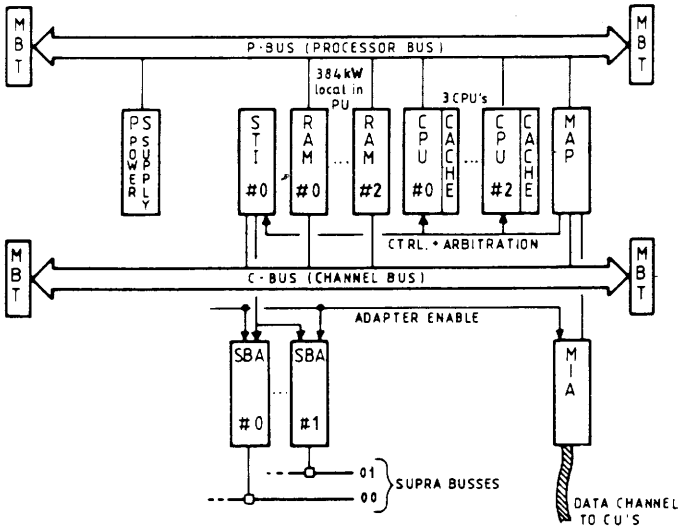
300-157

ITEM NO.	ITEM DESCRIPTION	M of N Req'd	MTBF (hours) each	λ (fpm) each	MTTR (hour)	λ equiv (fpm)	$\bar{A}$ equiv x 10^{-6} M of N	U_S M of N (hours)	D_S M of N (hours)	AVAILABILITY (A) (Equiv)	NOTES
5a	MBT	1of2		3.5	0.5	-	-				
5b	PS	1of2		36	0.5	0.001	-				
5c	CU CRATE, Basic	1		1.4	1.0						
5	CU CRATE	2		1.4	1.0	2.8	2.8				
6	LTU/LIA-S. 2 CU's										SEE SECTION 12.5.3.1
	EACH 16+1 Spare	2		4.9	0.5	9.8	5.0				
	SYSTEM				0.6	13.3	8.0			99.9992%	

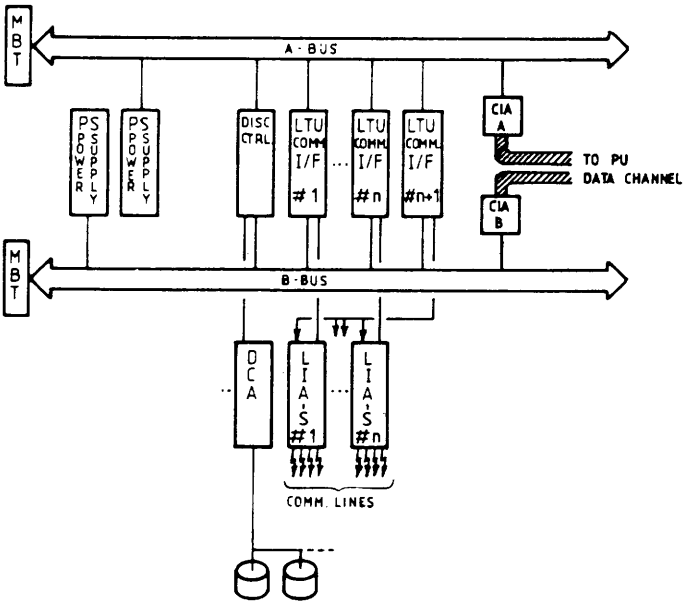
TABLE 12.5.3.2b
SHEET 2 of 2

CR80 FATOM RMA EXAMPLE
TWO CU'S EACH 16+1 Spare LTU/LIA-S

300-157



CR80 FATOM PU CONFIGURATION
FIGURE 12.5.3.2-1

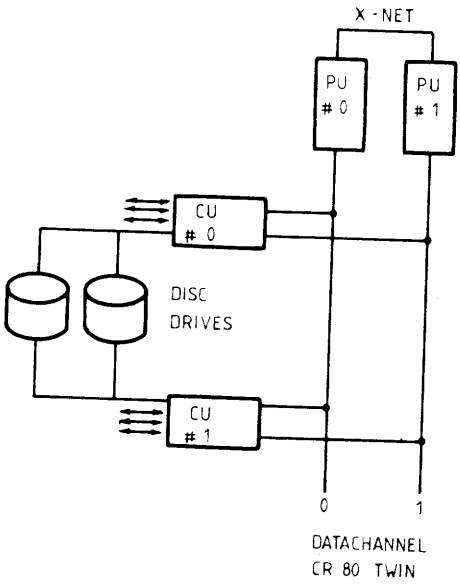


CR80 FATOM CU CONFIGURATION
FIGURE 12.5.3.2-2

12.5.3.3 CR80 TWIN

This example shows the RMA calculations on a basic CR80 TWIN system with 2 PU's, 2 CU's, and 2 Disc Drives. This example assumes negligible switching and recovery time between the redundant elements. Furthermore, the Maintenance and Configuration Processor (MCP) are not made part of the model. The CR80 TWIN configuration is shown below.

CR80 TWIN

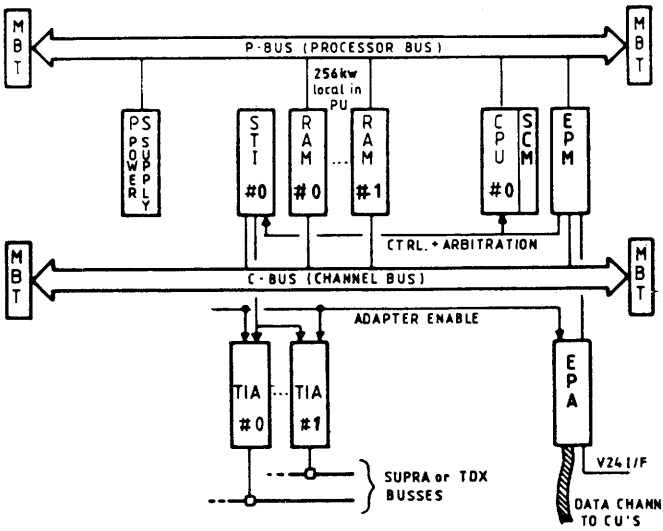


The calculations demonstrates a system MTBF in excess of 9 years.

The PU and CU configurations are shown in figures 12.5.3.3-1 and 12.5.3.3-2. The PU consists of 1 x CPU-SCM and 256K RAM. The CU contains the required number of LTU/LIA-S and a DISC CTRL/DCA. Two CU configurations are used:

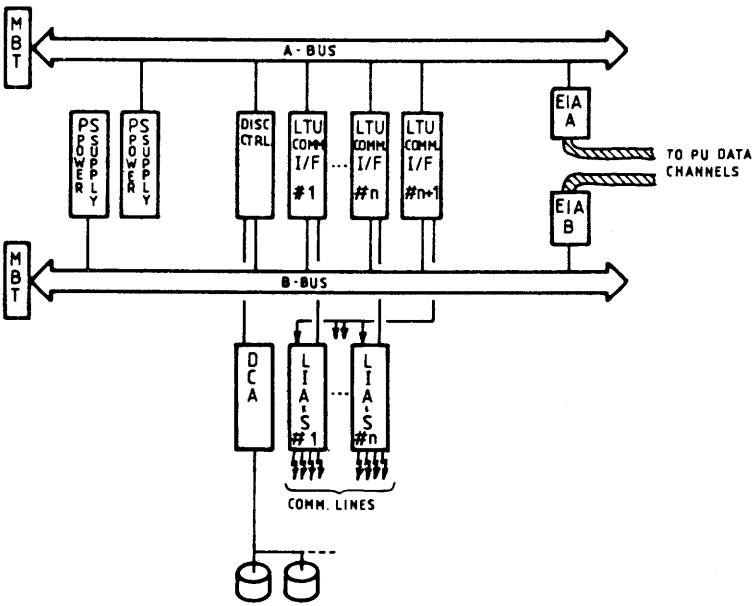
8 + 1 spare LTU/LIA-S groups, table 12.5.3.3a
16 + 1 spare LTU/LIA-s groups, table 12.5.3.3b

The RMA values are calculated using the form described in table 12.5.3.4a of section 12.5.3.4 and the values from table 12.4a, section 12.4.



CR80 TWIN PU CONFIGURATION

FIGURE 12.5.3.3-1



CR80 TWIN CU CONFIGURATION

FIGURE 12.5.3.3-2

ITEM NO.	ITEM DESCRIPTION	M of N Req'd	MTBF (hours) each	λ (fpm) each	MTTR (hour)	λ equiv (fpm)	$\bar{A}$ equiv x 10^{-6} M of N	U_S M of N (hours)	D_S M of N (hours)	AVAILABILITY (A) (Equiv)	NOTES
1	MFD	1of2		1.6	0.5	-	-				NEGLIGIBLE
2a	CPU-SCM	1		27.4	0.5						
2b	RAM	2		58.8	0.5						
2c	EPM	1		5.8	0.5						
2d	EPA	1		3.9	0.5						
2e	STI	1		40.3	0.5						
2f	TIA	1of2		8.5	0.5	-	-				NEGLIGIBLE
2g	PS	1		36	0.5						
2h	MBT	4		3.5	0.5						
2i	EIA	2		8.8	0.5						LOCATED IN CU-CRATES
2j	PU-CRATE	1		5	1.0						
2	PROCESSOR UNIT	1of2		270.8	0.5	0.07	0.02				
3	DISC DRIVE	1of2		125	1.5	0.05	0.04				
4a	DISC CTRL	1		33	0.5						
4b	DCA	1		21.3	0.5						
4	DISC INTERFACE	1of2		54.3	0.5	0.003	0.001				

TABLE 12.5.3.3a CR80 TWIN RMA EXAMPLE
SHEET 1 of 2 TWO CU'S EACH 8+1 Spare LTU/LlA-S

300-157

ITEM NO.	ITEM DESCRIPTION	M of N Req'd	MTBF (hours) each	λ (fpm) each	MTTR (hour)	λ equiv (fpm)	$\bar{A}$ equiv x 10^{-6} M of N	U_S M of N (hours)	D_S M of N (hours)	AVAILABILITY (A) (Equiv)	NOTES
5a	MBT	1of2		3.5	0.5	-	-				
5b	PS	1of2		36	0.5	0.001	-				
5c	CU-CRATE, Basic	1		1.4	1.0						
5	CU-CRATE	2		1.4	1.0	2.8	2.8				
6	LTU/LIA-S. 2 CU's										SEE SECTION 12.5.3.1
	EACH 8+1 Spare	2		2.3	0.5	4.6	2.4				
	SYSTEM				0.7	7.6	5.3			99.99947%	

TABLE 12.5.3.3a
SHEET 2 of 2

CR80 TWIN RMA EXAMPLE
TWO CU'S EACH 8+1 Spare LTU/LIA-S

300-157

ITEM NO.	ITEM DESCRIPTION	M of N Req'd	MTBF (hours) each	λ (fpm) each	MTTR (hour)	λ equiv (fpm)	$\bar{A}$ equiv x 10^{-6} M of N	U_S M of N (hours)	D_S M of N (hours)	AVAILABILITY (A) (Equiv)	NOTES
1	MFD	1of2		1.6	0.5	-	-				NEGLIGIBLE
2a	CPU-SCM	1		27.4	0.5						
2b	RAM	2		58.8	0.5						
2c	EPM	1		5.8	0.5						
2d	EPA	1		3.9	0.5						
2e	STI	1		40.3	0.5						
2f	TIA	1of2		8.5	0.5	-	-				NEGLIGIBLE
2g	PS	1		36	0.5						
2h	MBT	4		3.5	0.5						
2i	EIA	2		8.8	0.5						LOCATED IN CU-CRATES
2j	PU-CRATE	1		5	1.0						
2	PROCESSOR UNIT	1of2		270.8	0.5	0.07	0.02				
3	DISC DRIVE	1of2		125	1.5	0.05	0.04				
4a	DISC CTRL	1		33	0.5						
4b	DCA	1		21.3	0.5						
4	DISC INTERFACE	1of2		54.3	0.5	0.003	0.001				

TABLE 12.5.3.3b
SHEET 1 of 2

CR80 TWIN RMA EXAMPLES
TWO CU'S EACH 16+1 Spare LTU/LIA-S

300-157

ITEM NO.	ITEM DESCRIPTION	M of N Req'd	MTBF (hours) each	λ (fpm) each	MTTR (hour)	λ equiv (fpm)	$\bar{A}$ equiv x 10^{-6} M of N	U_S M of N (hours)	D_S M of N (hours)	AVAILABILITY (A) (Equiv)	NOTES
5a	MBT	1of2		3.5	0.5	-	-				
5b	PS	1of2		36	0.5	0.001	-				
5c	CU-CRATE, Basic	1		1.4	1.0						
5	CU-CRATE	2		1.4	1.0	2.8	2.8				
6	LTU/LIA-S. 2 CU's										SEE SECTION 12.5.3.1
	EACH 16+1 Spares	2		4.9	0.5	9.8	5.0				
	SYSTEM				0.6	12.8	7.9			99.99921%	

TABLE 12.5.3.3b
SHEET 2 of 2

CR80 TWIN RMA EXAMPLE
TWO CU'S EACH 16+1 Spare LTU/LIA-S

300-157

12.5.3.4 RMA Calculation Sheets

As an aid in RMA calculations form 300-157 can be used, see table 12.5.3.4a overleaf.

The first three columns are filled in from the RMA model. The values for MTBF, LAMBDA and MTTR are found in table 12.4a.
Using the appropriate calculation rules are calculated the resultant LAMBDA, Unavailability (A = B), MTTR and Availability (A).

ITEM NO.	ITEM DESCRIPTION	M of N Req'd	MTBF (hours) each	λ (fpm) each	MTTR (hour)	λ equiv (fpm)	$\bar{A}$ equiv x 10^{-6} M of N	U_S M of N (hours)	D_S M of N (hours)	AVAILABILITY (A) (Equiv)	NOTES

300-157

TABLE 12.5.3.4a RMA CALCULATION FORM
SHEET 1 of 1

NOTES:

w from the CR80 module production floor.

CR80 Computer integration- and test floor.

13 CR80 APPLICABLE SPECIFICATIONS AND STANDARDS

13.1 Environmental Specifications

Definitions

- Operating: These limits apply to equipment installed as specified and operating in a normal office or computer room environment.
- Storage: These limits apply to equipment properly packed and protected against dust, moisture, condensed water etc.
- Transportation: These limits apply to equipment properly packed for shipment.

Temperature

- Operating: $15^{o}C$ to $32^{o}C$
  Maximum rate of change
  $6^{o}C$ per hour.
- Storage/Transportation: $-40^{o}C$ to $70^{o}C$.

Humidity

- Operating: 20%RH to 80%RH non condensing. Maximum rate of change 10%RH per hour.

  Absolute water content in the room air shall be limited to 22g water per cubic meter of air.

- Storage/Transportation: 10%RH to 90%RH non condensing.

Altitude

- Operating: 0 to 2000 m
- Storage/Transportation: 0 to 10.000 m

Vibration

- Operating and Storage: 5Hz to 50Hz constant displacement of 0.02mm.

  50Hz smooth crossover
  50Hz-350Hz constant acceleration 0.2g.
- Transportation: 5Hz to 350Hz constant acceleration 1.5g.

Shock

- Operating and Storage: 1g, half sine wave, 10ms duration. Not to be repeated more often than one per 10 seconds.
- Transportation: 25g, half sine wave, 10ms duration.

Electrical Emmission

Radiated:	Conforming to VDE871 class C and VDE875 class G
Conducted:	Conforming to VDE875 class G.

Electrical Susceptibility

Radiated:	Electromagnetic Field Strength less than 1V/m with frequencies from 30 MHz to 500 MHz induced from distance of 3m.
Conducted:	Noise pulses on main wires with amplitude less than 1000V and risetime longer than 35n. Pulse duration 0.1 uS to 10 uS. Repetition rate not more than one per second.

13.2 Design, Test and Production Procedures and Specifications

13.2.1 CR80 Quality Assurance

AQAP1 NATO Quality Control System Requirements for Industry.
AQAP6 NATO Calibration System Requirements for Industry.
AQAP13 NATO Software Quality Control System Requirements (Amended as applicable for CR80 22/12 1980).

QA/PLN/0001	Quality Assurance Plan
QA/PLN/0002	Reliability Plan
QA/PLN/0003	Parts and Material Plan
QA/PLN/0004	Quality Control Plan
QA/PLN/0005	Configuration Management Plan
CSD/006/PLN/0001	Software Configuration Management Plan
QA/INSR/0002	Instruction in using of tags for inspection status (receiving)
QA/INSR/0003	Instruction in using of tags for inspection status (in-process)
QA/INSR/0005	Instruction in vendor control
QA/INSR/0006	Instruction in modification of printed circuit boards
QA/INSR/0007	Instruction in receiving inspection of mechanical parts
QA/INSR/0008	Instruction in nonconformance control
QA/INSR/0009	Instruction in receiving inspection of printed circuit boards
QA/INSR/0010	Instruction in assembly of mechanipal parts for CR80
QA/INSR/0011	Instruction in taken samples for receiving inspection

QA/INSR/0012	Design control
QA/INSR/0014	Visual inspection of racks, fans etc.
QA/INSR/0015	Instruction in In-Process Inspection
QA/INSR/0016	Isolation-and functional test of purchased OEM equipment
QA/PROC/0003	Audit of Quality Assurance System
QA/PROC/0007	Procedure for Raw Materials under age control
QA/PROC/0010	Control of Customer Supplied Materials

13.2.2 CR80 Hardware Design

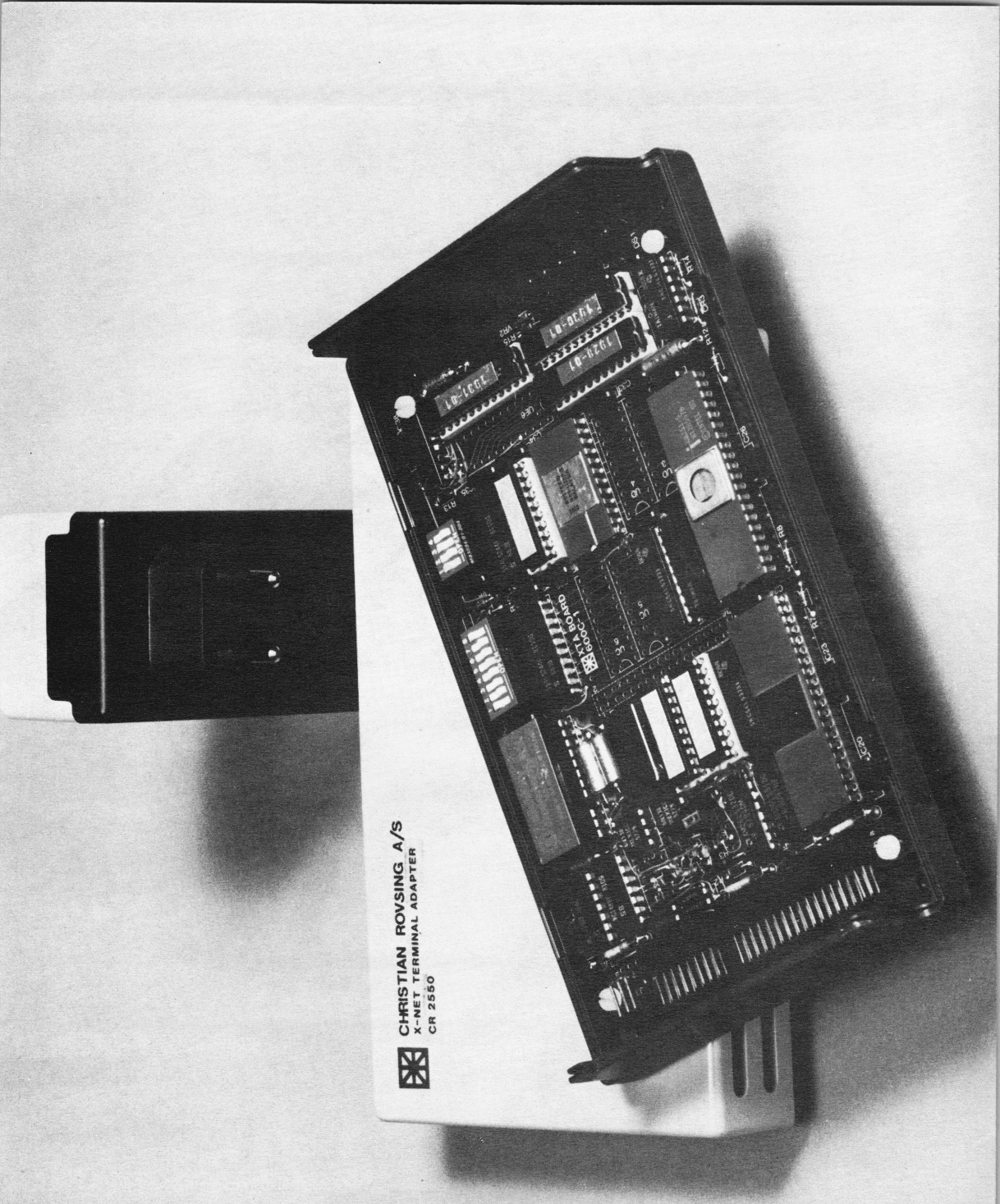
ED/STD/0001	Drawing Standard
ED/STD/0003	Design and Construction Criteria for Electronic Material
ED/SPC/0001	Design Reviews
ED/SPC/0002	Standard CR80 Environmental Limits
CSD/EPR/0003	Structure of Module Parts Lists
CSD/EPR/0004	CR80 Design Guide Lines
CSD/EPR/0005	CR80 Design Guide Lines, Derating
CSD/EPR/0007	Release and Control of Design Documentation
CSD/EPR/0008	Contend and Structure of Technical Manual
CSD/EPR/0009	Internal Configuration Control of Firmware

13.2.3 CR80 Hardware Production

ED/STD/0002	General Manufacturing Requirements
ASD-PROC-202	Manual Soldering

ASD-PROC-204	Cleaning of PWB's
ASD-PROC-205	Exchange of components on PWB
PRO/INSR/0004	Instruction in Purchasing
PRO/INSR/0013	Air Freight shipping
PRO/PROC/0001	Finish specification, yellow chromate passivated zinc
PRO/PROC/0004	ECO procedure
PRO/PROC/0005	Finish specification, Grey painting
PRO/PROC/0008	Silk screen procedure
PRO/PROC/0009	Preservation, packing and labelling

Net Local Area Network interconnects hundreds of terminals and computers via ghspeed coaxial cable within an area of several square kilometers. Here is own the X-Net Wall-Outlet, which is installed where needed, for plugging in rminals and computers to the coaxial cable net.



The X-Net Terminal Adapter (XTA), shown with its top cover removed. The XT based on a few Large Scale Integrated circuits ecomically interfaces virtually a model of asynchroneous terminal to the X-Net high speed Local Area Network. Th XTA is a complete stand-alone unit with its own mains connection requiring r modification of the attached terminal.

14. X-NET LOCAL AREA NETWORK

14.1 INTRODUCTION

X-Net is a Local Area Network (LAN) produced and delivered since 1978, for connecting up to several hundred terminals (VDU, printers etc.) with each other, and one or more small and large computers. X-Net based on the TDX Bus described in chapter 8, eliminates the presently used separate circuits and cables to each terminal and replaces these with a single pair of cables, which is common to all data equipment:
terminals, computers, text processing equipment etc. attached to the X-Net.

This makes it possible, within one or more buildings, to place X-Net outlets on the walls of every room, as presently done for telephone and power installations. It is therefore not necessary to have fixed distribution and placement of data equipment, as these can be moved freely between all rooms.

A very important aspect of the X-Net is the capability for one terminal to work with any computer or other data equipment connected to the X-Net, instead of being able only to work with a single computer in the traditional direct connection scheme.

The X-Net also allows for future extensions and changes in the terminal and computer installation, without having to move and install new cables to terminals. Mainframe and Terminal manufacturer independence is achieved in that PORTS to the X-Net can provide conversion between different interfaces, Protocols and access-methods of Computers, Terminals and Communication lines, thereby as an unique feature makes distributed front-ending and protocol conversion possible.
Installation of X-Net cable and X-Net Wall outlets in new or old buildings is very simple and comparable with installation of telephone outlets.

S-Net is a superstructure on the basic X-Net building blocks and interconnects up to 8 X-Net Local Area Networks, with full connectivity between all terminals and computers, and without impact on traffic speed or response times. Coupling of X-Nets via S-Net extends the number of terminals and computers handled to several thousands.

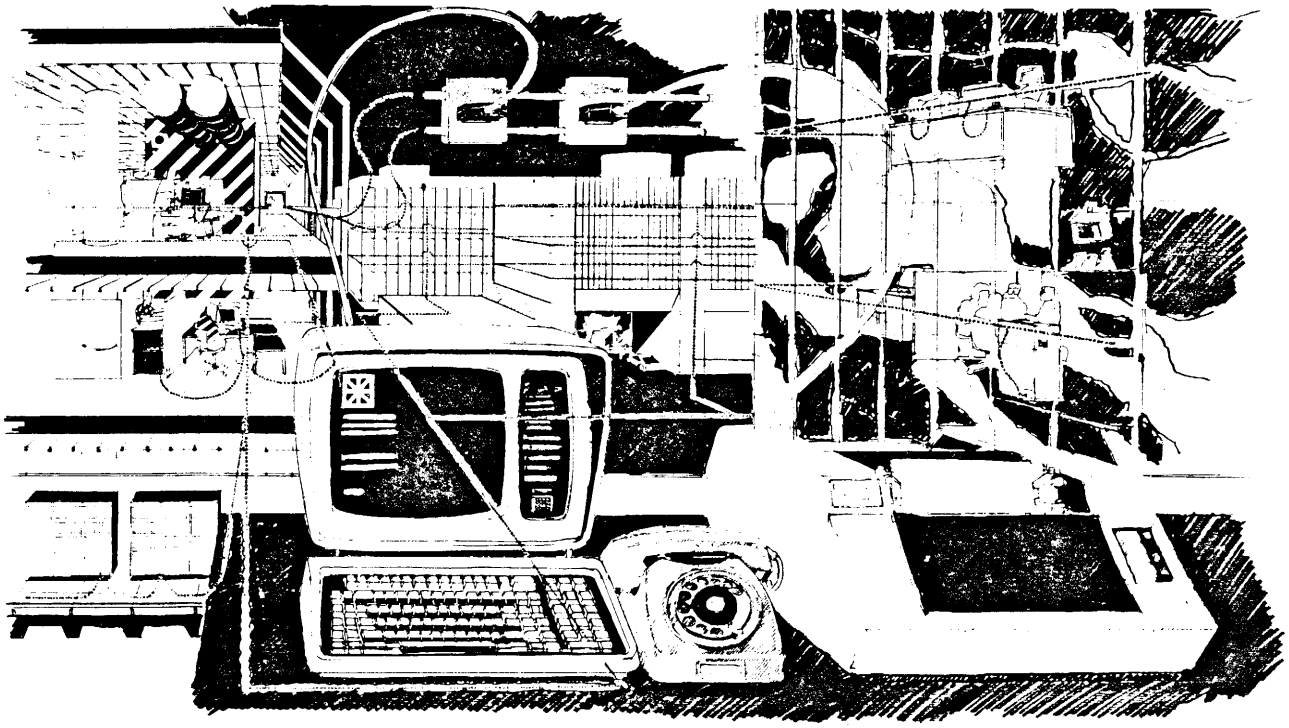


Figure 14.1-1 FLEXIBLE INTERCOMMMUNICATION BETWEEN COMPUTERS, MAINFRAMES AND TERMINALS VIA X-NET

14.2 X-Net Functional Overview

X-Net (multipleXed local area Net) shown in figure (14.2-1) below is an extremely fast shared digital link between terminals (VDU's, printers etc.) and one or more hosts. The transmission media is the TDX-bus, described in Chapter 8. Utilizing two screened twisted pair coaxial cables, data is transported in packets (HDLC) between terminals and host computers at a data rate of nearly 2 Megabit, divided into 6400 timeslots each second of each 288 bit, of which 128 (16 bytes) are utilized for data to or from terminals, and the remaining for routing, control, automatic error detection and correction (CRC check). One cable is utilized for data transport to terminals and the other for data transport from terminals.

Up 8 X-NETs can be coupled together via the S-NET* (described in Chapter 8) see figure (14.2-2), the 16 Megabit datarate of S-NET ensures transport of packets between the X-NETs without degradation in speed and connectivity in any of the connected X-NETs.

* S-Net is standard for CR80 Computers, described in chapter 2, announcement for X-Net in 1982

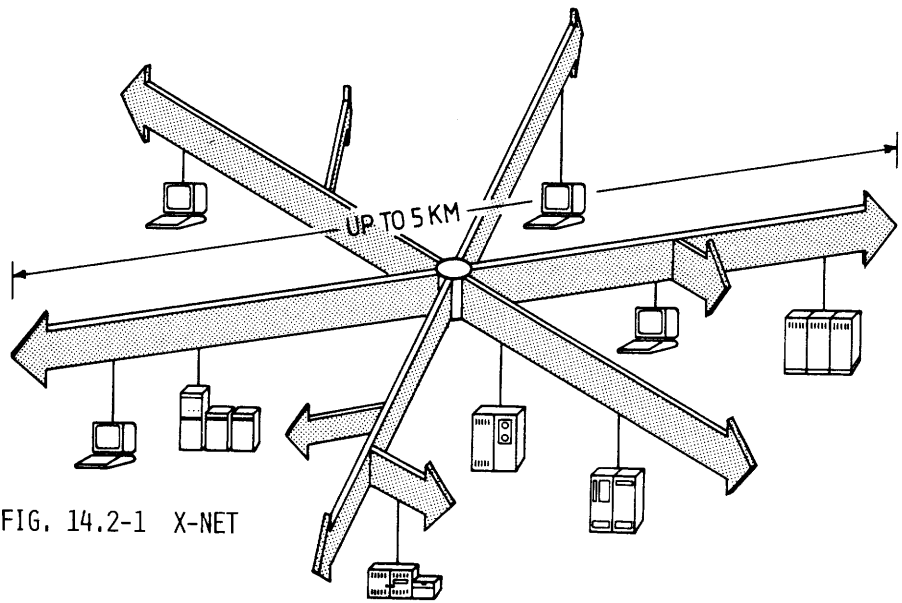


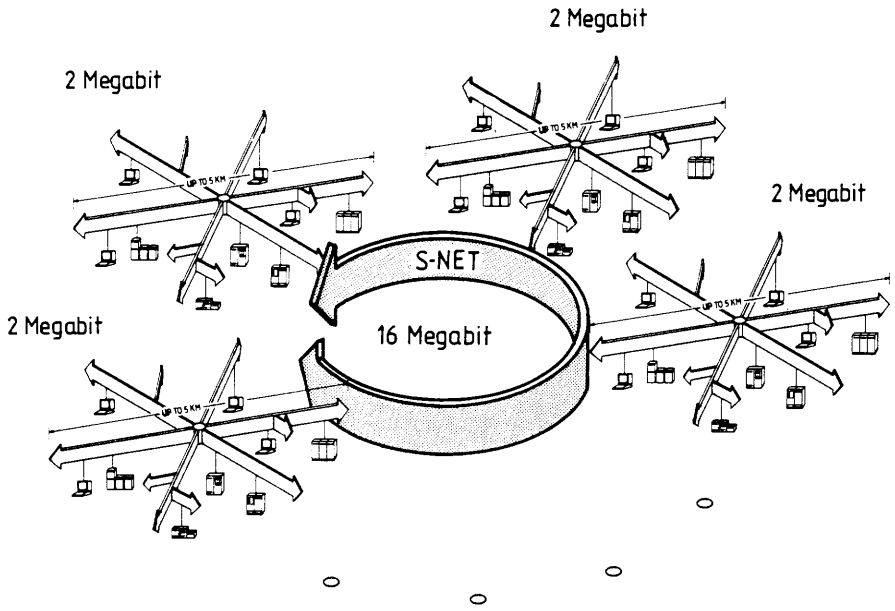
FIG. 14.2-1 X-NET

A terminal is allocated a number of timeslots each second on the TDX Bus, corresponding to its baudrate. The timeslots allocated to each terminal does not overlap, whereby, it is possible for many terminals to use the common cable (Time divisioned multiplex).

The data rate on the X-Net of a terminal is adaptively and dynamically changed, controlled by the actual datarate of attached terminals and computers, thereby e.g. allowing for high speed transfer of complete VDU-screen and return to low data rate for input from the keyboard.

X-NET consists of one or more controllers (XCT), X-Net cable, Wall Outlets, Computer or Communication Line Ports, and Terminal Adapters.

The Terminal Adapter (XTA) makes it possible to attach existing terminals, printers etc. as well as older terminals can be modified to modern protocols, using the built-in micro-processor and buffer capacity of the X-Net devices.



up to 8 X-nets connected
with full intercommunication
via the 16 Megabit S-NET

FIG. 14.2-2

The Terminal Adapter is connected to, and communicate via the terminals existing serial line interface. Therefore no modification is done to the terminals existing electronics, as well as the connection of the adapter, can be done in a few minutes. By removing the adapter plug, the terminal can be used as normally.

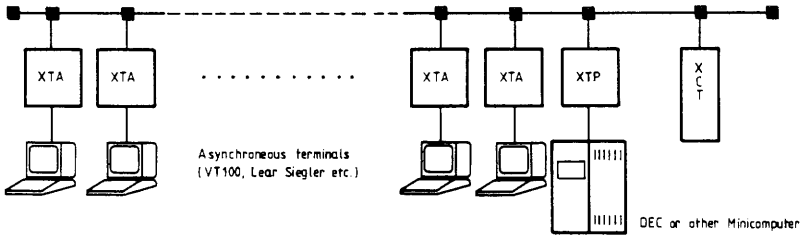
The X-Net Ports (XTP, XEP etc.) provide interface to Computers or Communication Lines via single, multiple or multidropped lines. Also the X-Net Ports provides protocol conversion and terminal emulation between peripherals communicating via the X-Net through the Port, and the Computers or communication lines connected to.

An Amplifier and Branching Unit (XAB) is inserted into the X-Net cable each 800 m to 1.2 km, for each 64 terminals or where a splitting of the cable into two separate cirucits are wanted. The maximum coverage of a single X-Net is a circle 5 kilometers in diameter. For X-Net coupled via S-Net the limitation is a circle up to 10 kilometers in diameter.

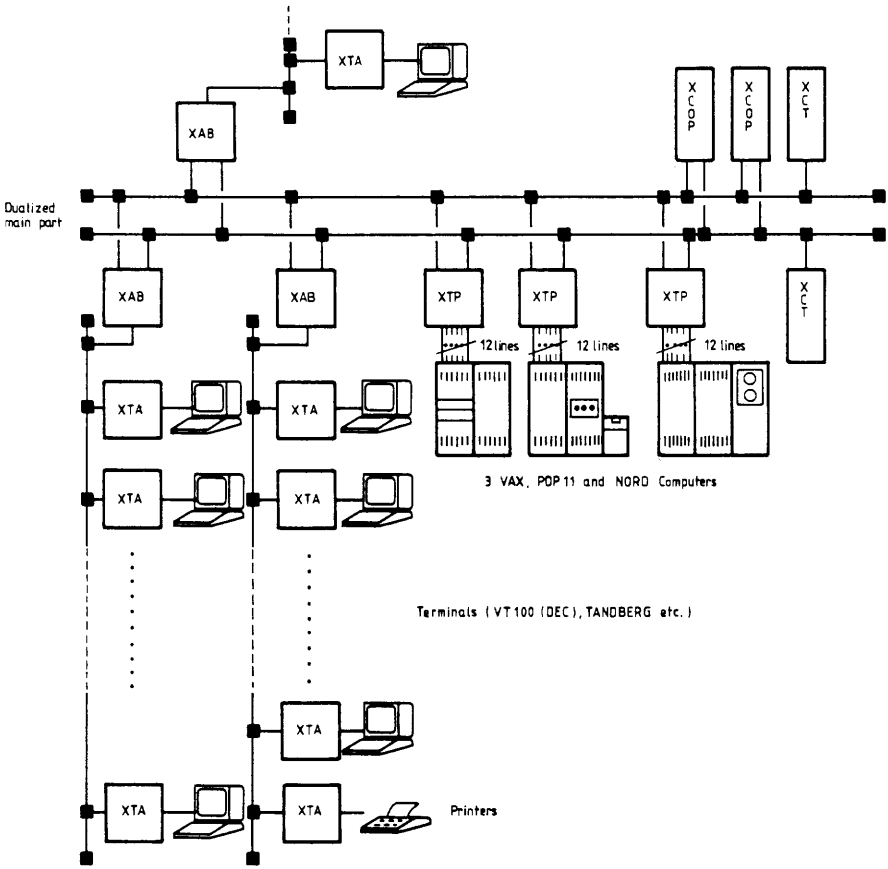
A unique feature of X-Net is that Fault Tolerance is a standard option. Any X-Net system element can have dual X-Net connection and cable sections can be dualized, with automatic fall-back to spare element or cable in case of faults.

X-Net configuration examples of small, medium and large Local Area Networks is found overleaf (Fig. 14.2-3 and 14.2-4).

Figure 14.2-3



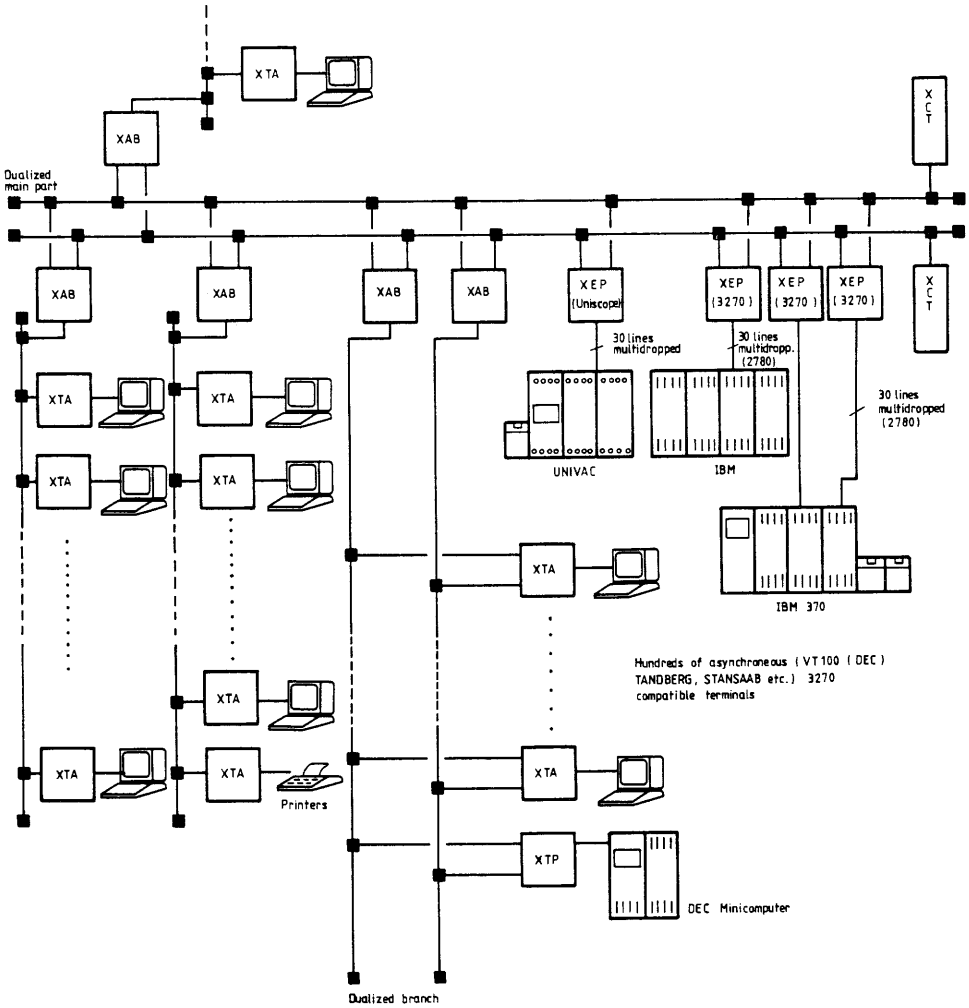
X-Net installation in small Mini-Microcomputer environment.



X-Net installation in Minicomputer environment.

3-3309

Figure 14.2-4



3310

X-Net installation in mixed mainframe and Minicomputer environment

14.2 X-NET LOCAL AREA NETWORK

Below the main characteristics of the X-net local network is given, and overleaf is shown possible attachment to the Network. The following sections describe possible forms of these attachments for discussion, and general configuration and logical connection information.

14.2.1 Physical Specification

- bit rate: 1.8432 Mbit/sec
- Max. inter Station distance - with Amplifier/branching unit: 5 km
  - without Amplifier/branching unit: 2.5 km
- Single X-Net

Maximum number of attached Devices :	254
Maximum number of attached Terminals:	960 [1]
Maximum number of attached Ports	240 [1]
Maximum number of attached Super Ports	12

- Multiple X-Nets (up to 8 nets coupled via Supergateways and S-Net)

Maximum number of attached Devices :	2032
Maximum number of attached Terminals:	7680 [1]
Maximum number of attached Ports	1920
Maximum number of attached Super Ports	96

- Medium: Shielded twisted pair coaxial cable, base band signalling
- Topology: Branching rooted tree
- Data link layer

  Distributed duplex HDLC like protocol an all links with Adaptive/Dynamic allocation of physical X-net station bandwidth
- Packet Protocol

  Variable sized packets (0-64 kbyte), guaranteed delivery
- Fault tolerance

  As standard option, part or all of an X-Net may be dualized

1) The maximum number of attached Ports and Terminals are correlated. See section 14.3.

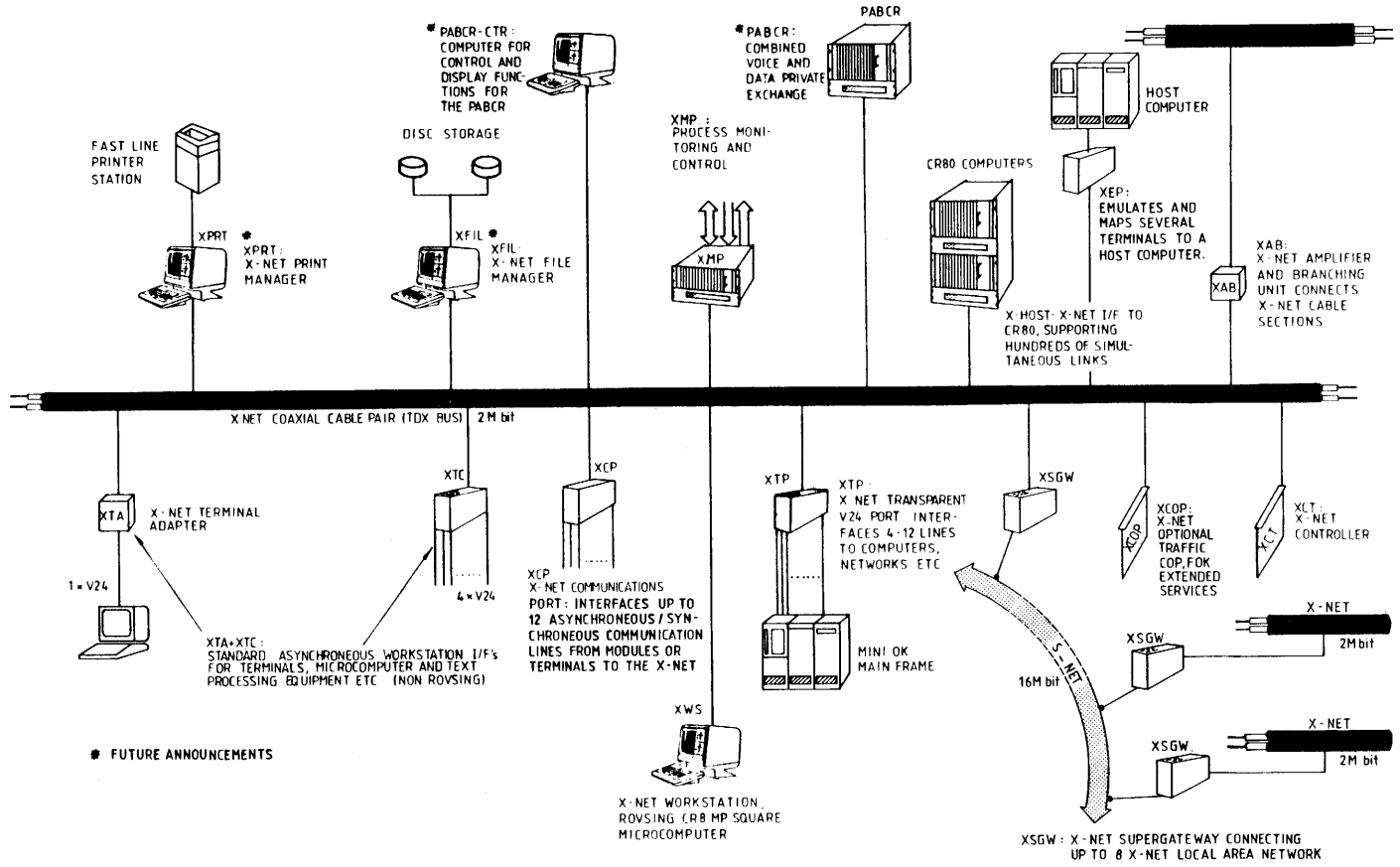


Figure 14.2.1-1 X-NET LOCAL NETWORK

14.2.2 X-NET Configuration

The physical limits that restrict the system designer in configuring X-NET Local Networks are given below. The drawing overleaf shows typical large configuration.
Explanation and calculation of the below fundamental X-Net constraints are found later in this chapter.

1. X-NET cable section is constituted of a coaxial cable pair terminated at each end, maximum coaxial cable length (Lsect.) of a X-Net section, depends on selected cable type:

TM3078	:	Lsect.= 800 meter (recommended type)
RG111A/U or RG22 B/U	:	Lsect.= 1300 meter
RG 108 AU	:	Lsect.= 500 meter

A maximum of 64 X-NET Wall Outlets (XWO) must be attached to any X-NET section. There are no restrictions to distribution of XWO's along the cable and different X-NET sections may use different cable types.

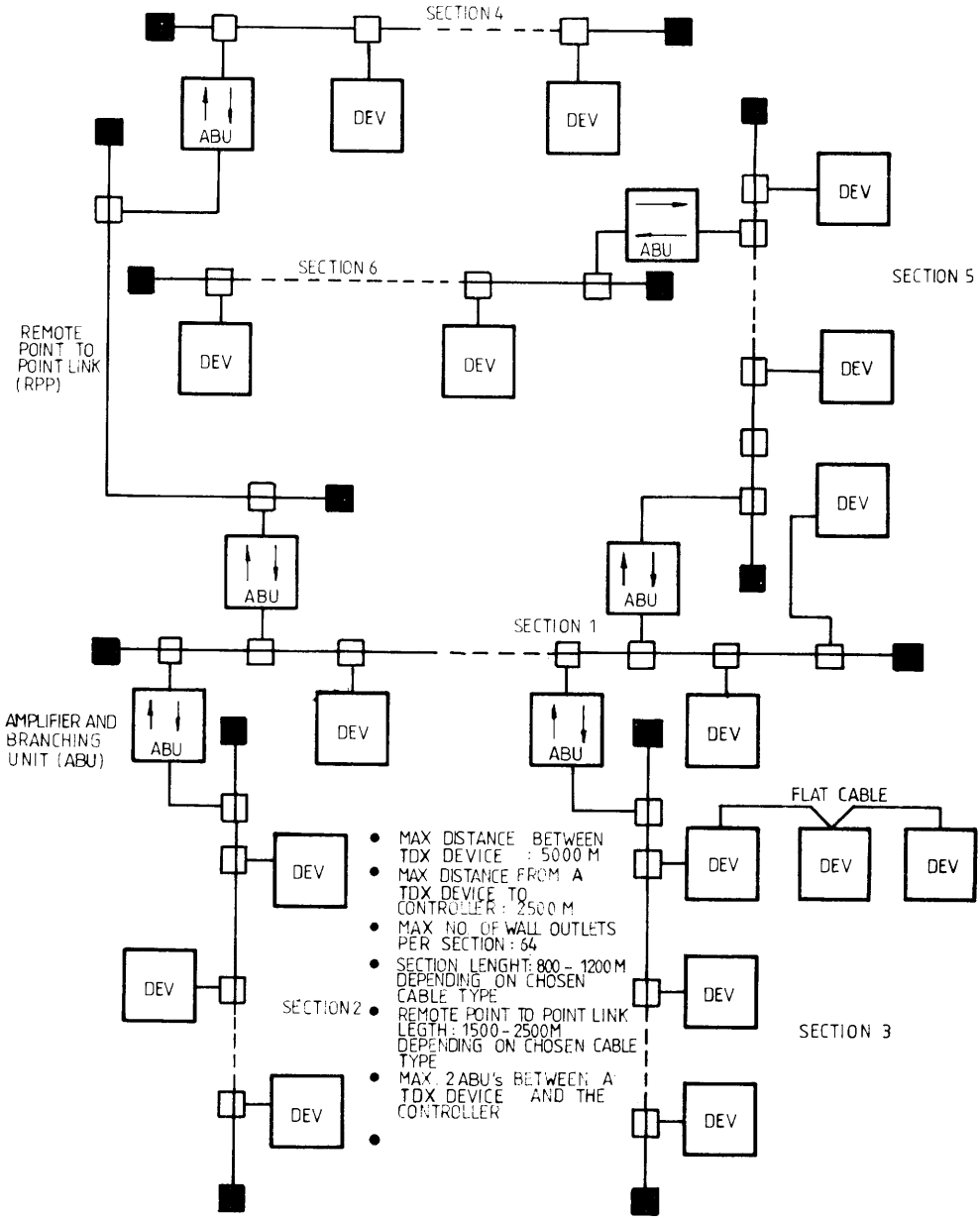
2. Maximum of 2 X-NET Amplifier and Branching Units (XAB) must be attached between the Controller and any X-NET station, this allows for a maximum of 4 XAB's between any two X-NET stations.

3. Maximum 2500 meters of Coaxial cable length (including Point to Point links) between the controller and any X-NET station, this allows for up to 5000 meters between X-NET stations.

4. Point to Point links, used for connecting remote X-NET sections in other buildings or floors within a large building, shall be within the following maximum length:

TM3078	:	1500 meter
RG111 A/U or GR22 B/U	:	2500 meter
RG108 A/U	:	1000 meter

A Point to Point link may use different cable type from the cable types used for the X-Net sections it connects to at each end.

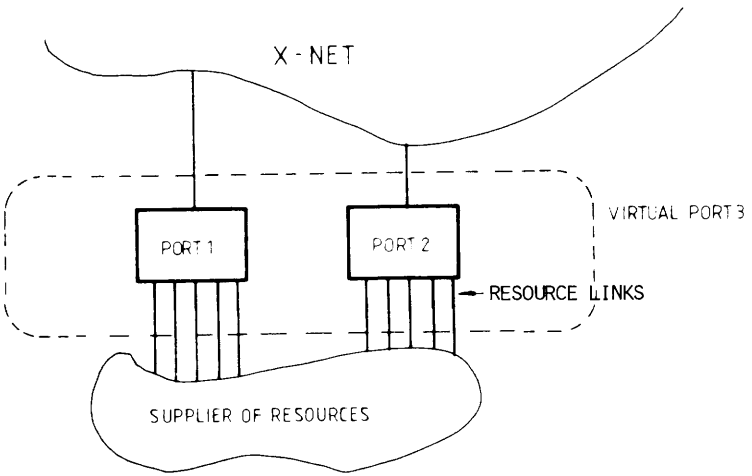


X-NET CONFIGURATION

14.2.3 Network Dialogue - Overview

The X-Net Local Area Network consists of a number of consumers and suppliers interconnected. A Member is a common designation for a consumer or a supplier. A number of members constitutes a Domain. A Domain may optionally be controlled by a Traffic Cop (X-COP). Consumers interfaces to the X-Net via Terminal Adapters (XTA) or Terminal Concentrators (XTC). Suppliers interfaces to the X-Net via Resource Adapters called Ports (e.g. XTP, XCP, XEP, etc.) which again connects to the supplied resources via resource links.

Any consumer may negotiate with any Port in the network to achieve a free resource.



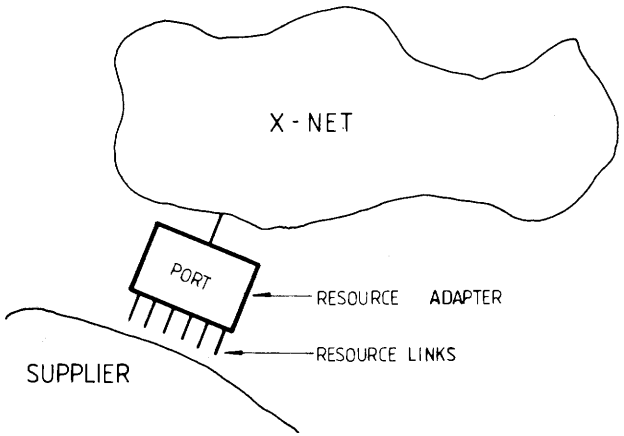
In a basic X-Net without a Traffic Cop (XCOP) only Ports may be addressed, while a network including XCOP is able to address virtual Ports consisting of one or more physical Ports.

Two basic commands establish and dismantle connections in the X-Net:

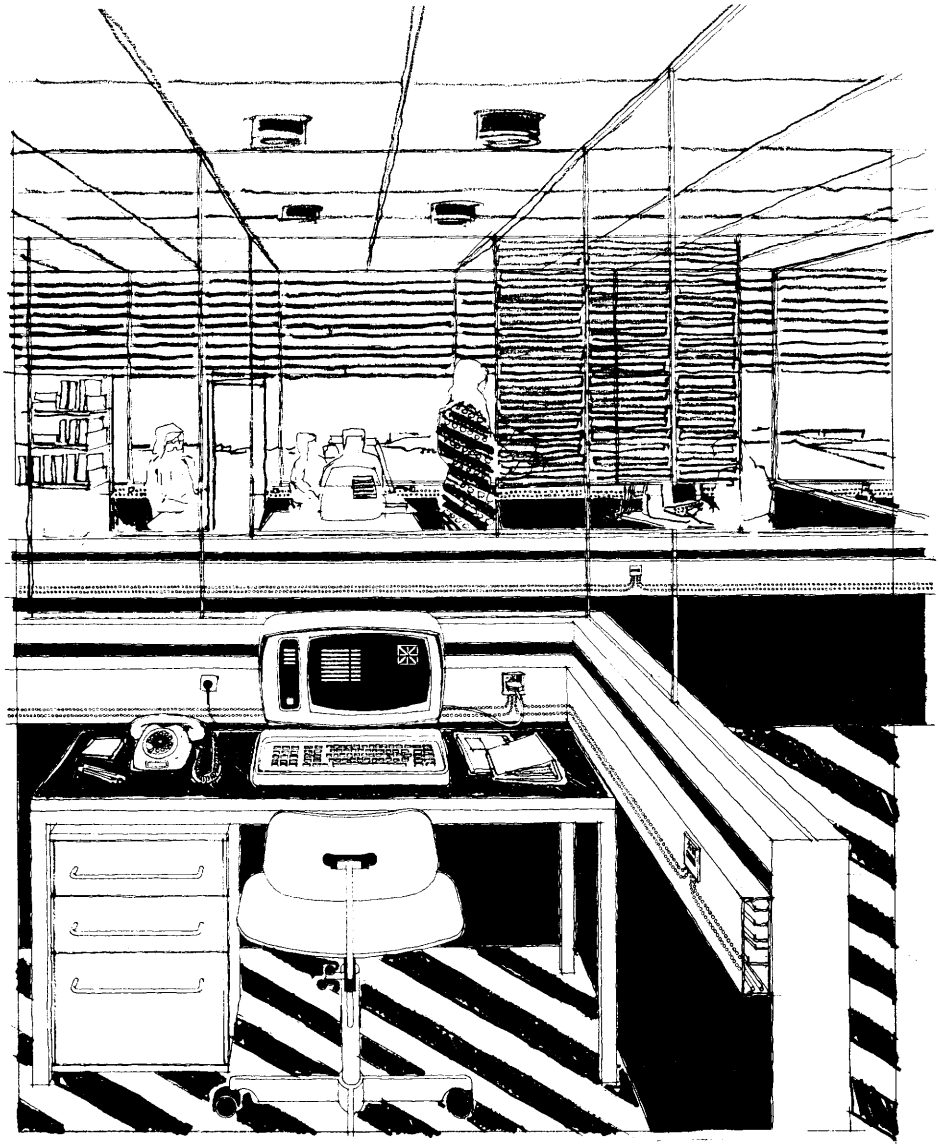
HELLO_<NN><CR>	which establishes a connection to a free resource in PORT NN
BYE _ <CR>	which dismantles a connections

14.2.4 Network Dialogue - Internetwork Communication

The Network Dialogue uses a dedicated logical channel in the X-net. Resource requests from consumers are routed to the resource adapter, where the negotiation and selection of resources takes place.



Traffic-Cops (XCOP) may be inserted in the X-NET to monitor the network dialogue between all or a subset of consumers/suppliers, defined as the COP-domain. The Traffic-Cop may be inserted or removed without impact on already established connections in the network. From insertion-time all network communication between members of the COP-domain, will be transferred via the COP. By removal of the COP the network resumes to the basic network communication again.



X-NET WALL OUTLETS INSTALLED IN OFFICE ENVIRONMENT

14.3 X-NET STATIONS

All X-Net Stations have a unique device-number in the range 0 to 254. The numbers are subdivided in groups:

number	X-Net Stations
0	Controller
1-12	Super Ports
13-253	Ports and Consumers Adapters
253-254	Traffic COP (optionally dualized)

All X-Net stations are based on one or more micro-processors, whereby a very powerfull and flexible system is achieved.

Resource Adapters:

a. Super Ports: Emulator Super Ports (XEP), Gateways (XGT), and Hosts (XHOST)

b. Ports: Transparant Ports (XTP), Communication Ports (XCP) and Process Monitor and Control Ports (XMP)

Consumer Adapter

Terminal Adapters (XTA), Terminal Concentrators (XTC) and Workstations (XWS).

14.3.1 Controller (XCT)

The X-Net Controller synchronizes the traffic on the X-Net and allocates bandwidth to all attached X-Net Stations. Furthermore a system diagnostic is executed as a low priority task. The XCT is a central part of the X-Net and to ensure a high degree of availability of the network dualization of the XCTis available as an option.

For detailed information about the function of the XCT see chapter 4.

14.3.2 Terminal Adapter (XTA)

The XTA is a small stand-alone unit with one asynchronous serial V24/RS232 interface for attaching existing customer facilities such as terminals, VDU's etc. to the X-Net. The XTA is a compact ruggedized box for connection directly to the terminal plug. The characteristics for the adapter are following:

a: V24/RS232 (circuit 101,102,103,104,105,106,107,109) or 20 mA Current Loop as option (circuit 103,104)

b: switch selectable speed:

9600	b/s
4800	b/s
2400	b/s
1200	b/s
600	b/s
300	b/s
110	b/s
50	b/s

c: switch selectable local echo:

local echo selected: inputted characters are assembled into blocks of max. 16 characters and forwarded to the X-Net when block full (16 char.), forward timer run out or on carriage return; the inputted characters are echoed locally on a character by character basis.

local echo not selected: as above, except no local echo.

d: character mode is 7 bits/char., even parity and one stop-bit (other available as customer option).

e: two V24 circuits (105,106) are transferred transparent through the X-Net for end to end flow control.

f: dual X-net interface (redundant cables) available as option.

g: with optional Traffic Cop (XCOP) included in the X-Net parameter b-e can, as an additional feature, be modified by downloaded parameters.

14.3.3 Port (XTP)

The X-Net Transparent Port is a modular device which interfaces 4,8 or 12 V24 serial interfaces to the X-Net. The negotiation about free resource allocation is integrated in the XTP and carried out by a request from a consumer. This makes the XTP e.g. suited for connection to existing serial links on a minicomputer or other data equipment whereby the available resource links is shared in an optional way between the consumers on the X-Net.

The characteristics for the XTP are the following:

a: V24/RS232 (circuit 101,102,103,104,105,106,107,113,114,115) or 20 mA Current Loop adapter as an option

b: switch selectable speed:

9600	b/s
4800	b/s
2400	b/s
1200	b/s
600	b/s
300	b/s
110	b/s
50	b/s

c: switch selectable local echo:

local echo selected:	inputted characters are assembled into blocks of max. 16 characters and forwarded to the X-Net when block full (16 char.), forward timer run out or on carriage return; the inputted characters are echoed locally on a character by character basis.
local echo not selected:	as above, except no local echo.

d: character mode is 7 bits/char., even parity and one stop-bit (other available as customer option).

e: two V24 circuits (105,106) are transferred transparent through the X-Net for end to end flow control.

f: dual X-net interface (redundant cables) available as option.

g: with optional Traffic Cop (XCOP) included in the X-Net, parameter b-e can as an additional feature be modified by downloaded parameters.

14.3.4 Terminal Concentrator (XTC)

The XTC is a concentrator interfacing up to 4 serial V24/RS232 asynchronous (optionally synchronous) links to the X-Net. It may be considered as four XTA's integrated in one unit with following characteristics:

a: V24/RS232 (circuit 101,102,103,104,105,106,107,108,115) or 20 mA Current Loop adapter as option

b: switch selectable speed: 9600 b/s 4800 b/s 2400 b/s 1200 b/s 600b/s 300b/s 110b/s 50b/s c:switch selectable local echo:

local echo selected: inputted characters are assembled into blocks of max. 16 characters and forwarded to the X-Net when block full (16 char.), forward timer run out or on carriage return; the inputted characters are echoed locally on a character by character basis.

local echo not selected: as above, except no local echo.

d: character mode is 7 bits/char., even parity and one stop-bit (other available as customer option).

e: two V24 circuits (105,106) are transferred transparent through the X-Net for end to end flow control.

f: dual X-net interface (redundant cables) available as option.

g: with optional Traffic Cop (XCOP) included in the X-Net, parameter b-e can as an additional feature be modified by downloaded parameters.

14.3.5 Traffic COP (XCOP) - Standard Version

The Traffic COP intercepts and can modify the network dialogue between members of its domains. The basic network dialogue between the COP-domain members are relayed through the XCOP and following services can be specified for domain members when using the standard version XCOP.

- Establishment of permanent connections through the X-Net
- XTA to XTA communication (this allows for example printers to be attached to the X-Net via XTA's).
- Virtual Ports including several physical Ports.
- Terminal or Comm. line interface parameter downloading (see XTA, XTP, etc).

These features increase the flexibility of the network as the Traffic COP may be connected and disconnected to the network without impact on the basic network dialogue.

The standard XCOP contains the specified services for members of its domain in PROM and these are therefore specified at installation time.

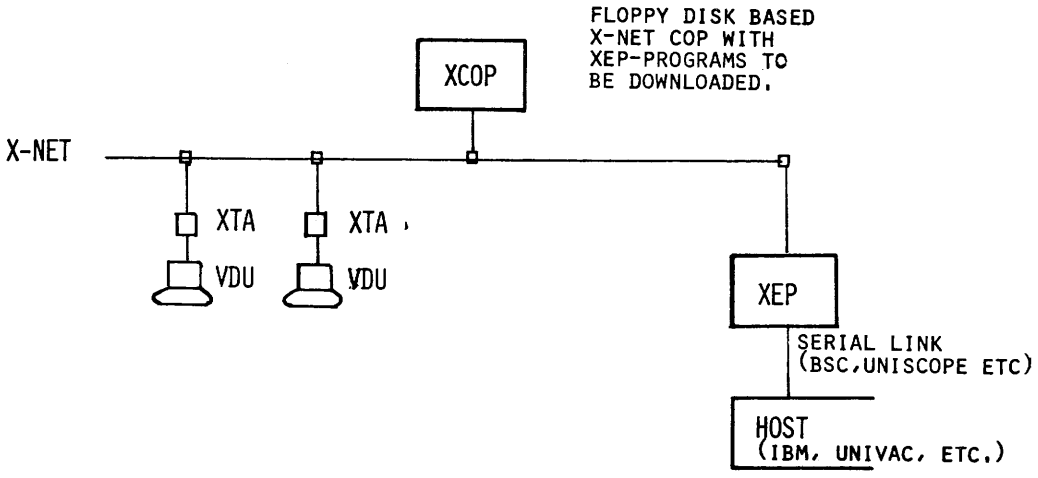
Note:
Custom versions of the XCOP function providing above and further services can on request be defined between customer and Christian Rovsing A/S.

Customer XCOPs with keyboard, floppy disc and editor can optionally be delivered from Christian Rovsing A/S. This makes it possible dynamically (e.g. operator dialogue) to change downloadable parameters, change virtual port-configurations and other user oriented parameters (e.g. downloading of emulator programs etc. to devices on the X-Net) at operationally feasible times.

14.3.6 Emulator Super Port (XEP)

The XEP is an integrated unit including power supply, emulator board, X-Net I/F, map-function to X-net attached terminals and a serial link to the Host-computer. The emulator board is based on the MP square board (see data sheets) which is a powerfull dual processor board developed and manufactured by Christian Rovsing A/S.

The emulator and map functions are either contained in PROM, loaded from floppy disc or downloaded via the X-Net from a traffic Cop (XCOP), giving a very flexible unit in a typical configuration shown below.



The XEP exists in several versions, where an example is a 3270 emulator supporting several terminals with map functions for either VT100 or Tandberg terminals. The emulator is equipped with standard map functions, but optionally other map functions can be defined between the customer and Christian Rovsing A/S.

14.4 X-NET - Functional, Formats and Protocols

The X-Net Synchronization, Formations and Protocols are identical to those found with the TDX-BUs, for details are referred to Chapter 8, section 8.3. The following section 14.4.1 constitutes a short introduction to the general funcion of X-Net, while section 14.4.2 and 14.4.3 describe the difference to the TDX Bus, with regard to Network Level and Bandwidth allocation

14.4.1 Introduction

The X-Net Local Area Network is an efficient, fast digital link between terminals (VDUs, printers etc.) and other terminals or a number of small and large computers.

The X-Net consists of one or a dualized set of controllers, X-Net cable, Wall Outlets, and a number of X-Net stations.

The design has been especially aimed to ensure very high immunity to electrical interferences and low generation of interference itself. The X-Net can therefore safely be used -instead of conventional data transmission - in electrically noisy locations or where it is essential to restrict radiated noise to a minimum.

Utilizing two screened twisted pair cables, one for data transport to the X-Net stations (lower bus) and the other for data transport from the stations (upper bus), data is transported in packets between X-Net stations.

The addressing scheme allows up to 254 stations to share a single X-Net. This greatly reduces the number of separate circuits required in many systems, so that many large and expensive multicore cables can be replaced by a single pair of cables.

Full packet protocol with error detection and correction is implemented in the X-Net for data integrity.

The unique addressing scheme allows multiple connections between stations on a single X-Net as each station comprises several datastreams distinguished by "Data Type" (see Chapter 8, section 8.3). Furthermore multiple connections between stations on different, interconnected X-Nets are supported.

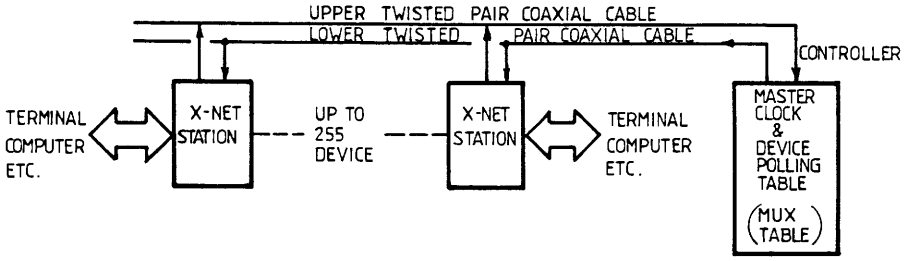


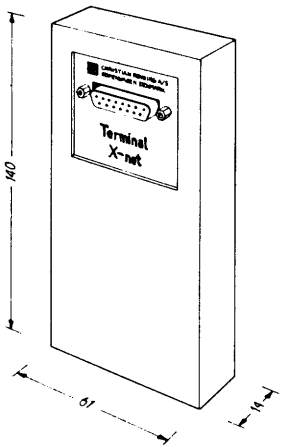
Figure 14.4.1-1

The Bandwidth allocated to each X-Net station is dynamically changeable by requests from any X-Net Station in the network. By a request about changed bandwidth to a station, the X-Net Controller (XCT) reconfigures the MUX-No table, whereby the required bandwidth on the X-Net is achieved, provided the total bandwidth of the X-Net is not exceeded.

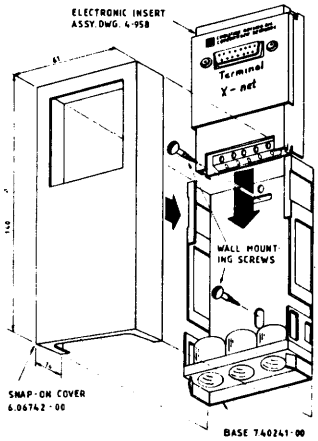
The X-net is designed such that any components, including the XCOP, can be attached or dismounted, with no need for a global power up. This is guaranteed by letting the XCOP continuously poll all devices, requesting their internal status and delivering their link parameters.

Overleaf the X-Net Wall Outlet is shown (Fig. 14.4.1-2) together with locating example

X-NET WALL OUTLET (XWO)



X-net Wall Outlet



Exploded View

Locating example :

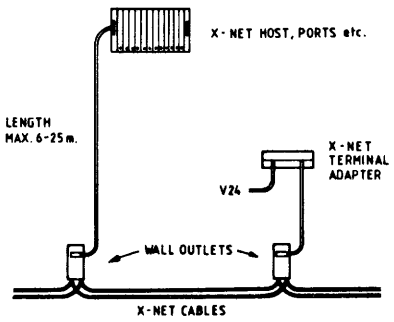


Figure 14.4.1-2

14.4.2 Network Level

The primary service of the network level is to setup datastreams between X-Net stations. The interface through which the application level uses the facilities of the network level consists of the routines: OPEN and CLOSE. These routines respond with a status code indicating if the operation succeeded or a failure has occurred. The operations are synchroneous, in the sense that each routine-call must be completed before a new request may be served. The next higher level sets up a line by calling:

OPEN (DEST, SOURCE)

DEST is the CR-ID of the remote end of the line:

SOURCE is the CR-ID of the local (subscribing) device:

A logical line is removed by application level by calling:

CLOSE (LINE), where
LINE is the number of the logical line.

Another service of the network level is to request the X-Net Controller to change the bandwidth assigned to the actual station on the X-Net. A request is made by application level by calling:

REQBW (t, LEVEL), where

t: is the X-Net station device no. (0 to 255)

LEVEL is a number between 0 and 14, which represents baudrates on: 0, 100 baud, 200 baud, 400 baud,...., 800K baud.

The network level responds with a status code indicating if the request is accepted or rejected by the controller.

The network level carries out its services through logical channel 0 for OPEN/CLOSE and channel 1 for bandwidth requests. The channel 1 traffic is a point to point connection between the actual device and the X-Net controller, and is monitored by a subset of the X-Net packet protocol. The communication on channel 0 may be shifting between the actual device and several other X-Net stations, and is therefore supervised by a Message Protocol.

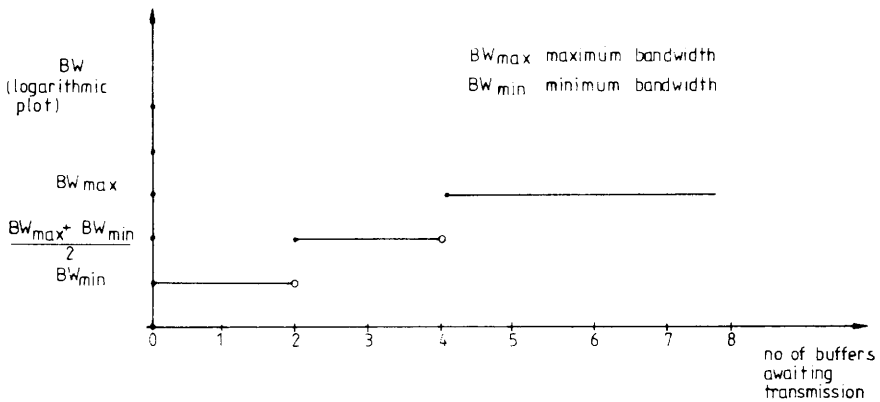
A channel is setup by sending a datagram containing the subscribing CR-ID and the remote CR-ID. The remote device reassigns the CR-Id of channel 0, opens the requested channel if it is free (closed) and responds with a datagram containing the CR-ID of the new-opened channel together with on ACK-status code. If the remote requested channel is already open or under opening a datagram containing an "Not-accepted"-status code is responded.

14.4.3 Bandwidth Allocation

The X-net supports two different ways to request new bandwidth to a device, either by request from the application layer or automatic adaptive allocation from the frame manager which requests bandwidth depending of no. of frames queued waiting to be transmitted.

The two methods may be used mixed in a network configuration as it is possible to lock and unlock the automatic bandwidth request.

The automatic bandwidth allocation method requires two parameters from higher levels indicating the maximal and minimal bandwidths wanted on the device. These parameters are used in the algorithm shown below.



The improved requesting algorithm both supports very strict requirements to allocated bandwidth as e.g. synchronous encrypted modem connections and supports extremely large virtual bandwidth in burst traffic environment as typical in man-machine interface.

14.4.4 Diagnostic and Statistic

System diagnostic consists of a low priority task in the controller supervising all appended X-net devices with a frequency of typically a few devices per second. Three requests from the controller not answered by the device, results in an error-message to higher levels.

Furthermore a fast switching between optionally dualized upper buses is implemented ensuring the best transmission path is selected. The switching criteria is the ratio of no. of valid frames in the network measured in two consecutive comparable time intervals. If this ratio is more than 4, a switch is executed.

Statistic and diagnostic information may be requested from each X-net device from higher levels via the system communication on the X-net. The information received are the following:

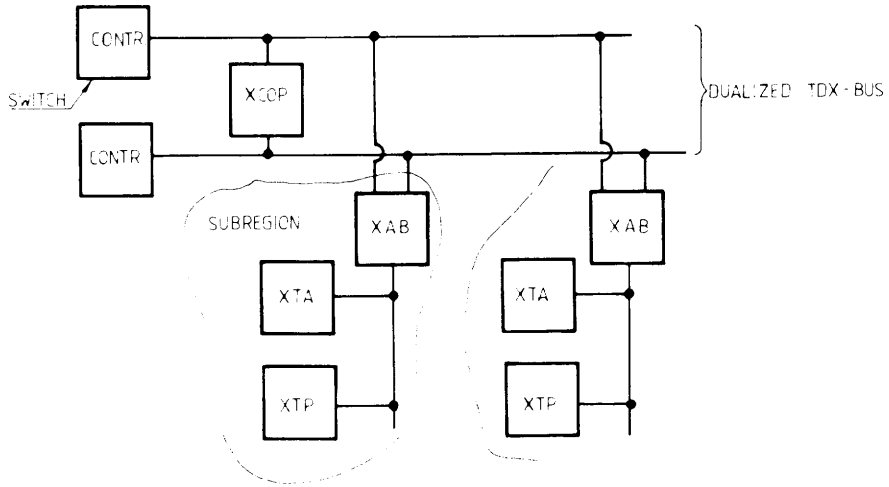
- no. of transmitted packets
- no. of retransmitted packets
- no. of retransmissions exceeding 3
- no. of received packets
- no. of received timeouts
- identification of lower bus used to reception.

This information is available to the XCOP (customized version).

14.5 Fault Tolerance

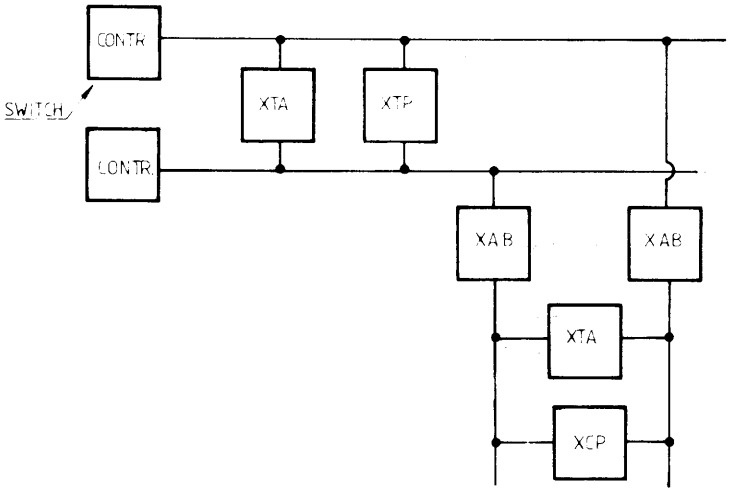
The X-Net can by standard options be made fault tolerant to the degree found feasible in the specific application, by dualization of the key elements e.g. cables, X-Net controller and X-Net COP.

14.5.1 X-Net cable limited dualization



According to the application in question, dualization of the main distribution part of the X-Net cable is the most feasible, resulting in that a failure can be restricted to a subregion in the network (dualization of X-Net controllers shown in illustration is not a requirement).

14.5.2 X-Net cable full dualization

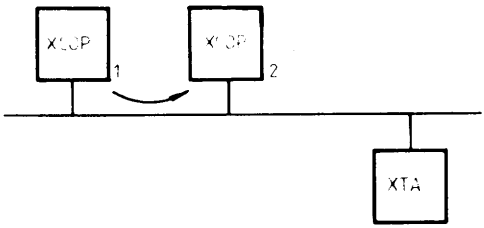


Each device interfaces to a dualized bus, whereby a bus failure will not affect operation of attached devices (dualization of X-Net controllers shown in illustration is not a requirement).

14.5.3 Dualized X-Net Controllers (XCTR):

Switching between the two controllers and busses is either done manually or supported by a watchdog. This is an X-Net device that continously monitors the traffic at the two controllers.

14.5.4 Dualized Standard COP's (XCOP)



It is an important feature in dualization to keep the status of the back-up device up to date. This is done in the case of the COP in the X-Net by letting it as part of its status update cycle perform a partial update of the network status at the back-up COP. The back-up COP will in case of a missing status update (timeout based) notify all the members in the network that it is taking charge, the failing COP will when reinstalled, as part of it's power up procedure inform the back-up COP and all members that it is again taking the role as the active COP.

14.6 X-NET Basic Components and Utilities

Below the X-Net Components are listed. For detailed information, please refer to datasheets chapter 10.

X-Net Basic Components:

1. X-Net Wall Outlet, CR2510 (XWO)
2. X-Net Terminal Adapter, CR2550 (XTA)
3. X-Net Port, CR2560 (XTP, XCP and Standard X-COP)
4. X-Net Amplifier and Branching unit, CR2520 (XAB)
5. TDX (X-NET) Controller, CR1070 (XCT, insertable into X-Net port CR2560)
6. X-Net Cable, CR2500
7. X-Net Termination, CR2501
8. CR8 Workstation, CR2001
9. MP-SQUARE microcomputer CR2000

   } (XWS, XEP, customized XCOP)

10 X-Net (MP-SQUARE) adapter CR2525
11. LTUX-S, CR1060 (XTC when packaged into CR2505 X-net chassis)

X-Net utility - and spare parts:

12 LTUX-M-FE, CR1090
13. LTUX-M-CPU, CR1091
14. LTUX-M-X21, CR1061
15. BTM-X, CR1082
16. Chassis for X-Net devices, CR2505
17. Back Panel, tp.1, CR1085
18. Back Panel, tp.2, CR1086
19. Cable for back panel tp.1 and tp.2, CR1096/035
20 Cable for Port (BTM-X) to Wall-Outlet, CR1069
21- Cable for X-Net (TDX) controller to Wall-outlet

NOTES:

iew of classroom in the CR80 scandinavian training center. Frequent courses on
l levels are here available in programming, maintenance and use of the CR80
mily of computers.

**The Computer room of the scandinavian CR80 training center in Copenhage
Hands-on experience is a vital part of most CR80 courses.**

15. DETAILED INDEX

CR80 MINICOMPUTER
HANDBOOK 1982/83

o STAY UP-TO-DATE ON THE LATEST
CR80 COMPUTER FAMILY DEVELOPMENTS

Fill out the below formular and mail to:

CHRISTIAN ROVSING A/S
Att.: Customer Relation Dept.
Lautrupvang 2
DK-2750 Ballerup
DENMARK

☐ YES! I would like the next edition of the CR80 Minicomputer handbook sent to me (without charge or obligation) at the following address:

NAME_____
COMPANY_____position_____
Street Address_____
City/State_____postal code_____

☐ I would like further material on the below marked items

☐ Please have a representative call to discuss my special application for:

- ☐ CR80 MINI
- ☐ CR80 TWIN
- ☐ CR80 MAXIM
- ☐ CR80 FATOM
- ☐ X-NET
- ☐ DAMOS
- ☐ AMOS (XAMOS)
- ☐ Other hardware/software products:_____

My application is in the area of:

- ☐ Commercial EDP:_____
- ☐ Communication, if yes please list area of interest:
  - ☐ Protocols, which: _____
  - ☐ Networking:
    - ☐ X25/OSI
    - ☐ SNA
    - ☐ DEC NET
    - ☐ other, which:

☐ Front Ending, which computer:_____
☐ Local Area Network:

which computers:

_____
_____
_____

which terminals:

_____
_____
_____

☐ Device emulation, to/from which devices:

_____
_____
_____

☐ Technical systems, Process Control etc.:

_____
_____
_____

☐ Defense Systems

Your comments and suggestions will help us in our continuous effort to improve the quality and usefulness of our handbooks.

What is your general reaction to this handbook ?
(accuracy, completeness, format, organization etc.)

_____
_____
_____

What features have you found most useful? _____

_____
_____
_____

Does the handbook satisfy your needs?_____

_____
_____
_____

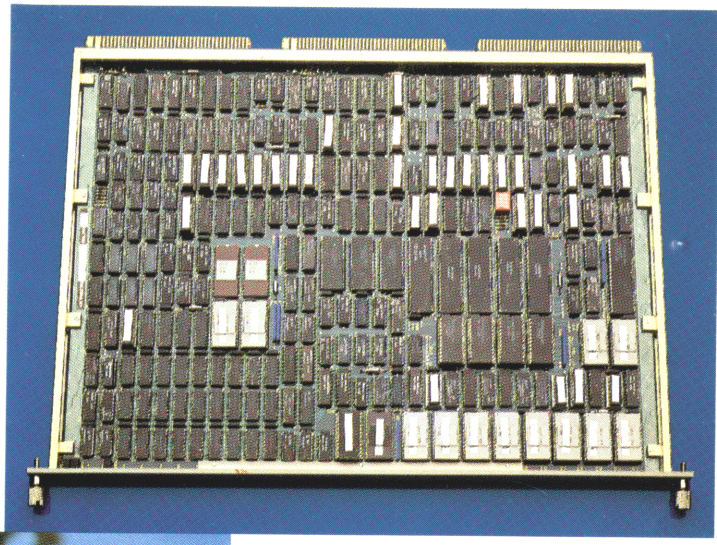
What errors have you found?_____

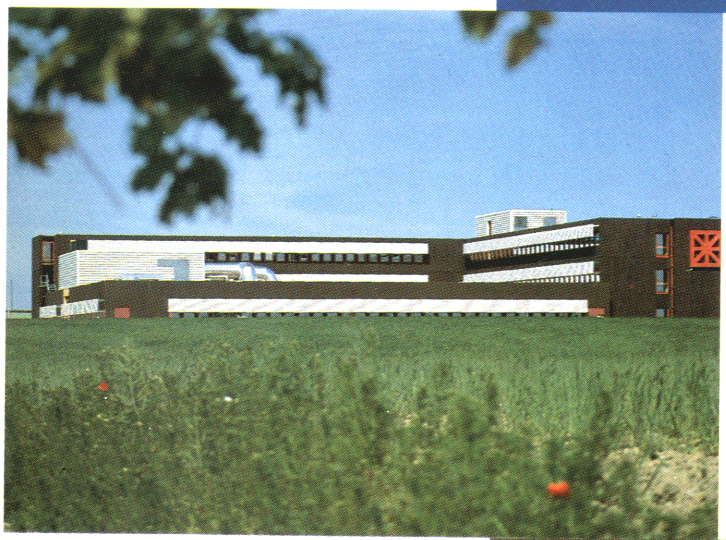
_____
_____
_____

Additional comments:_____

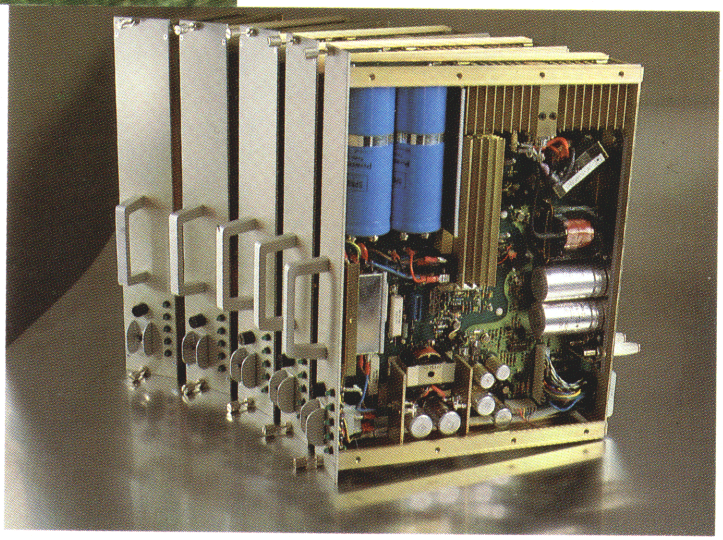
_____
_____
_____

Representative plug-in module of the CR80 Computer Family. The eight-layer printed circuit board contains more than 200 Medium and Large Scale Integrated circuits in high-speed Schottky TTL technology. The module is enclosed in a rigid aluminium frame and connects, via three gold covered high density connectors, to the multiple fast computer buses of the CR80.

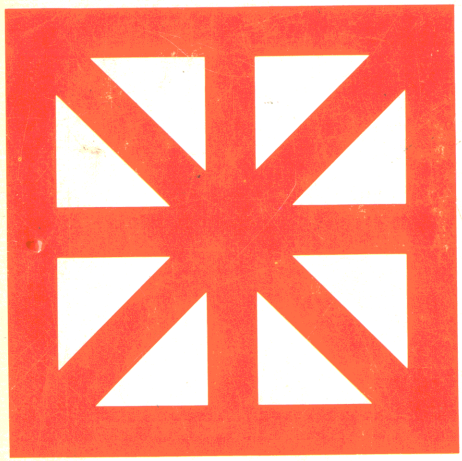




The 128.000 sq. ft. Scandinavian Headquarter of Christian Rovsing, part of approx. 200.000 sq. ft. of CR80 engineering and production facilities centered around Copenhagen. The high volume production line is based on the most modern semiautomatic component insertion, flow soldering and computerized automatic test equipment.



The high performance, multiple voltage switching power supply is a plug-in module, and, as all CR80 computer modules, exchangeable from the front without cables to be detached.



CORPORATE PROFILE

Christian Rovsing Designs, manufactures, sells and services the CR80 family of computers and related software, peripherals and supplies, as well as CR80 based turn-key systems. The company's products are delivered and used world-wide in a broad range of applications and programs, including commercial EDP, communications, networks, Front Ending, timesharing, Local Area Nets, industrial control, aerospace, computation, engineering, simulation and word processing.

CHRISTIAN ROVSING A/S
Lautrupvang 2
2750 Ballerup
Denmark
Phone: 02 - 65 11 44
Telex: 35 111 cr dk
Telefacsimile: 02 - 65 43 73

CHRISTIAN ROVSING CORPORATION
1337 Thousand Oaks Blvd. suite 220
Thousand Oaks
California 91362
U.S.A.
Phone: (805) 497-6722
Telex: 9103365733
Telefacsimile: (805) 497 8271

CHRISTIAN ROVSING A/S U.K.
230, Hammersmith Grove
London W6 7HG
United Kingdom
Phone: (1) 74 92 532
Telex: 947157 crasuk g

CHRISTIAN ROVSING INDUSTRIAL A/S
Lautrupvang 2
2750 Ballerup
Denmark
Phone: 02 - 65 11 44
Telex: 35 111 cr dk
Telefacsimile: 02 - 65 43 73

CHRISTIAN ROVSING IBERICA S.A.
Avenida Del Brasil 4
Planta 5A
Apartementos 4F
Madrid 20
Spain
Phone: (1) 455 4170
Telex: 48 930 cref

CHRISTIAN ROVSING INTERNATIONAL A/S
Dagmarhus
H. C. Andersens Boulevard 12
1553 Copenhagen V
Denmark
Phone: 01 - 13 11 66
Telex: 16 066 cri dk

CHRISTIAN ROVSING BV
Langstraat 58B
2242 KN Wassenaar
Netherlands
Phone: (17) 51 18 233

Distributor:

Rosenborg Tryk - Copenhagen