PHILIPS

PHILIPS TERMINAL SYSTEMS

TOSS SPECIALISTS' REFERENCE MANUAL

Internal Library

Module M14

Data
Systems

Date : November 1980

Code : 5122 993 16631

MANUAL STATUS SURVEY

Module M14 "TOSS SPECIALISTS' REFERENCE MANUAL"

This issue covers Release 10 of TOSS.

The TOSS Specialists' Reference Manual is intended to be of use to PTS specialists in both software and hardware, although more emphasis is placed on the software aspects. It is therefore desirable that the reader has some knowledge of PTS applications.

The first Chapters of the manual deal with some specific hardware devices, such as Selector Units and Transfer Units. The information was, to some extent, drawn from the three Support Engineer Handbooks listed below. These are useful additional reference sources for the reader who wants to know more about the hardware in detail.

* Support Engineer Handbook, Terminal Equipment. Volumes 1 & 2.
* Support Engineer Handbook, Terminal Computer,
* Support Engineer Handbook, Peripheral Equipment.

Readers should also be familiar with the following PTS User Library manuals:-

* CREDIT Reference (M04).
* Assembler Programmer's Reference (M06).
* Data Management (M23).
* TOSS Utilities (M08).
* DOS-PTS Programmer's Guide (M11 and M12)

Suggestions concerned with improving the usefulness of the manual are always welcome, and may be directed to SSS / T.& D at Apeldoorn.

2 SELECTOR UNITS

3 TRANSFER UNIT (TFU)

4 CONTROL PANELS

CHAPTER 1 INTRODUCTION

The Terminal Operating System Software (TOSS) Monitor, is the operating system used to control a PTS configuration in the production environment.

The monitor uses many tables and queues to keep track of tasks and control the various devices attached to the system. This manual contains detailed descriptions of the contents of these tables and their function within the Monitor. Tables used by applications written in CREDIT are also included.

Sharing of resources between tasks, and task dispatching, are realized by means of a priority-driven interrupt system.

The reference section at the back of the manual contains pointers to the Chapters containing details of the appropriate tables. There are also two fold-out pages showing relationships between tables for both the monitor and CREDIT applications. These provide a key to the reference section, which in its turn indexes the Chapters.

2.1 GENERAL

All selector units consist essentially of a power-supply unit and a logic unit. The power-supply unit supplies both the logic unit and the attached input/output devices with the power required. The unit contains a communication section, connected to the channel unit, and up to four buffer circuits which adapt the input/output devices to the communication section.

There two types of selector unit, Selector Unit Modular Local (SUML), and Selector Unit Modular Remote (SUMR).

The device addresses and input priorities of the devices connected to the selector unit are adjustable within the selector unit.

Figure 2.1 shows typical Selector Unit configurations.

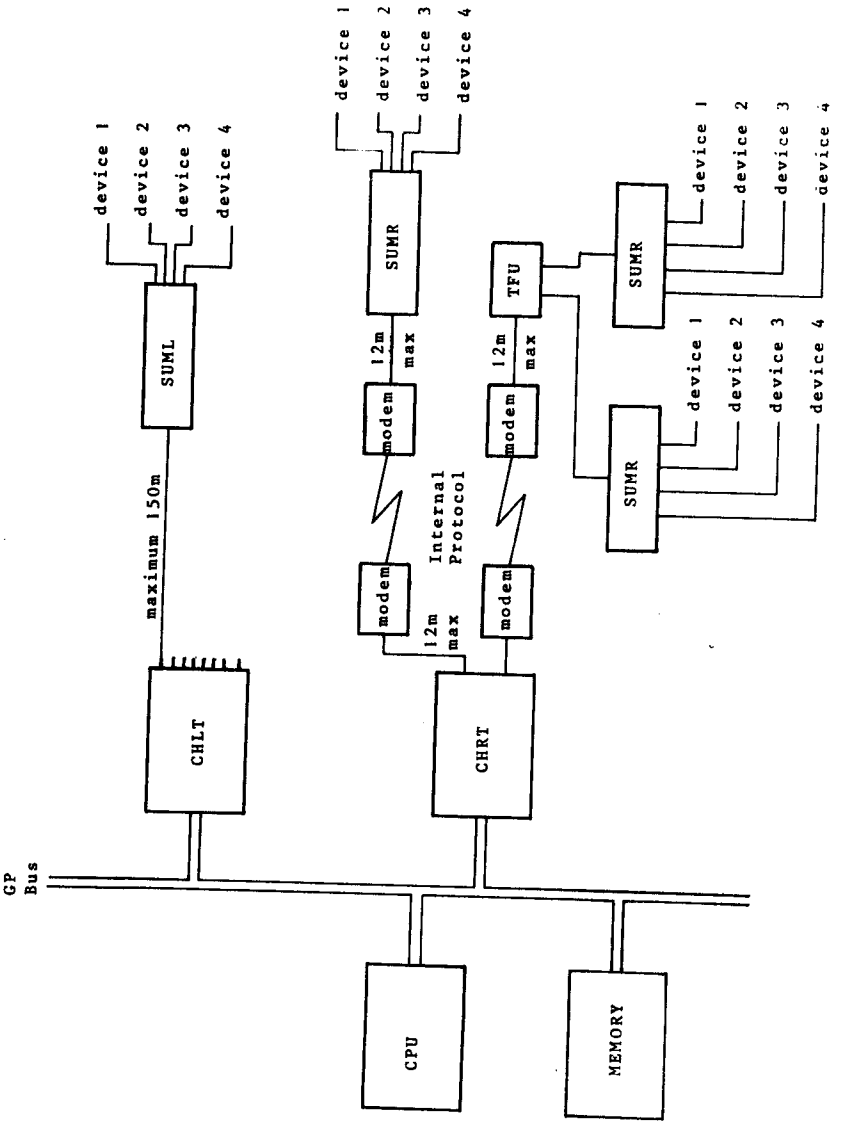


Figure 2.1. Selector Unit Configurations.

2.2 SELECTOR UNIT MODULAR LOCAL (SUML)

In this type of selector unit, up to three input/output devices can be connected through buffer circuits which are built on exchangeable plug-in adaptor boards. However, the printer buffer logic is integrated on the basic board which contains the communication section (figure 2.2).

The device addresses and the input priority levels (to the CHLT) are adjustable by means of jumpers on the adaptor boards (address selection), and on the basic board (priority selection).

Input device addresses are adjustable from 1 to 7.
Output device addresses can be 1, 2, 3 to 6 and priority-levels 1 to 4 (1 is highest priority).
Output device address 2 is always used for the print device.

Priority levels are only adjustable for Input.

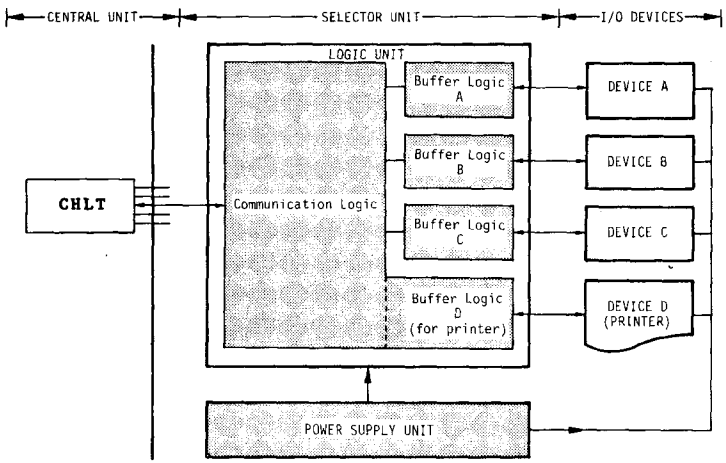


Figure 2.2. Basic structure of a modular selector unit.

ABBREVIATIONS USED IN THE FOLLOWING TEXT:-

ABC	Acknowledge Block Check.
ACK	ACKnowledge.
DIN	Data INput, from SUMR.
DOS	Data Out, Single character mode.
DOB	Data Out, Block mode.
DRD	Data Request Delayed.
DRI	Data Request Immediate.
FIFO	First In First Out.
NAK	Not AcKnowledged.
OBC	Output Block Check.
OER	Output ERror.
ROM	Read Only Memory.
SER	Selector unit ERror.
SUML	Selector Unit Modular, Local.
SUMR	Selector Unit Modular, Remote.
STD	STatus of Device.
SYN	SYNchronization character (X'55').
TFU	TransFer Unit.

2.2.1 Operation in Brief of SUML

2.2.1.I Layout of Output Messages

* Start Bit.

A complete output message from the CPU to CHLT consists of two bytes, an address byte and a data byte (figure 2.3). The first bit of the message is used as the start bit during the serial transfer between the CHLT and selector unit. The bit is set by CHLT hardware.

* Addresses and Data.

After the start bit, a 3-bit device address and a 3-bit terminal address are present, which specify respectively the terminal (plug on CHLT to which selector unit is connected), and the device (device on selector unit).

The last bit of the address character is reserved for a parity bit which is used only during the serial transfer between CHLT and selector unit. When the message leaves the CPU this bit and the first bit of the data character are not significant.

* Message from CPU to CHLT.

When the output message from the CPU reaches the CHLT, the CHLT will supplement the address and data characters with odd parity bits, and set the start bit to one.

When the CHLT then starts the serial transfer to the addressed terminal (selector unit), the 3-bit terminal address has served its purpose and is cleared (figure 2.3).

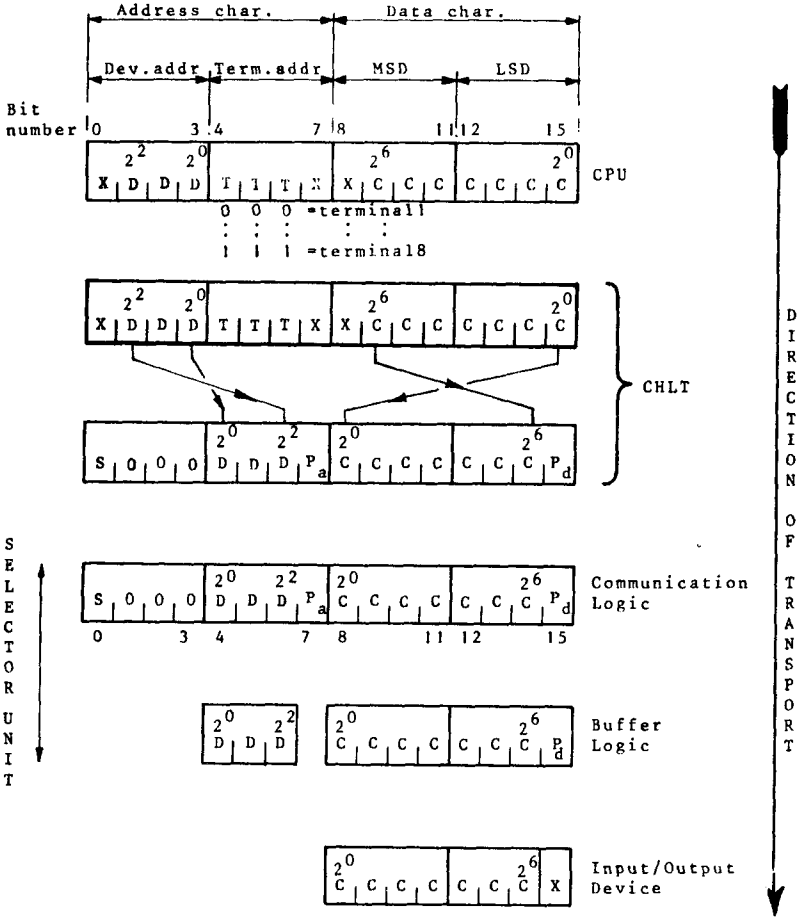
* Message from CHLT to SUML.

The output message from the CHLT is received by the communication logic of the selector unit, where the parity is checked. If the address parity is correct the data character will be passed to the buffer logic of the addressed output device (figure 2.3).

The 3-bit device address is passed on to the buffer circuits, which are each provided with address decoders.

* Character from SUML to Device.

The character code received by the addressed buffer logic will be supplied unchanged to the output device, provided that the communication logic finds that the data character has correct parity.



MSD - Most Significant Digits. LSD - Least Significant Digits.

D - Device Addr. Bits. | S - Start Bit. | X - Not Significant.
T - Terminal No. Bits. | P_a - Addr. Parity Bit.
C - Character Bits. | P_d - Data Parity Bit.

Figure 2.3. Message Transfer between CPU, CHLT, SUML and I/O devices.

2.2.1.2 Layout of Input Messages

Input messages are transferred from selector unit to channel unit and have the same layout as output messages.

* Character from Device to Selector Unit.

The character codes supplied by input devices are first taken into the appropriate buffer circuits in the selector unit, where the device address is encoded. The characters are then, in accordance with input priority, transferred one by one to the communication logic, which generates the required data parity bits and controls the transfer to the CHLT.

* Message from SUML to CHLT and CPU.

When the communication logic is loaded with the information from the buffer logic, the input message (for CHLT) is completed by setting:-

* start bit to one.
* terminal address to zero.
* odd parity for the address byte and character byte.

The serial transfer to the CHLT starts when the communication logic is polled by the CHLT. When the message is received by the CHLT it clears the parity bits and inserts the terminal address before the final transfer to the CPU takes place.

* Administrative Messages from SUML to CHLT.

Apart from the input messages which are supplied by the input devices, there are also some administrative input messages for the CHLT, which are supplied by the selector unit.
These messages have the same layout as shown in figure 2.3. However,the device address in the address byte will also be cleared.

A data character in an administrative input message can contain the following information:-

* Code X'83'. (X'03' when reaching CPU).
  Power failure , which indicates that the selector unit power has just been switched on or has just recovered after a failure.

* Code X'85'. (X'05' when reaching CPU).
  NAK , which indicates that the last character transmitted to the SUML has not been acknowledged, due to a parity error or the device not being in output mode.
  Retransmission will be performed by the software.

* Code X'07'. (X'07' when reaching CPU).
  ACK , which indicates that the message last transmitted to the SUML has been acknowledged.

* Code X'9B'. (X'1B' when reaching CPU).
  Input Trouble , which indicates that four consecutive attempts by the SUML to transfer a character to the CHLT have failed.
  (No hardware controlled ACK has been received from the CHLT).

2.3 SELECTOR UNIT MODULAR REMOTE (SUMR)

In this type of selector unit the communication logic consists of a common block and a printer interface located on a basic board, to which three additional interface boards can be connected in order to adapt three devices (figure 2.4).

The use of a Transfer Unit (TFU) enables two SUMR's to be connected to the same line, otherwise only one may be connected.

Each adaptor board is given a specific device address, selected by means of removable jumpers on the adaptor boards.

Table 1 shows which input and output addresses are used for various devices.

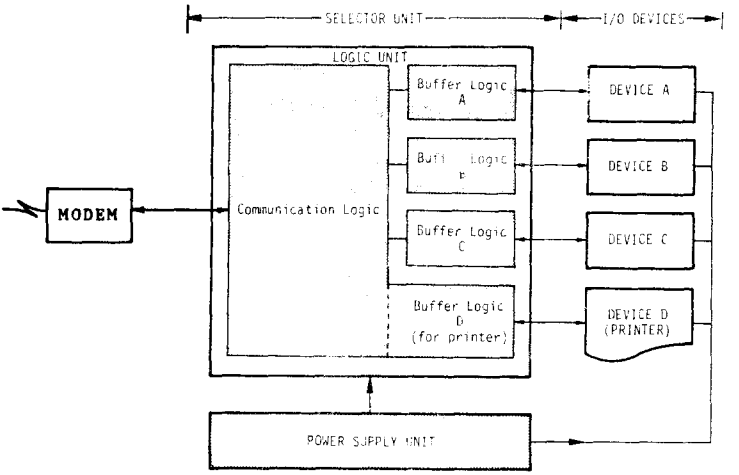


Figure 2.4. The structure of the Selector Unit Modular Remote.

DEVICE	MAIN ADDRESS	RESERVE ADDRESS
Permanent printer interface	2	-
Keyboard	1	5
Numeric indicator	4	3
Signal display	3	5
Printer (on optional adaptor board)	3	6
Display (VDU)	4	3
Other devices	6	any of 1 thru 5

Table 1. Device Addresses.

Input devices should be arranged in priority succession depending on input rate and access time for each device. The permanent printer interface and the three adaptor boards can be given any of four levels, selected by jumpers in a selection field in the communication logic of the SUMR. The priority level is applicable to the adaptor board position and not to the adaptor board itself. (figure 2.4).

The permanent printer interface, which always has device address 2, contains a 40 character FIFO buffer and logic for indicating printer status, e.g. printer operable, or voucher/passbook correctly inserted.

There is also an operational ROM package to generate two triple spaced characters usually used to print a trade mark. The generation is started by codes X'13' and X'14' respectively.

2.3.1 Operation in Brief of SUMR

2.3.1.1 General.

A SUMR transfers data asynchronously via a modem in full duplex mode between a CHRT and terminal devices.

2.3.1.2 Line Procedure.

Data to/from the SUMR is transferred on the line either character-by-character with demand of acknowledgement for each character, or block-by-block where the acknowledgment is checked after a complete block has been transferred. Block transmission and character transmission can be mixed on the line. Block transmission gives a higher transfer rate than character transmission; the transmission procedure is selected by software for output from a CHRT, and by a jumper on the relevant adaptor board for input.

If a character or block is not acknowledged because of parity failure during output transfer, it is retransmitted under program control. If there is parity failure during input to the CHRT for character-by-character transmission, then the character is retransmitted by the SUMR. For block transmission it depends whether or not the device is able to retransmit the block.

The SUMR contains logic for both Vertical and Longitudinal Redundancy Checks (VRC and LRC) on output from a CHRT. For input to a CHRT, either character or block transmission, a VRC is performed by the SUMR, and the data character is supplemented with an odd parity bit.

2.3.1.3 Message Layout.

Both input and output messages from/to a SUMR have a format of either 16 bits or 9 bits.

The 16 bit format (figure 2.5) is called data format, and the 9 bit format (figure 2.6) is called receipt format because ACK and NAK have that length.

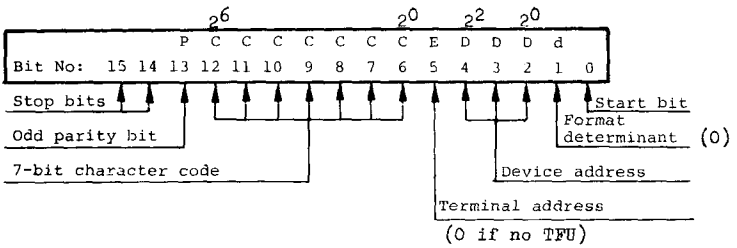


Figure 2.5. Data Format message.

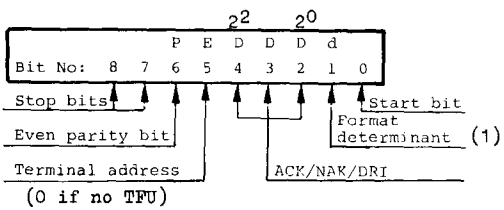


Figure 2.6. Receipt Format message.

2.3.1.4 Messages To and From the SUMR.

The formats shown in figure 2.7 are as they appear when measured on the line.

1. Output to SUMR, Data formats.

Message	St	d	D	D	D	E	C	C	C	C	C	C	C	P	Sp	Sp	
SYN	0	0	1	1	1	0/1	1	0	1	0	1	0	1	0/1	1	1	Each 500 ms*
Dos	0	0	1	---	6	0/1	DATA ZONE							0/1	1	1	Single character
DOB	0	0	1	---	6	0/1	DATA ZONE							0/1	1	1	Blockwise
OBC	0	0	0	0	0	0/1	DATA ZONE							0/1	1	1	Read and clear LRC-VRC

* If no other communication within the 500ms.

2. Output to SUMR, Receipt formats.

Message	St	d	D	D	D	E	P	Sp	Sp	
ACK_{in}	0	1	1	1	1	0/1	0/1	1	1	Data in, acknowledge
NAK_{in}	0	1	0	0	0	0/1	0/1	1	1	Data in, not acknowledge

3. Input from SUMR, Data formats.

Message	St	d	D	D	D	E	C	C	C	C	C	C	C	P	Sp	Sp	
DIN	0	0	1	---	7	0/1	DATA ZONE							0/1	1	1	Data from input device
STD	0	0	1	---	7	0/1	DATA ZONE							0/1	1	1	Same as DIN but the program recognizes that STD/DRD comes from output devices
DRD	0	0	1	---	6	0/1	0	0	0	0	0	0	0	0/1	1	1	
ABC	0	0	0	0	0	0/1	0	0/1	0/1	1	0	0	0	0/1	1	1	VRC/LRC status
SER	0	0	0	0	0	0/1	1	1	0	0	0	0	0	0/1	1	1	Selector unit error
			$2^0 \rightarrow 2^2$				$2^0 \rightarrow 2^6$										

4. Input from SUMR, Receipt format.

Message	St	d	D	D	D	E	P	Sp	Sp	
ACK_{out}	0	1	1	1	1	0/1	0/1	1	1	Data out, acknowledge
NAK_{out}	0	1	0	0	0	0/1	0/1	1	1	Data out, not acknowledge
DRI	0	1	1	---	6	0/1	0/1	1	1	ACK and data request from an output device

Figure 2.7. Formats of Messages to and from the SUMR.

2.3.1.5 Output Messages from CPU to CHRT.

There are three kinds of output messages, shown in figure 2.8.

* A data message to an output device (1-6).
* An Output Block Control (OBC) message.
* A Synchronization (SYN) message.

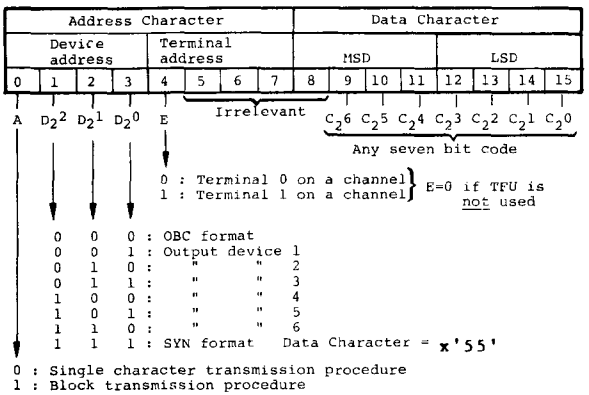


Figure 2.8. Layout of output messages from CPU to CHRT.

2.3.1.6 Input Messages from CHRT to CPU.

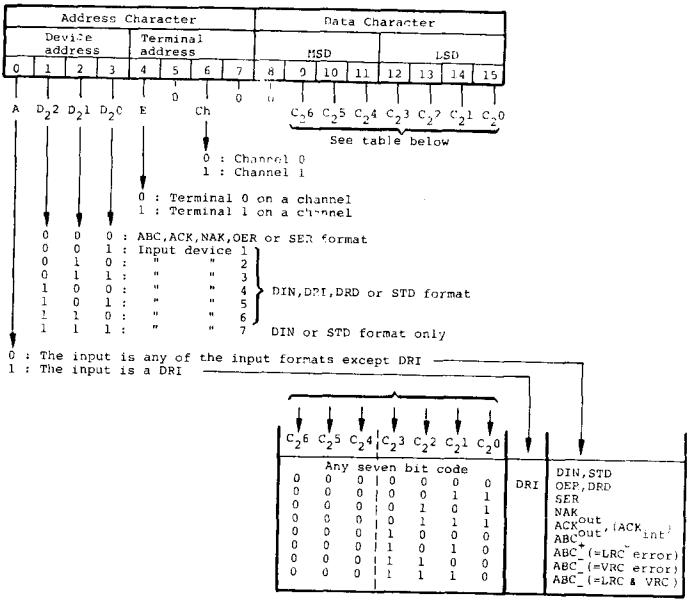


Figure 2.9. Layout of Input Messages from CHRT to CPU.

Input messages, shown in figure 2.9, can be divided into four basic types.

* data messages , which are either Data INput (DIN) messages from an input device (1-7), or STatus information from an input or output Device (STD),(1-7).

* data request messages , either 'Data Request Immediate' (DRI) which replaces ACK from the same output device, or 'Data Request Delayed' (DRD).

* receipt messages , like ACK and NAK, or 'Acknowledge Block Control' (ABC) which is the reply to an OBC message.

* error messages , which indicate 'Selector Unit Error' (SER) or 'Output Error' (OER) if the CHRT does not receive a reply within 100 ms.

Note A characteristic of receipt and error messages is that the Device Address is cleared.

2.3.1.7 Transfer Principle.

For every data message transferred between a SUMR and a CHRT, or vice versa, a receipt message is sent in the opposite direction. This applies to both character and block transmission.

During block transmission the receipt significance is omitted, but there are circuits to check that there is a reply for every message transmitted.

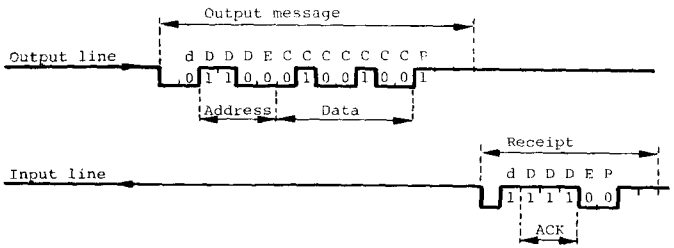


Figure 2.10. Output message with receipt, ACK or NAK.

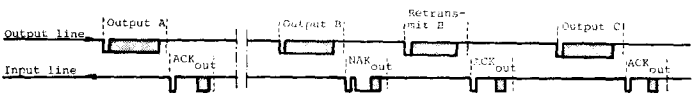


Figure 2.11. Single character output transmission.

Figure 2.11 shows an example of single character transmission of three data characters, namely A, B, and C. The SUMR detects a parity failure in character B, and replies 'NAK', which results in a retransmission of the character by the system software.

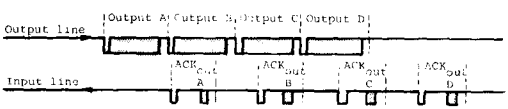


Figure 2.12. Block output transmission.

In block transmission, figure 2.12, the CHRT does not have to wait for the receipt of the previous message before the next one is transmitted to the SUMR (this is not applicable to printers).

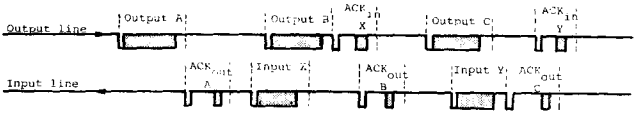


Figure 2.13. Simultaneous output and input, character transmission.

From figure 2.13 it can be seen that data messages and replies may be mixed if simultaneous input and output transmission occurs.

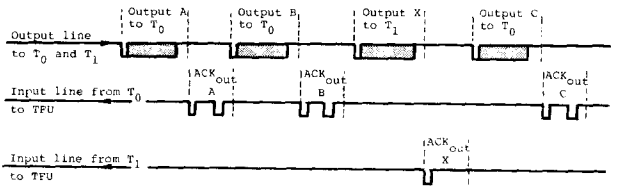


Figure 2.14. Mixed output to Terminal 0 and Terminal 1.

If two SUMR's are connected to the same telephone line via a transfer unit (TFU), data is sent to both terminals at the same time, as shown in figure 2.14. Both SUMR's receive the messages, but only the one whose identity corresponds to the value of bit E replies with ACK or NAK.

Note: A new character can not be transmitted until the previous one is received. Data is sent either to one terminal or to the other.

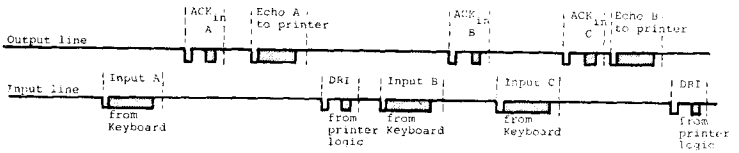


Figure 2.15. Example of Echo.

Figure 2.15 shows an example of input data from a keyboard and the 'echo' of the same data sent back to a printer. The usual ACK is replaced by a 9-bit receipt message, Data Request Immediate (DRI) which means ACK and Data Request.

Figure 2.15 also shows that several characters from the keyboard may be stored in the computer before they are echoed back to the terminal printer.

The selector unit error function, SER, includes the steps to be taken when an error situation occurs and a recovery of SUMR logic is needed.

SER handles the following situations:-

* Power-on in SUMR.
* Short power break (dip in +5V) in SUMR.
* Power-on in a device having a power supply of its own.
* Carrier missing from modem.
* Time-out at transmission attempt.

The last point only inhibits input from connected devices but the other four cause a general terminal reset.

All five situations set a flip-flop which, when the error situation clears, initiates transmission of an error status message, SER (figure 2.16), which is loaded into the transmit buffer via a multiplexer.

Sp	Sp	P	C	C	C	C	C	C	C	E	D	D	D	d	St
			2^6	2^5	2^4	2^3	2^2	2^1	2^0		2^2	2^1	2^0		
1	1	X	0	0	0	0	0	1	1	X	0	0	0	0	0

Figure 2.16. A SER message.

A retransmission of SER is initiated, besides the usual retransmission at NAK, if an ACK is not received within the 'Time-out' period, 250mS.

When ACK is received the SUMR returns to normal status.

The various I/O procedures are illustrated in figures 2.17 to 2.22, and an outline of the internal structure of a Channel Unit in figure 2.23.

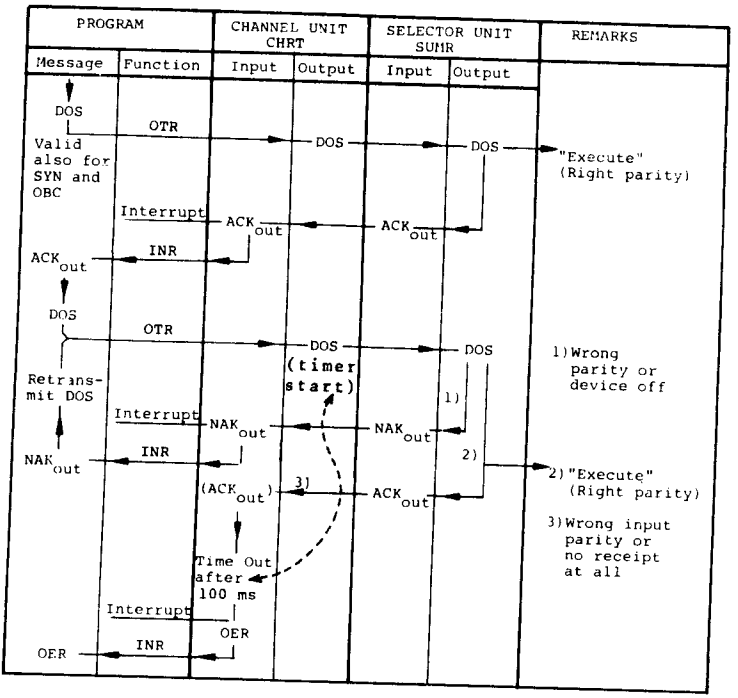


Figure 2.17. Character-by-character Output Procedure; no Data Request.

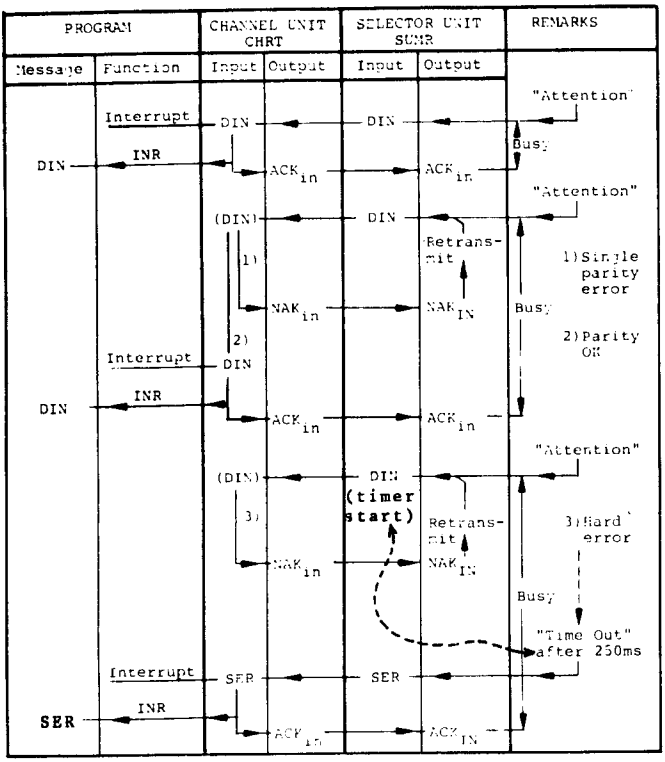


Figure 2.18. Character-by-character Input Procedure.

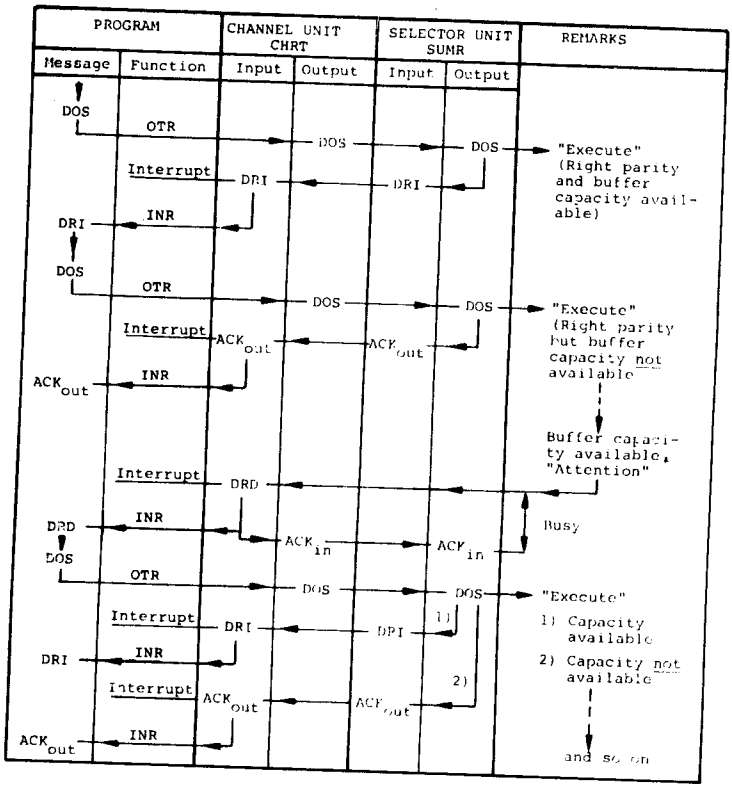


Figure 2.19. Character-by-character Output Procedure with Data Request.

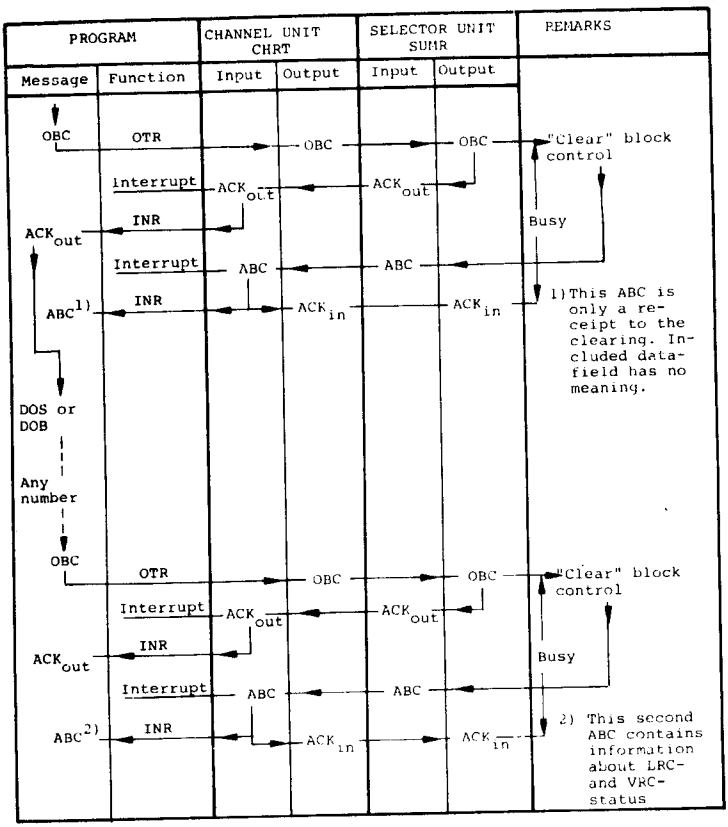


Figure 2.20. Output Procedure with OBC and ABC.

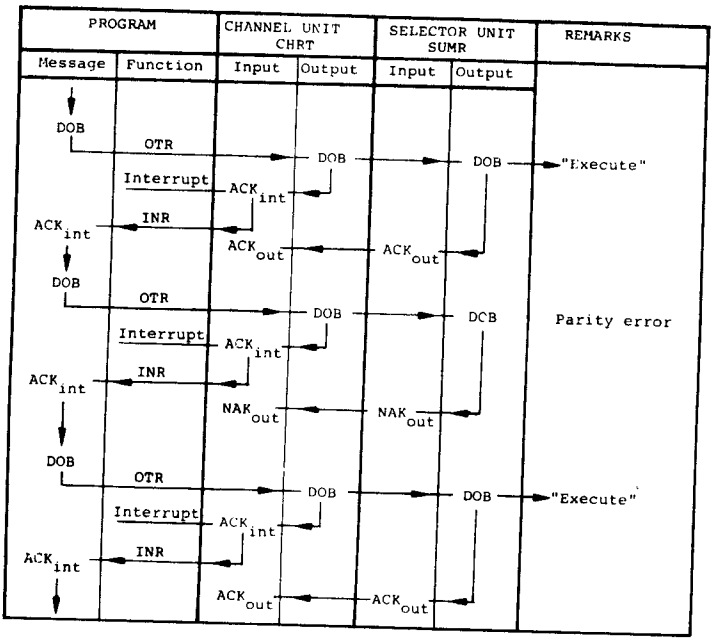


Figure 2.21. Block Output Procedure.

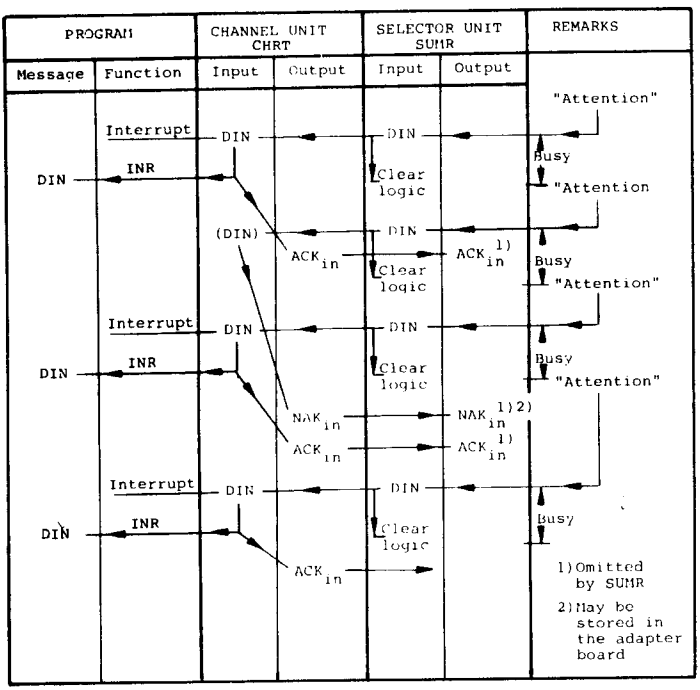


Figure 2.22. Block Input Procedure.

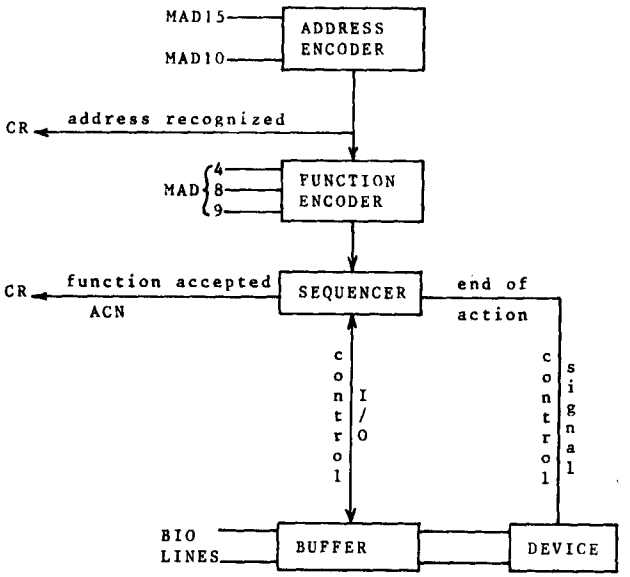


Figure 2.23. Channel Unit Internals.

3.1 GENERAL

The transfer unit (figure 3.1) makes it possible to connect two SUMR's to one modem.

Data can be transferred simultaneously in both directions, i.e. full duplex transmission.

On output from the CHRT, data is sent to both SUMR's at the same time, but only the one whose identity coincides with the terminal address in the character transferred will handle the data.

A terminal selector in the TFU enables one selector unit at a time to transfer data to the CHRT, and the selector unit which first made a request to send data will be enabled, whilst the other one is inhibited from transferring data until the first one is finished.

Before data can be transferred via the TFU the modem must be switched on by signals given by the TFU.

When in working state, the modem replies with three signals which are indicated on three lamps on the TFU front panel (figure 3.1).

The signals to/from the modem are standardized to meet the CCITT-V24 norms and are as follows:-

* Lamp 104 : Received Data.
* Lamp 107 : Data set ready.
* Lamp 109 : Carrier detected.

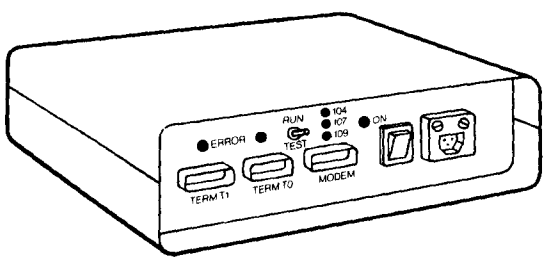


Figure 3.1. Transfer Unit (TFU).

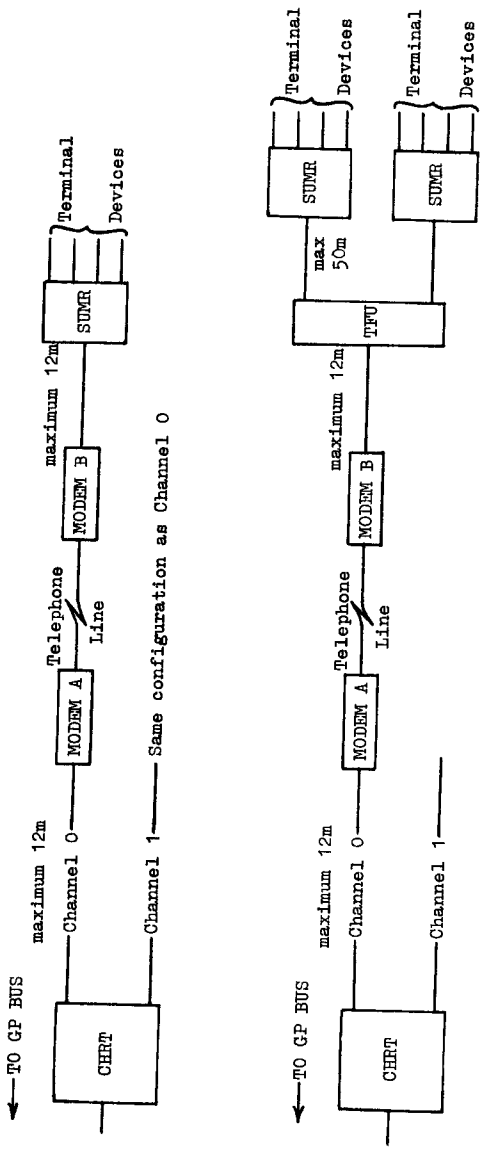


Figure 3.2. System configuration using CHRT, TFU, and SUMR.

3.2 OUTPUT DATA FROM CHRT

For data output, the TFU distributes data to both terminals simultaneously and it is up to the two SUMR's to ascertain for which the data is intended. This is done in the SUMR by comparing its identity with the terminal address in the transferred character.

If the SUMR detects parity failure in the received character, a signal is sent to the TFU which lights one of the error lamps, T0 or T1.

3.3 INPUT DATA TO CHRT

When one of the two SUMR's connected to a TFU wants to send data to the CHRT, the respective selector unit makes a 'Transmit Request' to the TFU. The TFU contains a terminal selector which replies 'Transmit Enable' if the other SUMR is not occupying the TFU. When input transfer from a SUMR starts, a 'Transmit Status' signal is sent to the terminal selector which inhibits the other SUMR from interrupting the transfer.

This means that for input, data can only be transferred from one SUMR at a time.

3.4 TEST

For testing the communication line and modem, the TFU has a switch marked RUN-TEST. With the switch in the TEST position, data from the CPU will return in the same form as it was sent, so that the CPU can compare input and output to see if the line is OK.

* RUN - Circuit 104.
  Received data is distributed to the SUMR.
  - Circuit 103.
  Transmitted data is transferred from SUMR to modem.

* TEST - Circuits 103 and 109 are disconnected from the logic inside the TFU. Circuit 103 is linked instead to 104. By this means, data received from the modem is returned to the modem and so transmitted back to the CPU.

Any normal data can be used as test data, but bit E must be set to 0. Bit E is the terminal address bit.

4.1 GENERAL

Two types of control panels are available to provide control and display facilities for the system. These are as follows:-

* System Operator's Panel (standard).

* Computer Full Panel (optional).

A Computer Full Panel may be fitted to PTS6810, 6812, or 6813 machines, but the PTS6813 has an extended computer full panel.

The System Operator's Panel (SOP) may be fitted and used independently of a full panel, whereas full panel operation requires that a SOP be fitted, or that certain facilities are available in the configuration.

4.2 SYSTEM OPERATOR'S PANEL (SOP)

Figure 4.1 shows the layout of the System Operator Panel. The functions of the panel are much the same for PTS6805, 10, 12, and 13, the only difference being that some types have switches, and some have buttons. Figure 4.2 illustrates two different types.

The functions provided by the SOP are described in the following sections.

4.2.1 Safety Key Switch

This is a three position key-operated switch providing control facilities. The three key positions are:-

* NO RTC

  In this position the key cannot be removed and the system is able to run with the Real Time Clock signal inhibited. All other control panel switches are effective.

* RTC

  In this position the key cannot be removed and the system is able to run with the Real Time Clock signal enabled. All other control panel switches are effective.

* LOCK

  In this position the key can be removed and the system is able to run with the Real Time Clock signal enabled. The Bootstrap/IPL switch on the SOP, and all switches on the Computer Full Panel, except the Interrupt buttons, are inhibited.

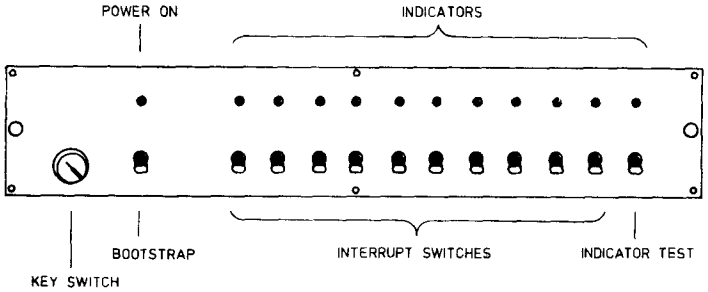


Figure 4.1. System Operator's Panel (SOP).

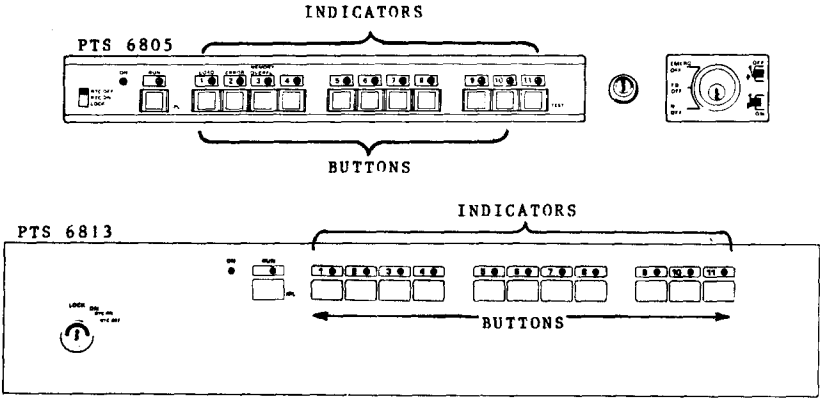


Figure 4.2. PTS 6805 and 6813 SOP's.

4.2.2 Display Indicators

Twelve Indicators are provided, as follows:-

* Power-On Indicator

  This indicator is mounted on the left of the panel, above the Bootstrap/IPL switch. It is used when the SOP is the only panel fitted, to indicate that the basic unit's mains switch is ON, and that power is being supplied to the unit. When the Computer Full Panel is fitted, the SOP's power-on indicator has no function and power-on is indicated on the full panel.

* System Indicators

  Eleven indicators are mounted to the right of the power-on indicator. These may be used as required by the system software and are lit or extinguished by program instructions.

4.2.3 Operator Toggle Switches or Buttons

Twelve single-throw toggle switches or buttons are provided:-

* Bootstrap/IPL

  The bootstrap/IPL switch is mounted to the right of the safety key switch, beneath the power-on indicator, and is operative when the safety key switch is in the NO RTC or RTC position. When operative and depressed the bootstrap switch initiates IPL action as follows:-

  - The bootstrap is loaded into main memory.

  - Current loading parameters are copied into CPU register A15.

  - The CPU is started in RUN mode.

  When using the IPL function, loading may take place from cassette, flexible disk, cartridge disk, or fixed disk.

* Interrupt Switches

  Ten interrupt switches are mounted to the right of the bootstrap/IPL switch. The switches are operative in all positions of the safety key switch and may be used to generate an interrupt to the running program. As the SOP is controlled within the system as an input/output device, program instructions are required to allow interrupts from the switches and to determine which of the switches caused the interrupt.

  The basic interrupt level raised by the switches is the same for all of them, normally level 9, and a hardware priority system within the panel distinguishes between them, giving priority from left to right to the ten switches. If more than one of the switches are depressed simultaneously, only one interrupt is raised. The panel responds with an indication of the highest priority switch depressed when it is interrogated following interrupt action. Following such an action, a second interrupt can only be raised after the release of all the switches depressed.

* Indicator Test Switch

  This switch is mounted on the right of the panel beneath the last system indicator, and provides a test facility for the system indicators. When depressed the switch causes all these indicators to be lit if they are in working order.

4.2.4 Test Panel

Since the SOP functions as an input/output device controlled directly by the channel unit for cassette tape, CHCR, there are certain programming requirements for correct functioning with respect to the operating modes of the channel unit.

The standard addresses, interrupt levels, and the program instructions accepted as control commands to the SOP are as follows:-

* Addresses

  - Standard Address = X'2E'.

  - Alternative Address = X'2F'.

* Interrupt Levels

  - Standard Interrupt Level = Decimal 9.

  - Alternative Interrupt Level = Decimal 11.

* Accepted Instructions

  - OTR.

  This instruction is always accepted and is used to control the system indicators. The value in the register specified by R3 in the instruction is used to select them.

  R3, bits 0 to 4 - Not used.

  R3, bits 5 to 15 - Select indicators 11 to 1 respectively.

  Note The indicators are numbered 1 to 11 from left to right, and a 1 bit set in the appropriate position in register R3 is used to light the corresponding indicator.

  - CIO Start.

  This instruction is used to enable the SOP to raise an interrupt and accept INR instructions. It must be executed at the beginning of the program, or after a CIO Halt instruction has been executed, if interrupts and INR instructions are to be accepted. The contents of the register specified by R3 in the instruction are not used and remain unchanged.

- CIO Halt.

This instruction is used to inhibit interrupts from the SOP. INR instructions sent to the panel after the execution of a CIO Halt instruction, and before the execution of a subsequent CIO Start instruction, will be rejected. The contents of the register specified by R3 in the instruction are not used and remain unchanged.

- INR.

This instruction will be accepted if the SOP has been sent a CIO Start instruction and no subsequent CIO Halt instruction, and providing that the CHCR is not still executing a previously received INR instruction for the panel. The instruction is used to identify the interrupt switch which has caused an interrupt, by transferring a 1 bit into the appropriate position of the register specified by R3 in the instruction.

R3, bits 0 - 5 - Insignificant (set to 1's).

R3, bits 6 - 15 - Interrupt switches 21 to 12.

Note The interrupt switches are numbered 12 to 21 from the left of the panel, switch 12 having the highest priority.

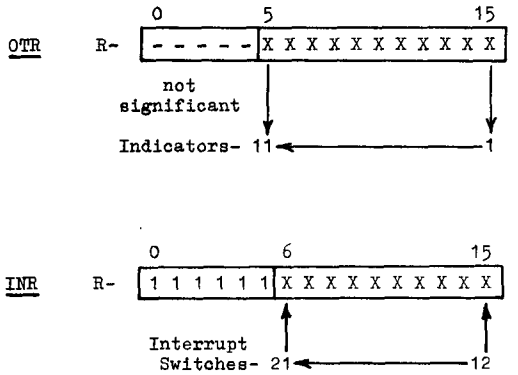


Figure 4.3. Relationship between the indicators and interrupt switches and the bit pattern of the register specified by R3.

4.3 COMPUTER FULL PANEL

Figure 4.4 shows the layout of the Computer Full Panel as fitted to a PTS6805, 10, or 12. The panel may be fitted in conjunction with the SOP to provide additional control and display facilities for system operators and service engineers.
If a panel is not fitted, a blank cover will be fitted in its place.

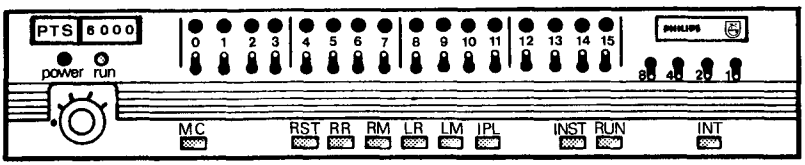


Figure 4.4. PTS6810 or 6812 Computer Full Panel.

The facilities provided by the full panel are listed below.

4.3.1 Display Lamps

Eighteen display lamps are provided as follows:-

* Power-On Lamp

  This lamp is mounted on the left of the panel and is used to indicate that the basic unit's mains switch is ON and that power is being supplied to the unit.

* Run Lamp

  This lamp is mounted to the right of the power-on lamp and is lit when the CPU is operating in RUN mode.

* Data Lamps

  There are sixteen of these, situated one above each data switch and numbered 0 thru 15. The lamps are lit to indicate the contents of the registers, memory, or status word depending on the settings of other control switches. A 1 bit is indicated where a lamp is lit.

4.3.2 Data Switches

There are sixteen numbered data switches mounted centrally on the panel. These are two-position switches used for loading data bits into a register or memory, depending on the setting of other control switches. A 1 bit is loaded when a switch is in the up position.

4.3.3 Register Address Switches

Four switches for register addressing are mounted to the right of the data switches, and they are used to code the address of the register to be used when reading or loading a register from the Data Switches. The switches are labelled with the decimal equivalent of the address value they represent in binary (8, 4, 2, 1) giving an addressing capability of 0 - 15.

4.3.4 Control Buttons

The five control buttons are situated centrally below the Data Switches. The buttons are spring-loaded to return to their original positions after being depressed and each selects and initiates a specific function, as follows:-

* RST - Read Status.

  Depressing this button causes the Program Status Word to be displayed on the lamps. Figure 4.5 shows the format.

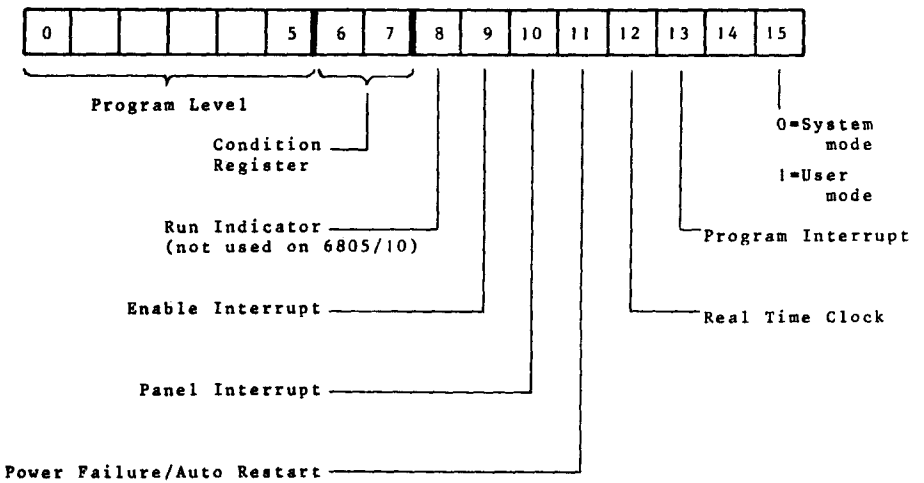


Figure 4.5. PSW display format.

* RR - Read Register.

  Depressing this button causes the contents of the register addressed by the Register Address Switches to be displayed on the Data Lamps.

* RM - Read Memory.

  Depressing this button causes the contents of the memory location addressed from the contents of the P-Register to be displayed on the Data Lamps. The contents of the P-Register are also incremented by 2.

* LR - Load Register.

  Depressing this button causes the value set on the sixteen Data Switches to be loaded into the register addressed by the Register Address Switches. The value is also displayed on the Data Lamps.

* LM - Load Memory.

  Depressing this button causes the value set on the sixteen Data Switches to be loaded into the memory location addressed from the contents of the P-Register. The value is also displayed on the Data Lamps and the P-Register is incremented by 2.

4.3.5 Mode Buttons

The two mode buttons are situated below Data Switches 14 and 15. They select and initiate the following modes of operation:-

* INST - Single Instruction Mode.

  Depressing this button causes the CPU to increment the contents of the P-Register, execute the instruction addressed from its contents, and then stop.

* RUN - Run Mode.

  Depressing this button causes the CPU to execute the instructions of a program commencing at the instruction addressed from the contents of the P-Register plus 2.

4.3.6 Service Buttons

There are three service buttons, listed below:-

* MC - Master Clear.

  Situated below Data Switch 0. Depressing this button raises the master-clear level throughout the system, causing a general reset of all the associated logic.

* INT - Interrupt.

  Situated below the Register Address Switches. Depressing this button raises a control panel interrupt. This button is the only control on the full panel which is operative when the Safety Key Switch of the SOP is in the LOCK position.

* IPL - Initial Program Load.

  Situated below Data Switch 10. This button provides the same automatic program loading facility as the bootstrap switch on the SOP. When the button is depressed it causes the Initial Program Loader to be loaded into central memory and the CPU to be started. Loading is carried out from the device specified by pressing the corresponding SOP switch/button (figure 4.6).

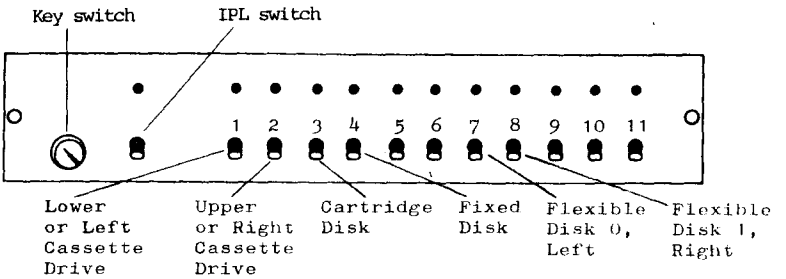


Figure 4.6. SOP switches used for program loading from various media.

4.4 COMPUTER FULL PANEL OPERATION

The full panel provides memory and register loading and display facilities, and program execution may be started and controlled manually.

4.4.1 Loading and Display Facilities

Figures 4.7 and 4.8 show the procedures for loading and displaying the contents of the general purpose registers and of memory locations. The CPU Program Status Word may be displayed by depressing the RST button.

4.4.2 Manual Program Control

In addition to the use of the IPL facility, programs may be started, checked, tested, and altered using the load and display facilities. The manual starting of any program whose start address is known is as follows:-

* Using the load register routine (figure 4.7), load register A0 with the required start address.

* If the program is to be run continuously, press the RUN button.

* If the program is to be run one instruction at a time, press the INST button.

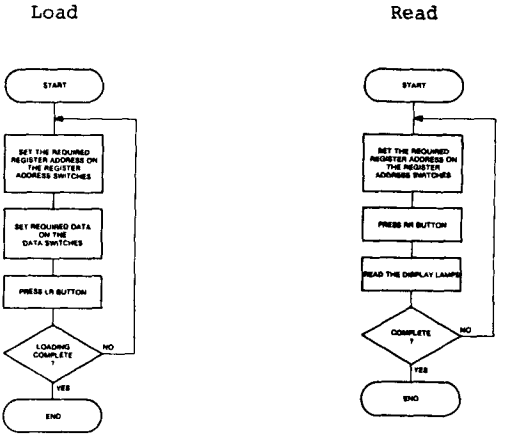


Figure 4.7. Computer Full Panel, Load and Read Register routines.

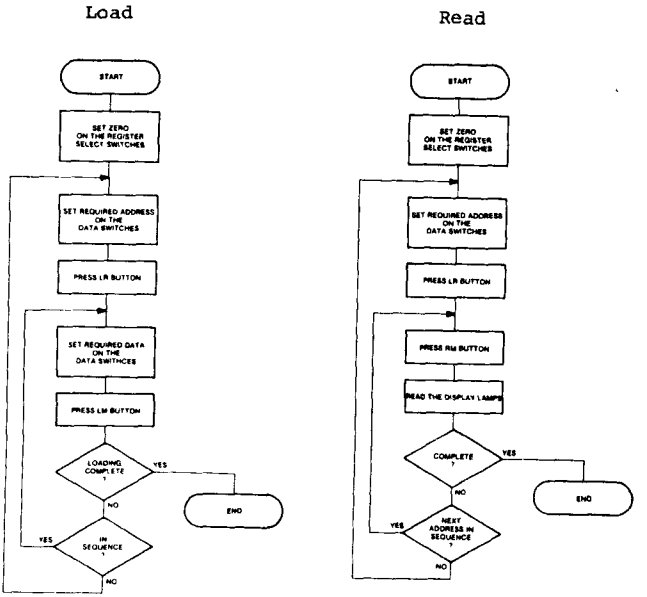


Figure 4.8. Computer Full Panel, Load and Read Memory routines.

4.5 EXTENDED CONTROL PANEL

The Extended Control Panel (figure 4.9) may be fitted to a PTS6813 machine.

This panel provides the facility for displaying an address and its contents at the same time. It also provides more extensive debugging facilities, since processing may be stopped at any address set previously on the upper row of switches. The user may then load new data if so desired.

Addressing from this panel is in terms of words.

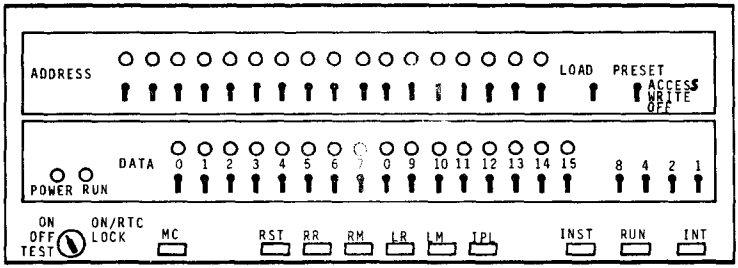


Figure 4.9. Extended Control Panel.

The functions of the displays and switches on the Extended Control Panel are as follows:-

4.5.1 Display Lamps

Seventeen lamps situated above seventeen address switches; when the computer is running, the lamps on the upper part of the panel are lit to indicate addresses, and the contents are displayed on the lower lamps. When the CPU stops, the contents of the next instruction's address is displayed on the lower lamps, and the address of the instruction on the upper lamps.

4.5.2 Address Switches

Seventeen address switches on the upper part of the panel; each switch has two positions, and they are used for loading the appropriate address. A 1 bit is loaded when a switch is in the up position.

4.5.3 LOAD Switch (spring-loaded)

Used to load an address in an address register contained in the control panel.
The required address is set on the address switches. The LOAD switch is pressed downwards and the address is displayed on the address lamps.

4.5.4 PRESET Switch

A three position switch useful for debugging purposes.

* ACCESS - stop on memory address coincidence.
  In this position the CPU stops when a physical address generated by the CPU is identical to the pattern coded on the address keys. When that address is detected, the relating instruction is executed and the CPU enters the idle state.

* WRITE - stop when writing into memory at selected address.
  In this position the CPU only stops if a store operation is performed at the location whose address was set previously on the address keys.

* OFF - disables debugging facilities described above.

4.5.5 Read Memory Procedure

1. Load the address register with the address wanted (see LOAD above).

2. Press RM button. The contents of the memory location will be displayed on the data lamps.

3. The control panel register is incremented by two and the next address is displayed on the address lamps.

Each time the RM button is pressed the address register is incremented.

Figure 4.10 shows the memory display procedure in flowchart form.

4.5.6 Load Memory Procedure

1. Load the address register with the address wanted (see LOAD above).

2. Set the value to be loaded on the data switches.

3. Press LM button.
   The value is displayed on the data lamps. The address register is incremented by 2 and displayed on the address lamps.

Figure 4.11 shows the memory loading procedure in flowchart form.

All other push-buttons and switches have the same functions as described above in the Computer Full Panel section, 4.3.

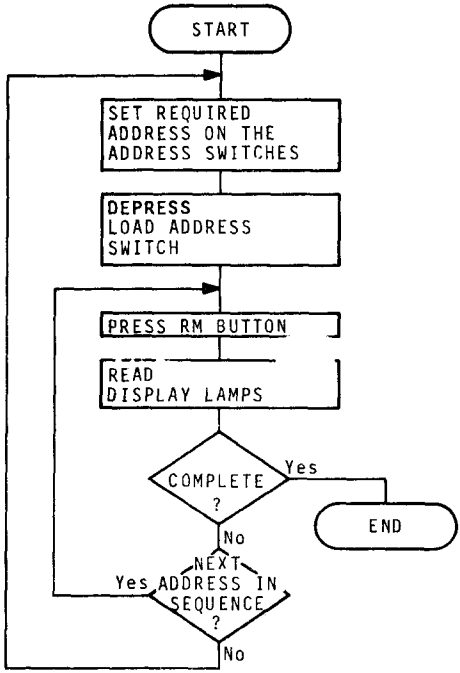


Figure 4.10. ECP procedure for displaying the contents of memory.

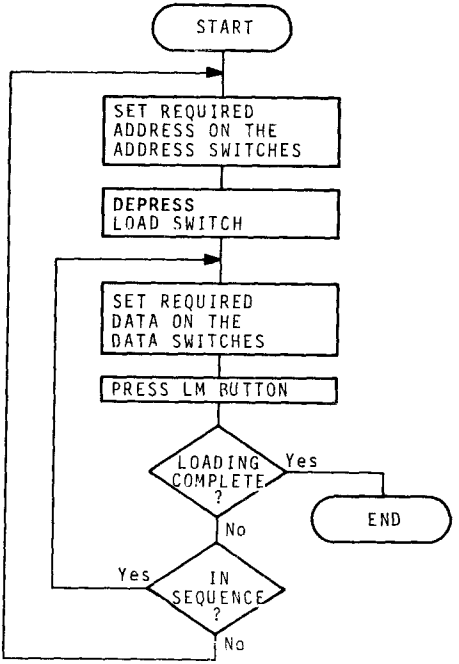


Figure 4.11. ECP procedure for loading data into memory.

The Initial Program Loader provides the system with the ability to load programs from standard cassette tape, flexible disk, cartridge disk, or fixed disk.

The IPL facility may be initiated either from the computer full panel or from the system operator's panel (SOP).

When the key switch is not in the LOCK position, operation of the IPL switch or button causes the following events to occur:-

* The contents of the IPL ROM are copied into the first 64 words of main memory.
* The CPU is put into 'Inhibit Interrupt' mode.
* The P-register is zeroised and the CPU is started in 'Run' mode in order to execute the program just loaded from the ROM. In other words, 'bootstrap'.

The bootstrap program which is executed will identify the SOP switch which has been depressed, and the SOP switch number will be used to select a device. Possible devices are cassette 1, cassette 2, fixed or exchangeable disk, flexible disk 0, or flexible disk 1.

A loader program is copied from the selected device into main memory, starting at address X'0080'.

After completion of this copy, control is given to the loader program at address X'0084'.

This loader program then moves itself to address X'FFFE' and lower, and proceeds to load the monitor and applications.

See figure 5.1.

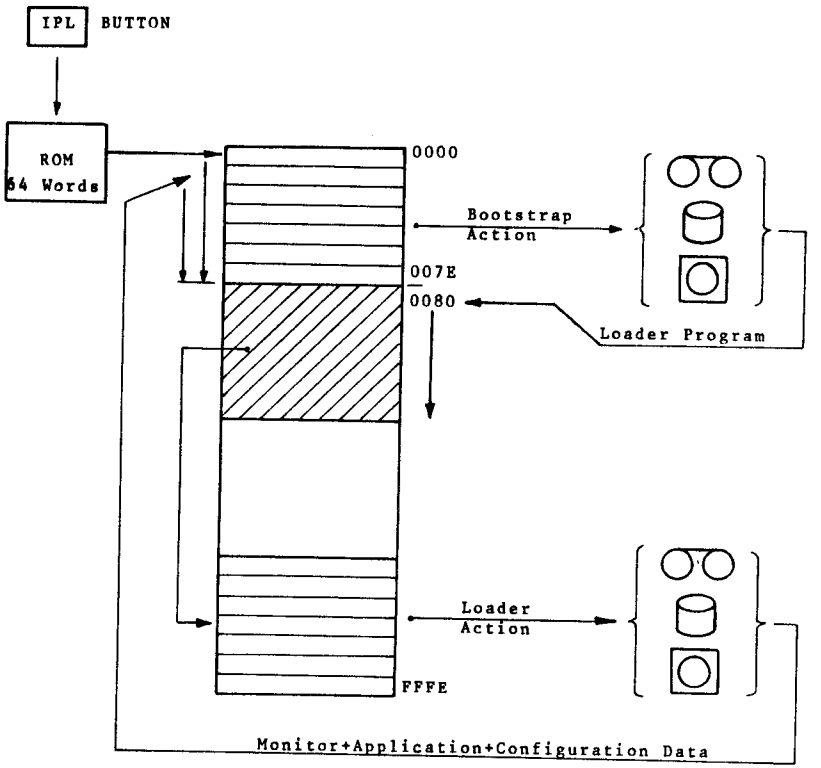


Figure 5.1. IPL, Bootstrap, Loader.

The trap facility is provided so that instructions which are unrecognisable to the hardware can be dealt with in an orderly way. Whenever an unrecognisable instruction is detected, whether in the initial decode stages or in the addressing stages, the trap is sprung as follows. The first three steps are common to the handling of all interrupts:-

* Execution of the instruction is stopped.
* The address of the instruction and the Program Status Word are stored in the system stack.
* The CPU is put into 'Inhibit Interrupt' mode.
* The trap interrupt-handling routine is started by loading the P-register with the start address of that routine, which is held at address X'007E'.
* This trap routine causes the running program to be stopped by executing a HALT instruction.

Note Certain development and other specialized system software may use the trap routine for instruction simulation. In such cases the routine would not stop the program, but would execute a simulation routine before returning to continue normal processing.

The trap facility is supported only on a PTS6810 configuration, and only when all routines are written in Assembly language. 'No Credit' must have been specified at SYSGEN time.

The addresses are set by a combination of fixed and changeable jumpers situated in each unit.

ADDRESS HEXADECIMAL	PTS 6000/8000	PTS 6810 CONCENTRATOR
00	ASC-6	
01	CHRT1/CURT1/ASC-6	CHRT 1
02	CHLC IN/ASC-6	LINE CONTROL IN 1
03	CHLT1/CULT1/ASC-6	CHLT 1
04	MFD-1	LINE CONTROL IN 3
05		LINE CONTROL IN 4
06		LINE CONTROL IN 5
07		LINE CONTROL IN 6
08	F/R DISK-1	DISC UNIT 1
09	FLEX DISK 1/CASSETTE CHANGER	FLEX DISK
0A	LINE CONTROL IN (6805/8000)	Line control trunk
0B	LINE CONTROL OUT (6805/8000)	Line control trunk
0C	MAG TAPE/ASC-5	MAG TAPE 1
0D	CARD READER/ASC-5	MAG TAPE 2
0E	TAPE CASSETTE/LP2/ASC-5	TAPE CASSETTE
0F	LINE PRINTER/ASC-5	LINE PRINTER
10	TYPEWRITER	TYPEWRITER
11	CHRT2/CURT2	CHRT 2
12	CHLC OUT	LINE CONTROL OUT 1
13	CHLT2/CULT2	CHLT 2
14	MFD-2	LINE CONTROL OUT 3
15		LINE CONTROL OUT 4
16		LINE CONTROL OUT 5
17	80MB DISK-1	LINE CONTROL OUT &
18	F/R DISK-2	DISK UNIT 2
19	FLEX DISK 2	
1A	SALC-1 IN	
1B	SALC-1 OUT	
1C	ASC-1	
1D	ASC-1	
1E	ASC-1	
1F	ASC-1	

ADDRESS HEXADECIMAL	PTS 6000	PTS 6810 CONCENTRATOR
20	ASC-3	LINE CONTROL IN 7
21	CHRT3/CURT3/ASC-3	LINE CONTROL IN 8
22	ASC-3	LINE CONTROL IN 2
23	CHLT3/CULT3/ASC-3	LINE CONTROL IN 9
24	MFD-3	LINE CONTROL IN 10
25		LINE CONTROL IN 11
26		LINE CONTROL IN 12
27		LINE CONTROL IN 13
28	F/R DISK-3	DISK UNIT 3
29	FLEX DISK 3	
2A	SALC-2 IN	
2B	SALC-2 OUT	
2C	SALC-3 IN	
2D	SALC-3 OUT	
2E	SOP	SOP
2F		
30	ASC-4	LINE CONTROL OUT 7
31	CHRT4/CURT4/ASC-4	LINE CONTROL OUT 8
32	ASC-4	LINE CONTROL OUT 2
33	CHLT4/CULT4/ASC-4	LINE CONTROL OUT 9
34	MFD-4	LINE CONTROL OUT 10
35		LINE CONTROL OUT 11
36		LINE CONTROL OUT 12
37	80MB DISK-2	LINE CONTROL OUT 13
38	F/R DISK-4	DISK UNIT 4
39	FLEX DISK 4	
3A	SALC-4 IN	
3B	SALC-4 OUT	
3C	ASC-2	
3D	ASC-2	
3E	ASC-2	
3F	ASC-2	

Note Some devices can not be used in all of the system types.

ASC = ASCU4Z; SALC = SALCUZ.

Figure 8.1. Standard Channel Unit Addressing.

For ASCU4Z and SALCUZ, Channel Unit Addresses and Line transmission speeds can be selected by jumpers on the cards. For ASCU4Z the line speed is normally set to 9600 b/s and there are four possible connection configurations:-

* 4 Displays on lines 1 - 4 (4 half-duplex).

* 2 Displays/Keyboards or 2 Printers on lines 2 and 4 (2 full-duplex).

* 2 Displays on lines 1 and 2 (2 half-duplex) and 1 Display/Keyboard or 1 Printer on line 4 (1 full-duplex).

* 1 Display/Keyboard or 1 Printer on line 2 (1 full-duplex) and 2 Displays on lines 3 and 4 (2 half-duplex).

SALCUZ transmission speeds may be 1200, 2400, 4800, or 9600 b/s.

	Line 1	Line 2	Line 3	Line 4
ASCU4Z-1	1C	1D	1E	1F
ASCU4Z-2	3C	3D	3E	3F
ASCU4Z-3	20	21	22	23
ASCU4Z-4	30	31	32	33
ASCU4Z-5	0C	0D	0E	0F
ASCU4Z-6	00	01	02	03

	Input	Output
SALCUZ-1	1A	1B
SALCUZ-2	2A	2B
SALCUZ-3	2C	2B
SALCUZ-4	3A	3B

Figure 8.2. ASCU4Z and SALCUZ Channel Unit Addresses.

9.1 GENERAL

The programmed channel is the basic PTS I/O facility and is a standard part of all systems. Its function is to control the flow of data between peripheral channel units and the CPU.

It is also used as the initialization path between CPU and I/O processors, if present in the system. In all uses of the programmed channel, the CPU is the controlling unit.

9.2 I/O INSTRUCTIONS

Data is transferred via the G.P. bus under the control of a program written using the following instructions, which may only be issued in system mode.

* CIO Control Input/Output. Start or Stop an I/O operation.
* INR Input to Register. Transfer one word or character from a Channel Unit buffer to a CPU register.
* OTR Output from Register. Transfer one word or character from a CPU Register to a Channel Unit buffer.
* SST Send Status. A Channel Unit status word is set in a CPU register.
* TST Test Status. Test whether a Channel Unit is busy.

The transfer of each word or character requires a separate instruction and so program loops are used to transfer blocks of data. The programmed channel has two modes of operation, namely Inhibit (or Wait) Mode, and Interrupt Mode.

9.3 CHANNEL UNIT STATES

Once a Channel Unit has been activated by the programmed channel, it is capable of transferring data to or from a peripheral. Processing may continue while I/O is in progress between Channel Unit and device. The following description is applicable to a CHLT or CHRT. Details of the operation of other channel units may be found in the appropriate System Engineering manual.

As shown in figure 9.1, a Channel Unit may be in one of four possible states, as follows:-

* Inactive The unit is idle, and the only instruction it can accept in this state is CIO. Switching to Execute state will occur on reception of CIO Start.

* Execute The unit is operable and transfers data from a peripheral device to a buffer in the Channel Unit, or vice-versa. Unless an error occurs, or a CIO Halt is received, switching will occur to Exchange state on completion of Execute for input. For output, the Channel Unit remains in Execute state.

* Exchange The unit is operable and generates an interrupt to the CPU when this state is entered. The interrupt handler responds with an INR, and data transfer takes place from a Channel Unit buffer to a CPU register. On completion of Exchange a switch will occur to Execute state, unless CIO Halt is received, or an error is detected.

* Wait (For Send Status). An interrupt to the CPU is generated on entry to this state. Wait state is entered from Execute or Exchange states following a CIO Halt, or an error condition. The interrupt handler responds to the interrupt with an SST instruction, and switching then occurs to Inactive state.

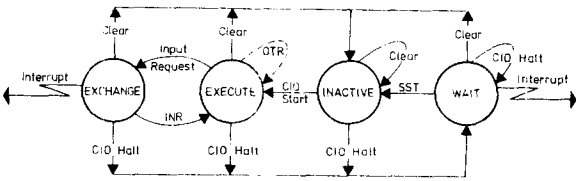


Figure 9.1. Channel Unit States and State-switching.

9.4 PROGRAMMED CHANNEL, INHIBIT (OR WAIT) MODE

In Inhibit Mode the instructions necessary to perform the required I/O are included in the program which needs that I/O. Interrupts are disabled in this mode.

Figure 9.2 depicts a possible I/O sequence in Inhibit mode. Events proceed from left to right in the diagram. Assume that the Channel Unit is in Inactive state at the start of the sequence, e.g. after power-on.

1. The program issues a CIO Start instruction and then waits. This causes the Channel Unit to switch into Execute state and transfer a word from the relevant Selector Unit to the Channel Unit buffer. Exchange state is then entered so that the transfer can be completed. An interrupt is sent to the CPU but is not acted upon because interrupts are inhibited.

2. The program now issues an INR instruction which, when accepted by the Channel Unit in Exchange state, causes a word or character to be transferred from the Channel Unit buffer to a Register in the CPU. After execution of the INR, the Channel Unit switches to Execute state and transfers another word or character, if available, from device to Channel Unit buffer. A switch is made back to Exchange state on completion of Execute, and another interrupt is generated but not acted upon.

3. Repeat (2).

4. The program has now performed all the input transfers it requires and issues a CIO Halt instruction. This switches the state of the Channel Unit from Exchange to Wait, and again a non-actioned interrupt is generated. An SST instruction in the program now switches the Channel Unit back to Inactive state ready for the next CIO Start instruction.

5. The program has now obtained the data it requires and may continue processing.

An important point to notice about Inhibit mode is that the CPU is forced to wait while I/O operations between Channel Unit and device are in progress, and so it is idle for that time. This can be largely overcome by exploiting the interrupt system and using the other mode of the programmed channel, namely Interrupt Mode.

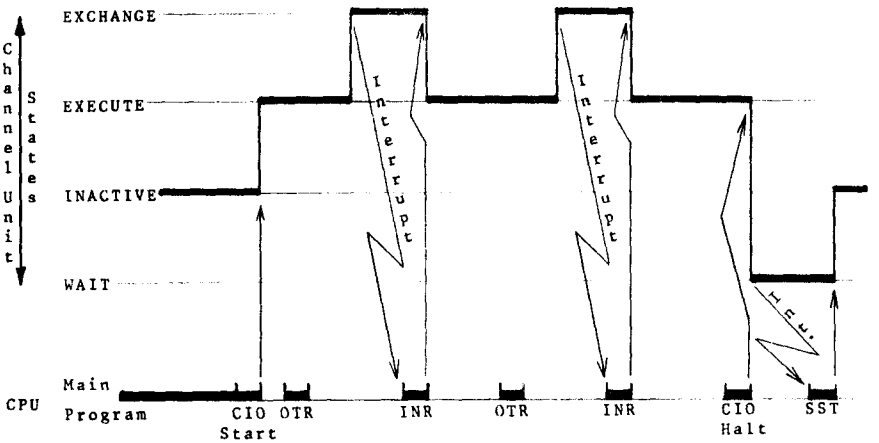


Figure 9.2. Programmed Channel, Inhibit Mode Sequence.

9.5 PROGRAMMED CHANNEL, INTERRUPT MODE

Unlike Inhibit Mode, the instructions required for I/O in this mode are not included in the Main program, but are part of a special Interrupt Routine, invoked by interrupt signals raised by the Channel Unit hardware. Interrupts must, of course, be enabled.

1. The program issues a CIO Start instruction and continues with processing. The Channel Unit switches into Execute state and a word can be transported from a peripheral device to the Channel Unit buffer. After receipt of the word a switch is made to Exchange state, and this generates an interrupt to the CPU.

2. Control passes from the main program to the interrupt routine, which issues an INR instruction. The Channel Unit, in Exchange state, accepts this and transfers the word or character from the Channel Unit buffer to a CPU register. The main program resumes processing after the INR is completed. The Channel Unit switches to Execute state and transfers another word from device to CU buffer, whereupon a switch is made back to Exchange state, and another interrupt is generated.

3. Repeat 2.

4. All required input has now been performed, and the interrupt routine issues a CIO Halt instruction. The Channel Unit switches from Exchange to Wait state and generates another interrupt to the CPU. The main program is suspended and the interrupt routine gains control and issues an SST instruction which switches the Channel Unit back to Inactive state.

5. The main program resumes processing until it requires the data, which is ready to be used in memory at this point.

This example assumes that the program is able to continue processing while the data is being retrieved. If the I/O had not been completed before it was required by the program, it would have been necessary to include coding so that the program would wait for completion.

It is important to note that, unlike Inhibit Mode, the main program is not held up waiting for the Channel Unit. As long as there is processing to be done, it can carry on simultaneously with Channel Unit/device action, except for short interruptions by the interrupt routine which contains the I/O instructions.

Note that upon CIO Start for output, the Channel Unit switches from Inactive to Exchange, not Execute. This is because, for output, the first part of the transfer path is from CPU to Channel Unit buffer, whereas for input, it is from device to Channel Unit buffer.

10.1 CONFIGURATION

As seen in chapter 9, the operation of the Programmed Channel imposes a load upon the CPU because it has to execute instructions in order to effect the required transfers. The Input Output Processor (IOP) relieves the CPU of this load by performing the data transfers in its hardware. This offers a faster method of block data transfer between memory and peripherals, and allows the CPU, after initializing the IOP, to continue processing while the I/O is handled by the IOP. The G.P. Bus remains the data path used. Initialization of the IOP is carried out by the CPU using the ordinary programmed channel.

Channel unit states and sequences are as described for the programmed channel in Chapter 9 except that the interrupt raised in exchange state is replaced by a 'break' signal wired directly to the IOP from a channel unit. Figure 10.1 shows that, when the programmed channel is used, the wait and exchange state interrupts are wired together. When an IOP is used these two signals are separated, and the exchange state interrupt is wired directly to the IOP and known as a 'break', as shown in figure 10.2.

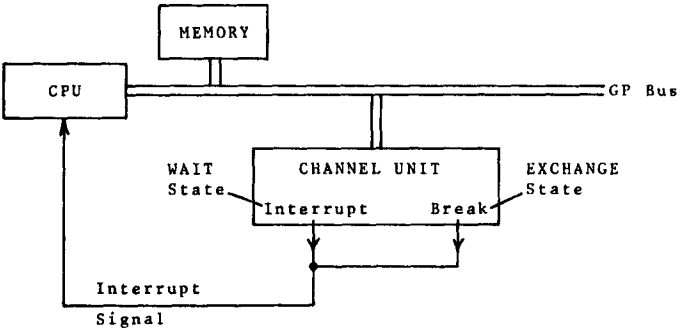


Figure 10.1. Channel Unit Interrupt Lines, no IOP.

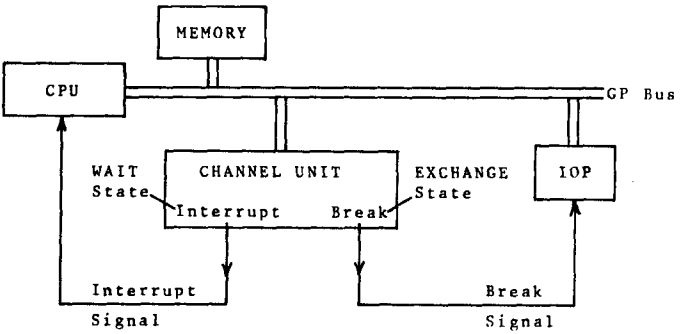


Figure 10.2. Channel Unit Interrupt Lines with IOP.

Up to 8 IOP's may be configured in a PTS system, each of which is able to control up to 8 channel units. Each IOP is known as a channel, and its 8 subdivisions are known as sub-channels. Associated with each sub-channel is a pair of 16-bit registers which are used to hold information relating to the transfer to be carried out by that sub-channel and its channel unit. These registers are referred to as control registers and they hold the transfer parameters which are used and updated by the IOP during transfer operations.

Sub-channels are allocated priorities from 0 thru 7 so that, in the event that more than one break signal is present at a particular moment, the IOP deals with the channel unit having the highest priority break signal outstanding, 0 being the highest priority level. The address of a channel unit which is connected to an IOP consists of 6 bits as shown in figure 10.3.

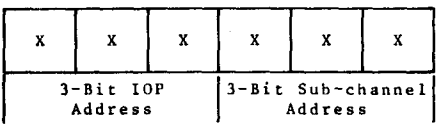


Figure 10.3. Address of a Channel Unit attached to an IOP.

10.2 INSTRUCTIONS AND CONTROL WORDS

Before a transfer is initiated the two registers in the IOP must be correctly set up.

Two instructions are available to write to or read from the control registers in the IOP. They are restricted to use in system mode, and are as follows:-

* WER Write External Register.
  The contents of a specified CPU Register are set into a specified IOP Register.

* RER Read External Register.
  The contents of a specified IOP Register are set into a specified CPU Register.

Layouts of these instructions are shown in figure 10.4, and those of the Control Words in figure 10.5.

In figure 10.4, Channel Address is the address of the IOP, of which there may be 8 in the system, and Sub-channel Address is the priority level of the channel unit which is to be used in the transfer. The CPU register involved in the operation is defined by the 3 bits, R.

WER is used in the setting up of an IOP for a transfer, and RER to check the control words at transfer end, or in the event of an error.

Any or all of the 8 sub-channels in an IOP may be in use at a time, and there is no need to check the status of any sub-channel before using the IOP, except the sub-channel which it is intended to activate. The status of the desired sub-channel should always be checked because a WER instruction is always actioned by the CPU, even if the sub-channel is already busy, and this would result in any transfer in progress being corrupted.

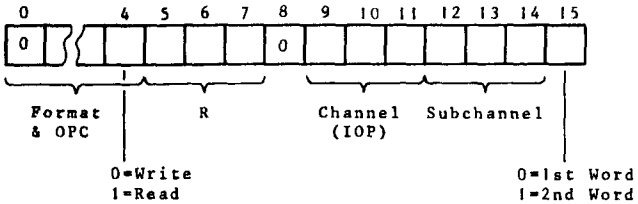


Figure 10.4. Layouts of WER and RER instructions.

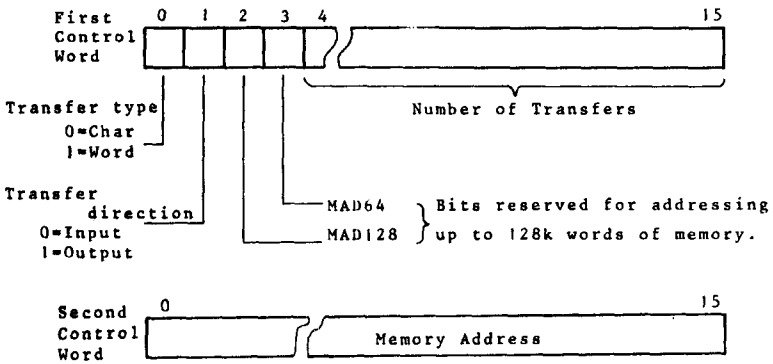


Figure 10.5. Layouts of Control Words.

If the required sub-channel is found not to be busy, the program may set up the two control registers using two WER instructions and start the channel unit concerned using a CIO Start instruction.
The channel unit switches to execute state for input, or to exchange for output. After receiving a character or word in execute state, a switch is made to exchange. In exchange state the channel unit emits a break signal to the IOP.

The transfer will now carry on independently without any further program action until it is complete and the channel unit switches into wait state and issues an interrupt to the CPU. The interrupt routine for the relevant channel unit will intercept the interrupt signal and switch the unit into inactive state if necessary, by issuing an SST instruction.

If the status returned by the channel unit indicates that an error condition exists, the contents of the IOP's control registers may be accessed (by RER) and used in either error recovery or error reporting routines.

10.3 MODES OF OPERATION

Three separate control paths are used during an IOP transfer, and the IOP may be in one of three distinct modes of operation associated with each data flow path.

1. Scan Mode.
   The IOP is scanning its Break lines for pending I/O requests.

2. CPU to IOP (CPU Mode).
   The CPU is the master of the exchange and is issuing WER or RER instructions to the IOP in order to initialize it for a transfer, or establishing the status of a completed transfer.

3. IOP to Memory or Channel Unit (Exchange Mode).
   The IOP is the master of the exchange and is controlling a transfer between memory and a channel unit or vice-versa. In this mode the IOP is responding to a break signal raised by a channel unit in exchange state, and is updating the transfer parameters held in its control registers. On completion of a transfer the channel unit generates an interrupt and the IOP reverts to Scan mode unless another break signal is outstanding, in which case it remains in exchange mode.

If a power failure occurs when the IOP is master of the exchange, the current exchange is aborted or completed and control of the G.P. Bus is returned to the CPU, regardless of outstanding break signals.

11.1 GENERAL DESCRIPTION

TOSS (Terminal Operating System Software) is the operating system used in the full range of PTS systems.

The TOSS monitor is a real time monitor with the ability to control a number of independent tasks running at different priority levels.

According to one definition, a monitor is:-

> 'A collection of routines stored permanently in memory, which control the operation of user programs and coordinate the various hardware and software activities; synonymous with executive program. Such a system controls the allocation of Store and peripheral units to programs, the loading and scheduling of programs, time sharing of input/output operations, and multitasking.'

Four main functional blocks make up the major part of TOSS:-

* Dispatcher
* Monitor tables
* Monitor processors
* I/O drivers and interrupt handlers

The dispatcher allocates CPU resources to the different tasks and monitor processors.

The TOSS Monitor is table oriented, which means that special monitor tables are used for describing the system and terminal configurations. Monitor tables also contain work areas for devices, status information related to each device and task, pointers and queue links for handling different queues, etc.

The following tables are described in Chapter 15:-

* SCT System Control Table
* TCTAB Task Control Table
* TTAB Task Table
* CDTAB Common Device Table
* DWT Device Work Table
* DAB Driver Address Block

Monitor tables are necessary for, among other things, the monitor processors, to allow them to perform their functions for different tasks, e.g. I/O, Wait, Exit, Activation, etc.

The requested monitor processors are included and the required tables built up from the information supplied to SYSGEN (System Generation program) and SYSLOD (System Loading).

I/O drivers and interrupt routines are responsible for all communication with the devices in the system. There is also one driver that is not related to any specific device in the system, the Intertask Communication driver.

All drivers have their own specifications with full descriptions of calling sequences, return codes, and system adaptations specific to each driver. These details are described in the Assembler Programmers' Reference Manual, Part 2.

In systems with MMU some special tables are added to the monitor:-

* SEGTAB Segment block table.
* PAGTAB Page block table.

See Chapter 18 for details.

An 'interrupt' is an event which causes control to be passed, at the completion of the current instruction, to an address held in one of the 'interrupt vectors' in memory words 0 to 63. For example:-

* Power failure.
* LKM request.
* Real time clock update.
* Completion of I/O action.
* Attempted Execution of an illegal instruction.

The particular vector used depends upon the type of interrupt. Each vector contains a pointer to an associated 'interrupt handler'.

For example, memory word 0 is the interrupt vector for power failure. When power failure occurs the sequence of instructions currently being executed is interrupted, and control is handed to the instruction pointed to by memory word 0 (the power failure interrupt handler).

When an interrupt occurs the currently executed instruction is always completed before control is passed to the appropriate interrupt routine. All interrupt routines pass control to the dispatcher when their actions are completed. An interrupt is also generated in the event of system stack overflow (the stack is pointed to by A15). This will cause a system halt because processing can not be allowed to continue as the consequences are unpredictable.

As can be seen from figure 12.1, the interrupt handlers in boxes 1, 2, & 4 are self contained; they do not call any subsidiary modules. The handlers in boxes 1 & 2 terminate by halting the machine. The handler in box 4 terminates by branching to the dispatcher (box 38).

The interrupt handler for the real time clock (box 3) activates a special clock task (box 43) every 100 ms. The hardware interrupt occurs every 20ms. The handler terminates by branching to the dispatcher (box 38). The interrupts generated during I/O operations are serviced by the appropriate device driver (eg. box 25).
At completion of the I/O, the driver calls TENDIO (in module TOSSIO) before branching to the dispatcher.

The interrupt handler for LKM requests processes only request type 0. The interrupt handler saves registers A1 to A14 in the task table (TTAB) of the current task and, if necessary, branches to one of several LKM processors in order to process the remaining types of LKM requests.

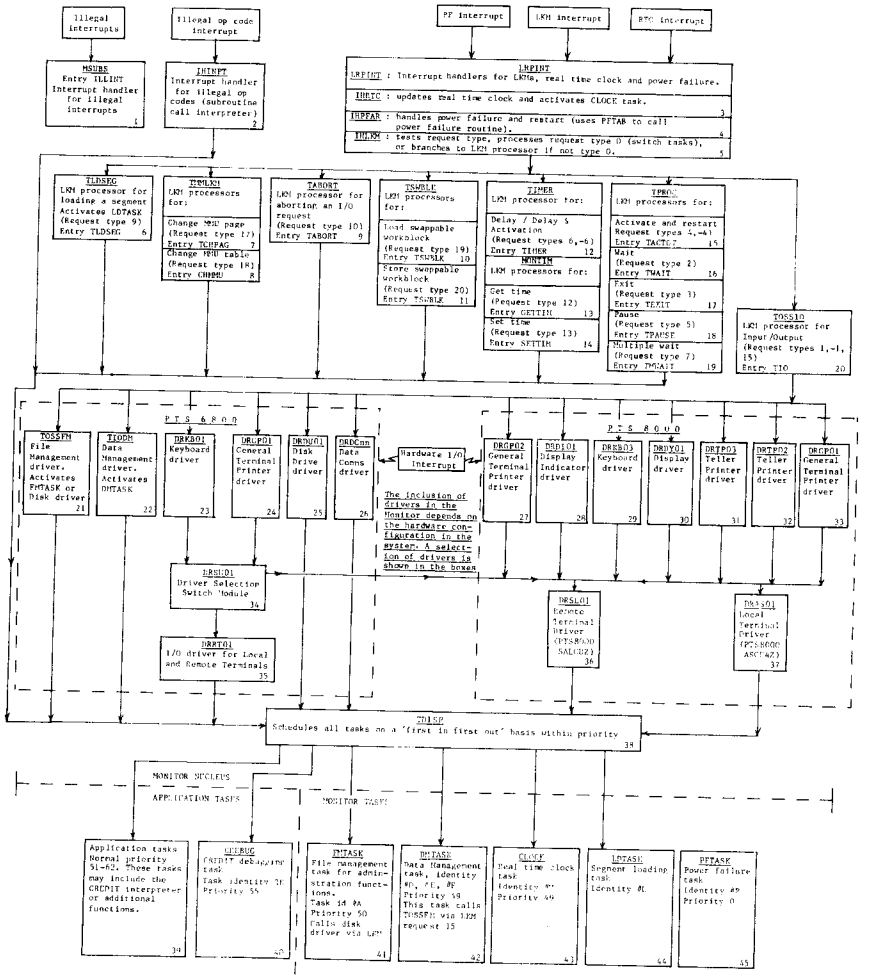


Figure 12.1. TOSS Component Overview.

13.1 GENERAL

I/O drivers perform device control functions and (optionally) process data communications and data management I/O requests. The type of function to be performed is specified in register A7 by the requesting task. Examples are as follows:-

/00	Test Status	/05	Basic Write
/01	Basic Read	/06	Standard Write
/02	Standard Read	/0A	Random Read
/03	Numeric Read	/0B	Random Write

Further parameters are specified in an 'Event Control Block'.

The address of the event control block is placed in register A8 by the current task.

A separate driver is available for each type of device (for example boxes 23 to 33 in figure 12.1) and some devices may have more than one driver (for example a keyboard). The required drivers must be built into the monitor during system generation. Separate drivers are also available for performing device control (e.g. 35) at either local or local and remote work positions comprising one or more of the following devices:-

* Keyboard
* Teller Terminal Printer
* General Printer
* Numeric Display
* Indicator Display/Keyboard Lamps
* Video/Plasma Display

These devices must be connected to the Terminal Computer via a Channel Unit for Local Terminals (CHLT) or a Channel Unit for Remote Terminals (CHRT). Devices which are used remotely (i.e. via modems) must be connected to the CHRT.

For 6800 systems one of two drivers may be used to control devices attached to the CHLT or CHRT.
The situation is somewhat different for PTS 8000; see section 13.4, which also describes the driver switch module for DRAS01 and DRSL01 in PTS 6800 systems.

Driver DRLT01 is used to control devices attached to a CHLT. Driver DRRT01 is used to control devices attached either to a CHLT or a CHRT. That is, driver DRLT01 controls locally connected devices only, and driver DRRT01 controls both locally and remotely connected devices. Only one of these drivers is included in the Monitor during system generation.

Drivers DRLT01 and DRRT01 are normally entered only from the individual device drivers for the above devices (for example boxes 23 and 24). The only exception to this rule is the 'test remote line' function. In this case the driver DRRT01 is entered directly from the LKM processor (box 20).

Drivers DRLT01 and DRRT01 terminate by branching to the Dispatcher (box 38).

The drivers for the remaining devices not listed above (for example box 25) are self contained. That is, they do not enter any additional driver. These drivers all terminate by branching to the Dispatcher.

Separate drivers are also available for each type of communication line discipline. These drivers terminate by branching to the Dispatcher.

I/O drivers for devices and data communications are only included in the Monitor if they are specifically requested during system generation.

A separate driver, TIODM, is available for data management (box 22). This 'driver' activates a special data management task (box 42) to process data management requests. This task issues disk I/O requests. The data management driver is only included in the Monitor if data management is requested during system generation. A similar situation exists with file management (boxes 21 and 41).

13.2 DRIVER STRUCTURE

An I/O driver in TOSS consists in general of the following parts:-

* Activation of request
* Interrupt handler
* Procedure handler
* Termination of request
* Abortion of request
* Power on Initialization

Some of these are optional but the activation and termination parts, for instance, are always present.

In most cases the parts can all be regarded as interrupt handlers, internal (LKM) or external.

Only if the driver has to perform a lot of processing may a task be activated from the driver, for example SNA handling in the DC driver for IBM 3600 emulation.

The connection from the application program to the device (data-set) is done via a file code which is used in scanning the current task table (TTAB) for the corresponding device work table (DWT).

In some cases one DWT may serve several devices, the file code being then connected to a DWT plus an index (for example a magnetic tape controller for eight tape stations).

Only one request per DWT is handled by the driver, and the driver may simultaneously handle several requests for different DWT's. The DWT is also used as a queue element.

The driver always works with the DWT and the file code is never seen. The first part of the DWT is standard, the rest is driver dependent.

13.2.1 Activation of Request

When the application executes a monitor request for I/O (LKM DATA 1), control is first given to a common I/O request handler (TIO) which performs some standard functions such as looking up the DWT, checking if the device is free, queueing, etc.

The activation entry in the driver is found via the DWT (DWTADR). It is then called with all information necessary to perform the request passed in the DWT (DWTOR).

The activation part in the driver normally ends by starting up some hardware action, and a branch to the dispatcher (TDISP) is then made.

13.2.2 Interrupt Handler

The interrupt handler is called when a control unit is ready with a data or status transfer, and has raised an interrupt. The CPU saves the current program counter (P) and status word on the A15 stack and gives control to the interrupt handler, which is found via the appropriate entry in the interrupt vector table. Registers A1 thru A8 are saved by the driver.

After executing the proper I/O instructions, either a return to the interrupted program is made, or a branch to the procedure handler, or a call to the termination part.

13.2.3 Procedure Handler

The procedure handler can be viewed as a higher level interrupt handler. For example, in a data communications driver the interrupt handler assembles a complete message and passes it to the procedure handler.

After performing the necessary actions the procedure handler either returns to the interrupted program, or calls the termination part.

13.2.4 Termination of Request

When the request has been executed (for example a message received or transmitted), a monitor routine (TENDIO) is called. This routine checks if the device queue is empty. If not, than the first task in the device queue is put into the dispatcher queue. After this, the task which completed the I/O (with no wait) is put into the dispatcher queue.

The driver then exits to the dispatcher (TDISP) to return to the interrupted task, or to start the task that had its I/O terminated, depending on which has the highest priority.

13.2.5 Abortion of Request

If the application wants to abort a previously issued I/O request, a special monitor request is available. This may happen when, for example, three I/O requests with no wait are outstanding and the others may be aborted on completion of one.

The monitor request handler calls the driver at an abort entry point found via the DWT, and then calls TENDIO to terminate the request, indicating in the return code that it was aborted.

Only a few TOSS drivers support the abort request (for example keyboard, badge card reader, and data communications).

13.2.6 Power-On Initialization

After power-on or system load, each driver is called at a special entry point where recovery and initialization functions are performed. These entry points are contained in a table, PFTAB.
I/O operations which were active immediately before the break are repeated, except for keyboard operations (see SYSGEN).

13.2.7 Support Routines

One support function is available to drivers in the monitor. This is Start a Timer (SETTIM).

It is possible for the driver to start a timer which, after a specified time, will activate a specified entry point in the driver.
This code in the driver is treated as being part of a monitor task and the time is given in multiples of 100 ms.

13.2.8 Environment

* Driver Entry Points:-

  - ACTIVATE
  - ABORT (optional)
  - POWER-ON
  - INTERRUPT HANDLERS

* Driver External Points:-

  - DISPATCHER (TDISP)
  - ENDING OF I/O REQUEST (TENDIO)
  - TIMER FUNCTION (SETIMP)
  - QUEUE MONITOR TASK (QMJOB)
  - DEVICE WORK TABLES (DWT)

13.2.9 Register Usage Conventions

The dispatcher (TDISP) must be called with eight registers (A1-A8) from the current active task on the A15 stack. (The idle loop does not use any registers).

* The activation part may use any register (A1-A14).

* The other parts may use any registers that are saved at the interrupt and restored at return. (A1-A8 are automatically restored when calling TDISP).

13.2.10 I/O Request Flow

When a task performs an I/O operation, control is passed to the TOSS monitor (see figure 13.1).

The TOSS monitor starts the I/O operation (e.g. with wait) on the requested device. When the I/O is started, a check is performed to see if another task can be dispatched (within priority level).

After completion of the I/O, the task is scheduled for dispatching on a First-In-First-Out basis within priority. When the task is dispatched again, it will continue after the I/O instruction.

The detailed actions performed and the TOSS modules involved are as shown in figure 13.1. The numbers in the figure are explained below:-

* Task A0's code is interpreted (e.g. CREDIT code) and an I/O instruction (e.g. KI) is executed (1).

* The interpreter passes control to the TOSS monitor with the LKM DATA 1 instruction. Control is given to the TOSS module IHLKM. Because of the I/O operation, control will be passed to module TIO (2), which searches for the corresponding device worktable (DWT) and the activation address of the driver.

* Control is passed to the activation part of the driver (3). The device driver starts character transfer via the DRRT01 or DRLT01 driver and passes control to the dispatcher module TDISP (4).

* The next task within priority is dispatched, e.g. Task A1 (5). While Task A1's code is being interpreted, and A1 has not yet requested I/O, an interrupt is raised due to reception of a character via DRRT01 or DRLT01 (6).

* Normal interrupt actions are performed and control is passed to the interrupt handler in the device driver (7).

* After the character is stored in a buffer, the dispatcher module (TDISP) gets control (8) and the interpreter continues interpreting the code for Task A1.

* On completion of the I/O operation, the terminating part of the driver is activated, which calls the module TENDIO (9). This module checks if other tasks have outstanding requests for this device (device queue). If so, the queued task will be scheduled and then the task which completed its I/O will be scheduled. Control is then passed to the dispatcher module (TDISP).

Figure 13.2 shows the dispatching sequence in more detail.

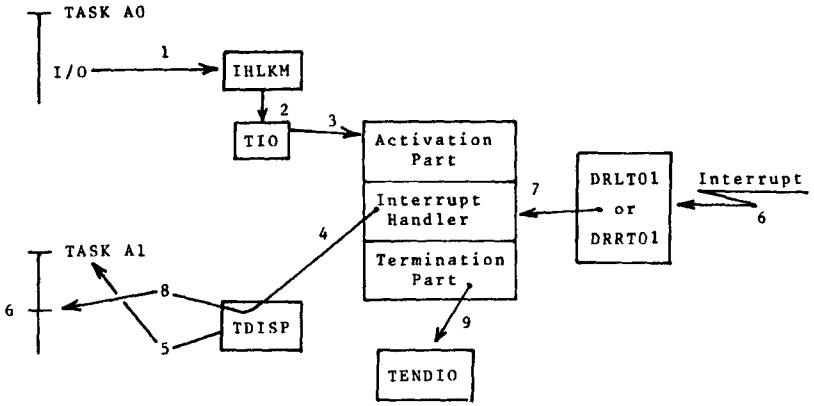


Figure 13.1. TOSS I/O Request Flow.

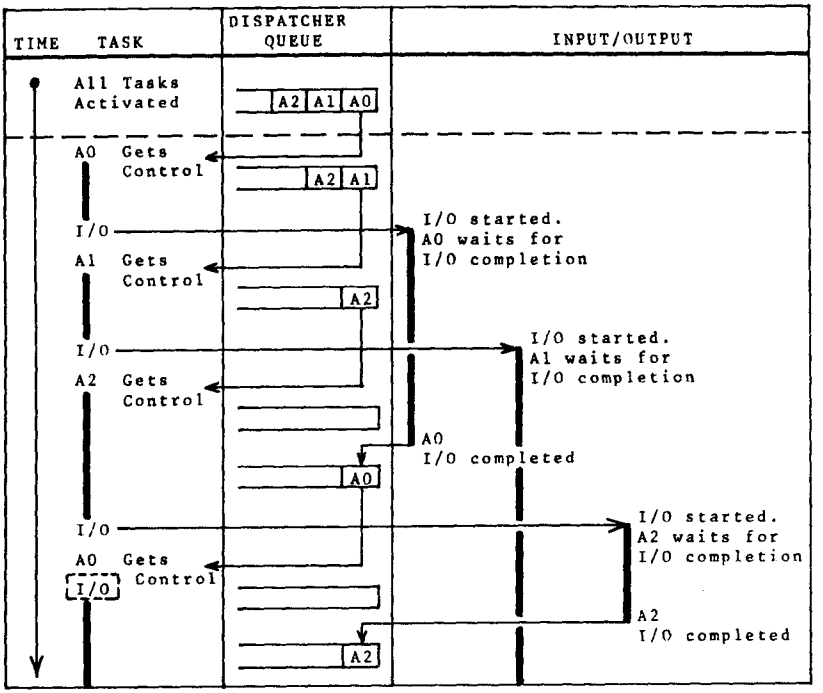


Figure 13.2. TOSS I/O and Dispatching.

13.3 DRIVER FOR LOCAL AND REMOTE TERMINALS (DRRTO1)

13.3.1 General

The system software handling workstations either locally or remotely connected to a PTS computer is divided into two parts, the channel unit driver and the device drivers. See figure 13.3.

When only local workstations are configured, the driver for local workstations (DRLTO1) should be part of the monitor. This driver has no interface to the application software, only to the device drivers.

When both local and remote workstations are configured, the channel unit driver for local and remote workstations (DRRTO1) must be included in the monitor. In this case DRLTO1 must be excluded. Only one of the drivers DRLTO1 and DRRTO1 can be present in the monitor. DRRTO1 has only one point of connection to the application software, the test remote line function. DRRTO1 has connections to:-

* Application software (test remote line)
* Device drivers
* CHRT/CHLT hardware

The main purpose of DRRTO1 is to supervise the transmission of characters between device drivers and Channel Units. A message reporting the result of any transmission is sent to the device drivers by DRRTO1.

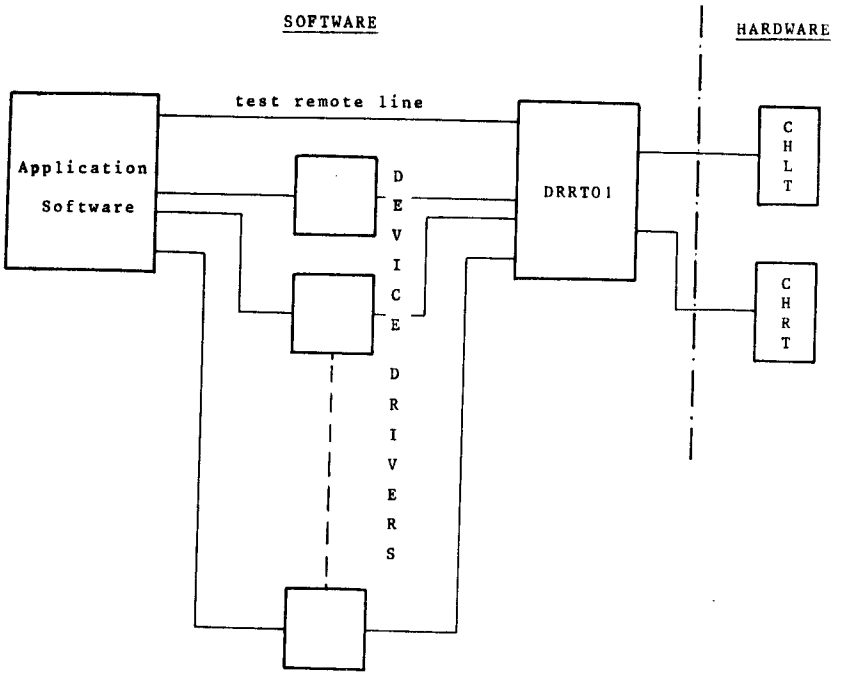


Figure 13.3. Driver Interfaces.

Figure 13.4 shows the system software actions carried out when an LKM request occurs. At activation the device driver orders DRRTO1 to send or receive a character. DRRTO1 executes the command and, when the interrupt comes (output control character), checks it and gives control back to the device driver with a message concerning the result of the input or output transfer. The device driver then orders input or output until all requested characters are transferred, or an error occurs.

The device work table (DWT) serves as a communication link between DRRTO1 and device drivers.

See also figure 13.6 which shows control block relationships.

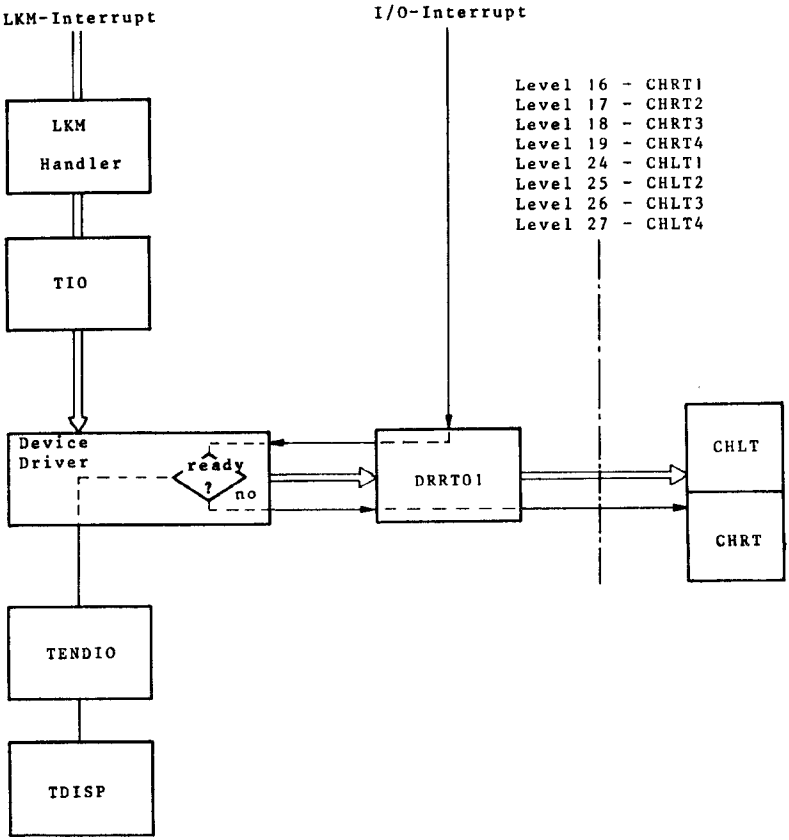


Figure 13.4. LKM Request Sequence.

The following abbreviations are used in the description:-

		Hex Codes (Right Byte)
DRRTO1	DRiver for local and Remote Terminals	
ACK	ACKnowledgement of transmitted character	07
NAK	Negative AcKnowledgement of transmitted char	05
OER	Hardware time-out on output transmission	00
DRI	Data Request Immediate (printers) ..... Bit A = 1.	00,04,08,0C
DRD	Data Request Delayed (printers) ..... Bit A = 0.	00,04,08,0C
SER	Power-on character from selector unit	03
STD	Status input (e.g. printer voucher status) .. Bit A = 0.	00,04,08,0C
OBC	Output Block Control function	08,0A
ABC	Acknowledgement of a Block transmission incl. information about LRC and VRC control in the selector unit ..... Note: For OBC and ABC the left byte is used for LRC.	0C,0E
DIN	Data from INput device.	
SYN	Output for SYNchronisation of modem	55

Figure 13.5 is a copy of figure 2.9, repeated here for reference.

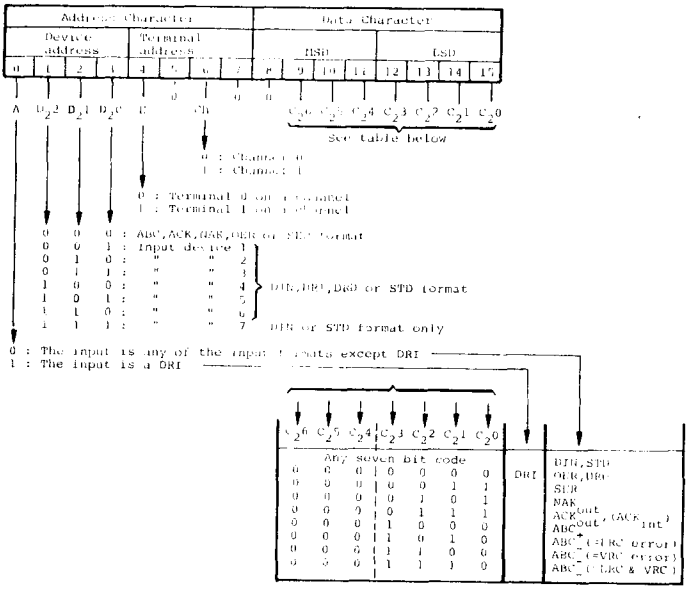


Figure 13.5. Layout of Input Messages from CHRT to CPU (copy of 2.9).

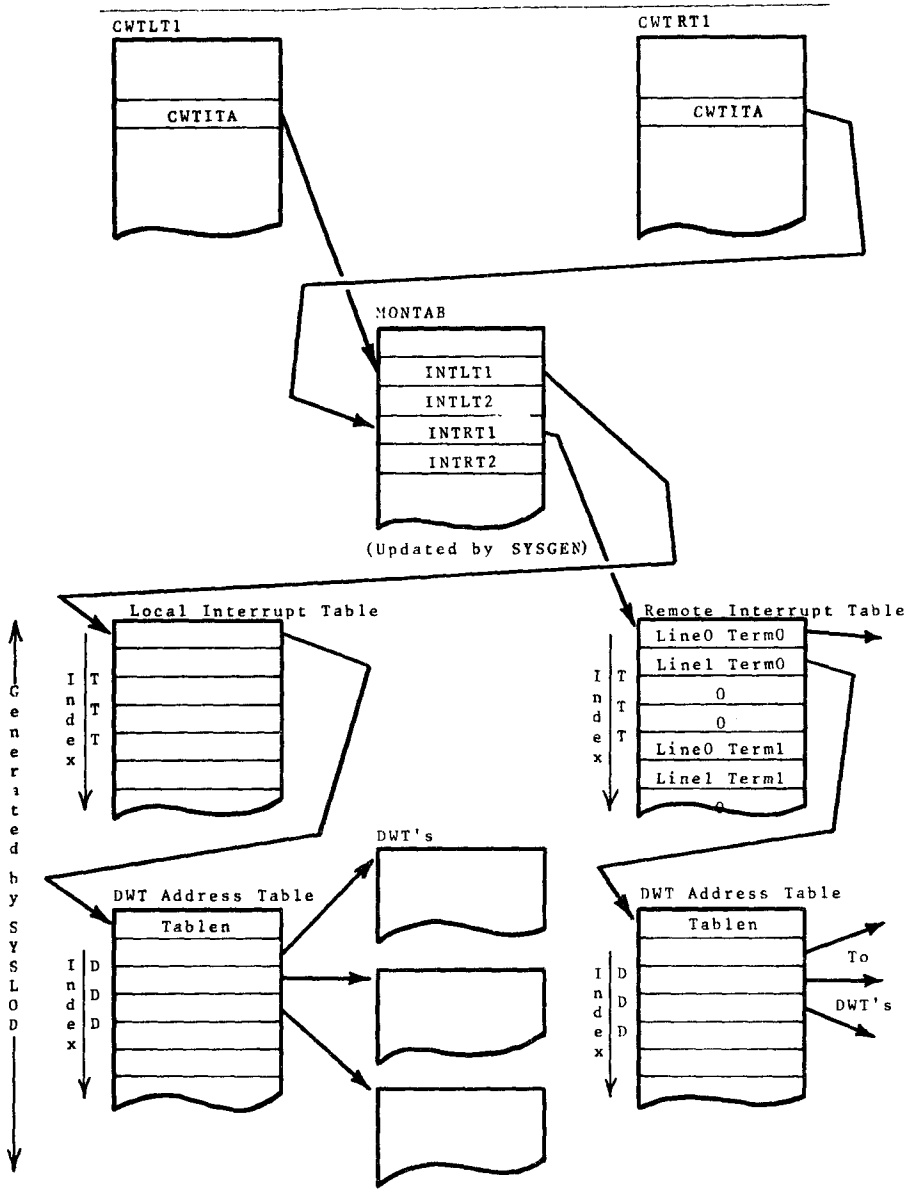


Figure 13.6. Channel Work Table and Associated Control Blocks.

13.3.2 Modules in DRRTO1 used by Device Drivers

Below are described five routine calls used by device drivers. The routines are parts of DRRTO1:-

1. CF A5,OUTPUT

All outputs to devices connected to a CHLT or CHRT are administered by DRRTO1. This call is the normal way to send a character from device driver to device.

Register contents before calling:-

A2 = Character to send (bits 8-15).
A3 = Work Register.
A4 = Work Register.
A5 = DWT Stack pointer.
A6 = DWT address.
A7 = Order.
A8 = ECB address.

Procedure:-

* Contents of registers A3 thru A5 are saved in the DWT.
* 'Interrupt Allowed' is indicated in the DWT (bit 7 in DWTST).
* When the channel unit is free, a character is sent to the device via the channel unit (word zero in the CWT is tested).
* If the channel unit is busy the output request is queued for transmission later.
* Control is given to the dispatcher while the channel unit is waiting for a return character.
* Possible entry points in the device driver after transmission are the interrupt handler or a recovery routine.

2. CF A15,OUTLIN

OUTLIN is a subsection of the routine OUTPUT described above. The difference between the calls is that OUTLIN does not save A3-A5 in DWT before transmission, and 'interrupts allowed' is not indicated. Simple drivers which do not use many registers use this call.

Register contents before calling:-

A2 = Character (bits 8-15).	A7 = Order.
A6 = DWT address.	A8 = ECB address.

Remotely connected devices select transmission in block mode if only one terminal is connected to a TFU. DRRTO1 performs this test. For devices with the data request function, e.g. printers, this implies that the block is sent character-by-character with LRC control of the whole block. Devices without the data request function use the CHRT block transmission feature with which LRC control of the block is performed.

The beginnings and ends of messages under LRC control are marked by the device driver sending the characters STX (/02) and ETB (/17). These characters must be complemented for DRRTO1 by the setting of the leftmost bit in the byte. There is no need for separate routines in the device driver for locally and remotely connected devices, as these two special characters are ignored in DRRTO1 for local devices.

3. CF A5,STREG

Input is possible only in single character transmission mode.
A circular input buffer is used to store characters from an input device. At the time of a request, this buffer is collected first, and then the device driver has to prepare itself to get the next input character from the channel unit driver. This preparation is performed using the above call.

Register contents before calling:-

A3 = Work Register.	A6 = DWT address.
A4 = Work Register.	A7 = Order.
A5 = DWT Stack base.	A8 = ECB address.

Procedure:-

* Registers A3 to A5 are saved in the DWT.
* 'Interrupt allowed' is indicated (bit 7 in DWTST).
* Control is given to the dispatcher while waiting for a channel unit interrupt.
* Possible entry points in device drivers after input transmission are the interrupt handler or a recovery routine.

4. CF A15,GETCHR

This subroutine is frequently used by drivers as a general printer, video display, and numeric display driver. It is used to fetch the next character from the ECB buffer and, when the end of the buffer is reached, to store the effective length in the ECB. Register A3 is used as an index and contains a displacement from the start of the buffer to the character to be fetched.

Register contents before calling:-

A3 = Buffer Index.	A8 = ECB address.

Register contents after return:-

A1 = Positive if not last character. Else zero or negative.	A3 = Updated Buffer Index.
A2 = Character (bits 8-15).	A8 = ECB address.

5. ABL LDREG

This call is used by almost all device drivers, especially in the interrupt handling part. After some tests the interrupt handler gives control back to the running part of the device driver by executing return from the call CF A5,OUTPUT or CF A5,STREG.

Procedure:-

* A test for 'interrupt allowed' is carried out, and if so the contents of A3, A4 and A5 saved in the DWT are restored. Bit 7 in the status word, DWTST, is reset, indicating interrupts not allowed. The Order is stored in A7 and the ECB address is stored in A8. A return is executed from the subroutine call CF A5,OUTPUT or CF A5,STREG.

13.3.3 Activation of Device Drivers from DRRTO1

Every output from DRRTO1 will cause an interrupt from the channel unit. Such interrupts, together with data inputs, are controlled by DRRTO1, and the device driver is activated. Two entry points are possible, depending on the result of the transmission:-

* Interrupt Handler.
* Recovery Routine.

Examples of situations where the interrupt handler is activated:-

* An output has been executed and found to be correctly transmitted.

* An output has been executed and found to be incorrectly transmitted; a special return code is set.

* Data input, for example from a keyboard.

Register contents before the interrupt handler in the device driver is entered:-

A1 = Return Code (0 = operable, 1 = not operable).
A2 = Input word (for example, containing voucher and journal status or data character).
A6 = DWT address.
A7 = Order.
A8 = ECB-address.

Examples of situations where the recovery routine is activated:-

* Selector unit powered after a power break.

* Hardware timeout for a transmitted character.

* Bad LRC on transmitted block.

* Seven NAK's from the selector unit for the same character.

* Software timeout on data request supervision for printers.

* Software timeout on block check character supervision for a selector unit.

* Voucher status change before the whole message has been transmitted to a teller printer.

Register contents before the recovery routine in the device driver is entered:-

A4 = 0 if selector unit power-on,
<u>or</u> hardware timeout for transmitted character,
<u>or</u> seven NAK's from a selector unit.
A4 ≠ 0 if LRC error (or any other type of error).
A5 = DWT stack base.
A6 = DWT address.

13.3.4 Interface Program for CHLT and CHRT

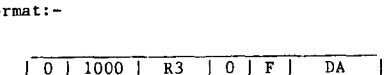
Channel units are controlled by instructions from the following set:-

* CIO Start ..... Start I/O.
* OTR ..... Output.
* INR ..... Input.
* CIO Halt ..... Stop I/O.
* SST ..... Send Status.
* TST ..... Test Status.

DRRTO1 uses only CIO Start, OTR, and INR in order to control the channel units for local and remote terminals.

1. CIO START.

Format:-



R3 = Operand Register
(not used)

DA = Device Address
/03 for CHLT1
/13 for CHLT2
/23 for CHLT3
/33 for CHLT4
/01 for CHRT1
/11 for CHRT2
/21 for CHRT3
/31 for CHRT4

Function and Condition Register contents:-

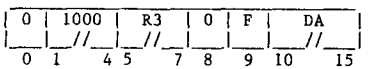
CIO Start is used to prepare the channel unit for transmission of characters; used at system start and after mains power failure.

Address not recognized:- CR = 3

Address recognized:-
command rejected - CR = 1
command accepted - CR = 0

2. OTR - Output one character.

Format:-

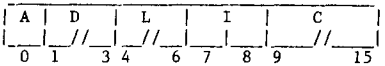


DA = Device Address
R3 = Operand Register
F for remote channel:-
0 = channel 0
1 = channel 1
F not used for
local channel.

Function and Condition Register contents:-

The OTR instruction is used to transfer data from a CPU register to a channel unit. It is accepted only if the channel unit is in exchange state.

Format of R3:-



A = Procedure control bit (remote only).
1 = Block transmission mode.
0 = Single character transmission mode.

D = Device Address.
0 = OBC message, clears LRC logic in SUMR (remote only).
1 thru 6 = Output devices.
7 = SYN format (remote).
or Output device (local).

L = Line number.
Local ..... Line numbers 0 thru 7.
Remote .... /0 = Terminal 0, Channel 0.
/2 = Terminal 0, Channel 1.
/8 = Terminal 1, Channel 0.
/A = Terminal 1, Channel 1.

I = Irrelevant.

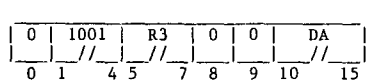
C = Character (any seven-bit code).

Address not recognized:- CR = 3

Address recognized:-
command rejected - CR = 1
command accepted - CR = 0

3. INR - Input one character.

Format:-



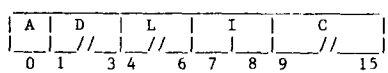
DA = CU Device Address.

R3 = Operand Register.

Function and Condition Register contents:-

The INR instruction is used to transfer data from a channel unit to a CPU register.

Format of R3:-



A = Procedure control bit (remote only).
1 = Data Request Immediate (DRI) in bits 9 thru 15.
0 = Any other input format in bits 9 to 15. This depends also on the value in D (device address).

D = Device Address.
0 ..... Following formats possible in bits 9 thru 15;
ABC - Block control character.
SER - Power-on.
OER - Hardware timeout control character.
ACK - Output acknowledged.
NAK - Output not acknowledged.

1 thru 6 ... Following formats possible in bits 9 thru 15;
DIN - Data input.
DRD - Data request delayed.
STD - Status input.
DRI - Data request immediate.

7 ..... SYN input (remote).
or Device input (local).

L = Line number.
Local ..... Line numbers 0 thru 7.
Remote .... /0 = terminal 0, channel 0.
/2 = terminal 0, channel 1.
/8 = terminal 1, channel 0.
/A = terminal 1, channel 1.

I = Irrelevant.

C = Character code and meaning; depends on A and D fields.

/0 =	OER	Hardware timeout.
/2 =	DRI	Data request immediate (remote).
/3 =	SER	Power on.
/5 =	NAK	Output not acknowledged.
/7 =	ACK	Output acknowledged.
/8 =	ABC+	Acknowledgement of block transmission, no error.
/A =	ABC-	Acknowledgement of block transmission, bad LRC.
/C =	ABC-	Acknowledgement of block transmission, bad VRC.
/E =	ABC-	Ack. of block transmission, bad LRC and bad VRC.

Any seven-bit code.
DIN = Data input.
STD = Status input (remote).
DRD = Data request delayed.

Address not recognized:- CR = 3

Address recognized:-
command rejected - CR = 1
command accepted - CR = 0

Note One important rule is always adhered to in program/channel unit communication:-

* Local: Every output results in a reply from the channel unit to the program after transmission. This includes the case where a power-on character appears from a selector unit just after an output. The three replies possible are ACK, NAK, and OER.

* Remote: There are four possible replies in this case. They are ACK, NAK, OER, and DRI.

13.3.5 Functions of DRRT01

The most fundamental functions performed by the channel unit driver are listed below.

* Activation of channel units.
* Output and input administration.
* LRC control for output messages.
* Transmission of synchronisation characters.
* Time supervision of data requests from printers and block control characters after block messages.
* Remote line Test.
* Hardware status control by use of error accumulators.
* Logging of inputs and outputs for checking of both hardware and software actions.

The channel unit driver can handle up to four local channel units and four remote channel units at the same time. These channel units are placed on eight different interrupt levels.

For programming purposes, a Channel Work Table (CWT) is needed for each channel in the system. The CWT holds information concerning the status of transmission on the channel.

Channel Work Table for Local Terminals, CWTLTy (y = 1 thru 4):-

CWTLDW	Last output DWT
CWTLOW	Last output word
CWTITA	Address of interrupt table
CWTINR	INR instruction
CWTOTR	OTR instruction
CWTCIS	CIO Start
CWTRTC	Retransmission counter
CWTEQ	Queue first terminal on channel
CWTADD	NAK accumulator
.....	Retransmission fault accumulator
Reserved	
ACKTIM	Timeout accumulator (3 seconds)

Channel Work Table for Remote Terminals, CWTRTy (y = 1 thru 4):-

CWTLDW	Last output DWT
CWTLOW	Last output word
CWTITA	Address of interrupt table
CWTINR	INR instruction
CWTOTR	OTR instruction
CWTCIS	CIO Start
CWTRTC	Retransmission counter
CWTEQ	Queue first terminal on channel
	Queue second terminal on channel
CWTADD	NAK accumulator
.....	Block error accumulator
Reserved	
ACKTIM	Timeout accumulator (3 seconds)
CWTTP	Timer pointer
CWTSYN	SYN character
LRCDWT	DWT of running LRC process, terminal 1
	DWT of running LRC process, terminal 2
LRCACK	LRC accumulator terminal 1
	LRC accumulator terminal 2
CWTRST	Channel Status Word (loop test)
CWTBLK	Block sending indicator

CWT field descriptions:-

CWTLDW Last output DWT.
When an output is done to the transmit buffer in a CHLT or CHRT the channel unit is not then able to receive any more characters. The software indicates this by storing the DWT address in CWTLDW. The DWT concerned is for the device occupying the channel unit for output transmission. When the channel unit receives a reply for the transmitted character, its state switches from Execute to Exchange, and further outputs can then be done, indicated by clearing CWTLDW.

CWTLOW Last output word.
The last output word used when a selector unit replies NAK for output transmission. The last output word may be retransmitted up to seven times.

CWTITA Address of interrupt table pointer.
Channel units have one interrupt table each. This word points to a word in MONTAB (INTLTx for local, INTRTx for remote) which points to the interrupt table. MONTAB is created at SYSGEN. The interrupt table reflects the physical configuration of the devices connected to the channel. A word in this table points to the DWT address table, which consists of pointers to the DWT's of devices present at the workstation. The device causing the interrupt can be determined from the device address received. For terminal recovery, the table is used to find which devices are connected to the terminal. See figure 13.6.

CWTINR, CWTOTR, CWTCIS — These words contain the INR, OTR, and CIO instructions used by the driver.

CWTRTC Retransmission counter.
If the channel unit has problems when receiving a character, a NAK is sent, the character is retransmitted, and this counter is incremented by one.

CWTEQ Terminal queue.
Every local channel and every remote terminal has its own output queue. CWTEQ is a pointer to the first device in the queue. The second device pointer is found in the DWT of the first device in the queue, and so on (DWTTQ). The output queue is built when an output request is made and the word CWTLDW is not equal to zero. See figure 13.7.

CWTADD NAK accumulator.
This accumulator is incremented by one whenever a NAK is received from the SUMR.

..... Retransmission fault accumulator (local).
When a character is received from a keyboard and the channel cannot accept it, up to four retries are made. This word contains the retry count.

..... Block error accumulator (remote).
In block mode, whenever a negative ABC is received, one is added into this accumulator.

ACKTIM Timeout accumulator (3 seconds).
When, for example, a data request (DRI) is missing, i.e. not received within 3 seconds, this accumulator is incremented by one.

CWTTP Timer pointer (500 ms).
Contains a pointer to a timeout block containing a value of 500ms. Used for SYN timing.

CWTSYN SYN character (remote).
Code X'55' stored in this word is sent every 500 ms when there is no other transmission on the channel. The terminals connected to the line are addressed alternately. This is needed in order to obtain the line status, for example when loop-connecting the line.

LRCDWT Every remote terminal has a pointer to the DWT for sending a message under LRC control. Other devices on the terminal needing to send such messages must wait until this indicator is cleared, i.e. the current message is sent, and the reply from the hardware about the result of the block transmission has come back.

LRCACK This accumulator is used for the control of even parity failures in the transmission of blocks. It is cleared at the beginning of every block to be sent, and when the last character in the block is sent, the value of the accumulator is sent down the line so that the selector unit can compare it with a hardware accumulator. Each remote terminal has its own software accumulator.

CWTRST Channel Status Word.
Bits 9, 14, and 15 when set have the following meanings:-
Bit 9 - CHRT not active or hardware failure.
Bit 14 - No reply received on output from channel (no ACK).
Bit 15 - No SYN received. TFU switch is probably not correctly set for looptest.
If bits 14 and 15 are both set, the line is probably broken.

CWTBLK Block sending indicator.
Bits 1 and 15 have the following meanings when set:-
Bit 1 - Block sending is active.
Bit 15 - Block sending is allowed (one terminal on TFU).

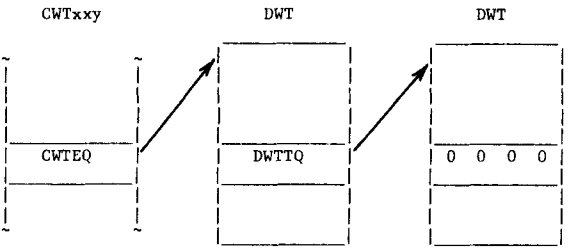


Figure 13.7 Terminal Queueing.

13.3.6 Starting of Channels

At system start local and remote channels are made active by CIO Start commands. The channels remain active all the time that the program is running. Mains power failure sets channels inactive, but the program activates them immediately on power-on.

Activation of channel units causes a power-on interrupt from each active selector unit in the system, which in turn causes device recovery for active terminals. For remote terminals, a timer is started to send SYN characters every 500 ms. If a channel is busy at power-on time, for instance if a character was being sent from the program just before the power break, a dummy is sent to the device to ensure that the device driver gets control of the situation.

13.3.7 Output Administration

All outputs are handled by the driver DRRT01. Every device driver orders output by preparing the device work table and calling a subroutine in the channel unit driver. If the channel unit is free, i.e. the result of the last character transmission has reached the program, the character is sent to the hardware.

However, if the channel unit is busy, the character must be queued. In the local situation, each channel has its own queue, and the characters are queued by the First-In-First-Out (FIFO) method. Every time the channel unit becomes free, i.e. a transmission control character is received from the channel unit as a reply to an output character, the next character in the queue is sent.

Each remote terminal has its own output queue, and each channel may have two terminals. When a channel becomes free, the queue belonging to the terminal not involved in the last transfer is checked. If the queue is not empty a character is transmitted, otherwise the other queue is checked.

For local printers every output character causes an ACK, NAK, or OER. If it is ACK, then Data Request Delayed (DRD) is sent when the selector unit is ready for the next character.

There is a 40-character receipt buffer in the selector unit for use by remote terminals. If this buffer is not full, Data Request Immediate (DRI) is sent by the selector unit as the reply to printer output. When the buffer is full, ACK is sent as reply, and when it is no longer full, DRD is sent to the channel unit. The program can check when the whole message has been printed by sending /03 and waiting for DRD after ACK, when the selector unit buffer is empty.

A special character, /01, can be sent to remotely connected printers when the program requires the printer to stop sending characters from the selector unit buffer.

13.3.8 Input Control

The following types of input interrupts are possible:-

* ACK - Output acknowledgement.
* NAK - Output not acknowledged.
* DRI - Data request immediate.
* OER - Hardware timeout on output character.
* DRD - Data request delayed.
* DIN - Data input.
* STD - Status input.
* ABC - Block control character.

ACK, NAK, or OER are possible replies from a local channel unit after an output. Remote units additionally use DRI.

DRRT01 operations caused by various interrupts are outlined below:-

* ACK The next character from the output queue is sent and a return is executed to the device driver.
  Note: If the ACK is from a printer, control is not passed to the device driver until DRD has been received.

* NAK The character is retransmitted up to seven times, then the recovery routine in the device driver gets control.

* DRI The next character from the output queue is sent and control is returned to the printer driver. (Remotely connected printers only).

* OER The first timeout causes device recovery. If two timeouts occur in succession, control is passed to the interrupt handler in the device driver with a 'not operable' return code.

* DRD The printer driver's interrupt handler is activated. For locally connected printers this data request character gives information about voucher and paper status. For a remotely connected printer the data request character is supplemented with voucher and paper status information in DRRT01.

* DIN The data input character is transmitted to the interrupt handler in the device driver.

* STD A status message is sent from printers when a change in voucher or paper status occurs. The new status is indicated in the DWT. Sometimes a change in voucher status requires termination of the current operation, for example if a voucher is taken away while printing. In this case the recovery routine in the printer driver is activated, and it sends a code /01 to clear the print buffer in the selector unit, stopping the printer immediately. (Remotely connected printers only).

* ABC There are four types of block control characters:-

  ABC+ ... Block transmission correct.
  ABC- ... Bad LRC.
  ABC- ... Bad VRC.
  ABC- ... Bad LRC and VRC.

  A check is made that block control characters are currently permitted. Then, if ABC+, control is given to the interrupt handler in the device driver. In the case of bad LRC or VRC, the recovery routine in the driver is activated.

13.3.9 Longitudinal Redundancy Check Control (LRC)

This feature is only implemented with remotely connected selector units. Each remote selector unit has an LRC accumulator for LRC control of messages sent to its devices.

Beginnings and ends of messages are determined by sending a special character, OBC, to device address zero. The first OBC clears the selector unit accumulator and the last one contains an accumulator set up by software to compare with the hardware accumulator.

Since only one LRC accumulator per selector unit is available, only one of the unit's devices at a time can send a message under LRC control. If one device is sending an LRC-controlled message and another device on the same selector unit also needs to send one, the second device must wait in a queue until the first device is ready with its message.

However, devices not needing LRC control of messages can mix their characters with characters from the device sending an LRC-controlled message. Two devices belonging to different selector units connected to the same TFU can send messages under LRC control at the same time.

The LRC feature may be used with both modes of transmission. Block mode is used particularly for video displays and character-by-character transmission under LRC control is used with printers.

Bad LRC in a printer message immediately causes activation of the recovery routine in the device driver, and its first action is to send a character /01 to clear the print buffer in the selector unit. Although the whole message has been sent to the selector unit when bad LRC is detected, the printer may not have printed all of it when it is stopped.

Block transmission is selected by setting a bit in the output register. The channel unit generates an internal ACK immediately for each character, and does not wait for a reply from the selector unit. This means that the device driver gets no information about the success of the transmission while it is in progress. It therefore has to send a character in single transmission mode after the block, in order to get TFU, selector unit, device, and line status.

13.3.10 SYN Transmission

A synchronisation (or SYN) character, code /55, is transmitted by DRRT01 every 500 ms when no other input or output messages are in progress. Device address 7 on the selector unit is reserved for SYN transmission. The two selector units connected to a line are addressed alternately, so that ACK is received at least every 2nd time each selector unit is busy. This is important when the line is loop connected.

The SYN timer is started when the channel unit is activated at system start or after a power break. Each input or output restarts the timer, and if there is no activity on the channel for 500 ms, a SYN character is sent to the selector unit.

13.3.11 Time Supervision of Data Request and Block Control Characters

For output to devices with data request (printers), and devices sending messages in block transmission mode, there is a timeout facility to supervise data request characters and block check characters.

Printers:-

The first output in a printer message to a selector unit (OBC or some other character) starts the timer. For every ACK or DRI during output of the message, the timer is restarted. DRD after ETX, which always ends a printer message, resets the timer.

The last block output character (OBC) is sent before ETX to test for even parity. If timeout occurs before ETX is sent, it means that either a data request or a block check character is missing. Either case causes message recovery.

Time before timeout is set to 3 seconds. If a timeout occurs when a teller printer is waiting for grasp or release, ETX characters are sent every third second until the grasp or release is executed by the operator.

Block mode devices:-

ACK of the first block output character (OBC) starts the timer, and the character after the last OBC resets it. If timeout occurs, one of the two block characters (ABC) is missing, and recovery of the message is carried out.

13.3.12 TFU Loop Switch and Test of Remote Line

For line testing purposes, the loop switch on the TFU is set to Test mode. The characters sent down the line turn round in the TFU and return to the channel unit. The channel unit normally responds with ACK, which again turns round in the TFU and returns to the channel unit. If the line is faulty, the hardware times out, and an OER interrupt is sent to the program.

A special request, test remote line, is available. SYN transmission is used for this test. The request ends when two SYN characters are sent to different selector units connected to the same TFU.

Examples of remote line testing are shown on the following page.

```
1.  SYN       ACK       SYN       ACK
      out                 out
    ------>   <------   ------>   <-----

2.  SYN       ACK       SYN       OER
      out                 out
    ------>   <------   ------>   <------

3.  SYN       OER       SYN       ACK
      out                 out
    ------>   <------   ------>   <------
```

Conclusions:- The loop switch is probably not in Test mode.
2 and 3 indicate that only one selector unit is active.

```
4.  SYN       OER       SYN       OER
      out                 out
    ------>   <------   ------>   <------
```

Conclusions:- If the TFU is in test mode the line is probably broken.
If the TFU is in normal mode it is possible that both selector units are inactive.

```
5.  SYN       SYN       ACK       SYN       SYN       ACK
      out       in                  out       in
    ------>   <------   <------   ------>   <------   <------

6.  SYN       SYN       ACK       SYN       ACK
      out       in                  out
    ------>   <------   <------   ------>   <------

7.  SYN       OER       SYN       SYN       ACK
      out                 out
    ------>   <------   ------>   <------   <------
```

Conclusion:- Line OK.

```
8.  SYN       SYN       OER       SYN       OER
      out       in                  out
    ------>   <------   <------   ------>   <------
```

Conclusion:- Line not OK.

The examples above show that the test remote line request must be repeated several times before any valid conclusions can be drawn.

When DRRT01 detects SYN-in as a reply to SYN-out, it switches status to test mode, i.e. all data inputs are blocked and all outputs to devices are queued. When the loop switch is set to the normal position from the test position, a power-on character is sent to the channel unit and the driver changes status from test to normal. Queues are resolved and inputs are allowed.

Only the traffic on the faulty channel is affected. One remote channel may be in test status while normal work is in progress on the other.

13.3.13 Error Accumulators

For test purposes, there are error accumulators in the control work table of each channel, updated while the channels are active.

These accumulators are:-

* Local - NAK.
  - Retransmission faults.
  - Undefined interrupts.
  - Software timeouts.

* Remote - NAK.
  - Block transmission errors.
  - Undefined interrupts.
  - Software timeouts.

Trouble in hardware configurations can often be diagnosed by analyzing the contents of these accumulators.

13.3.14 The Log Function

A log function is built into the channel unit driver for the testing of both device drivers and channel units. Every input and output is logged in a special circular log buffer.

13.4 PTS 8000 LOCAL & REMOTE TERMINAL DRIVERS

PTS 8000 systems use the V-24 interface for the connection of work-stations. The cards used to implement this interface are called ASCU4Z and SALCUZ for local and remote connection respectively; the associated drivers are DRAS01 and DRSL01.

13.4.1 ASCU4Z & SALCUZ Configuration

A maximum of 6 ASCU4Z's and 4 SALCUZ's may be configured in an 8000 system; devices which can be connected are PTS 8046 (6346) displays, PTS 8071/72 (6271/72) keyboards, and PTS 8081 printers.

Each ASCU4Z supports 4 half-duplex lines; a SALCUZ supports 1 full-duplex line. This gives a maximum of 24 half-duplex lines for local connections, and 4 full-duplex for remotes.

Possible device combinations connecting to an ASCU4Z are as follows:-

* 2 displays with keyboards.
* 1 display with keyboard + 1 printer.
* 2 printers.
* 4 displays.
* 2 displays + 1 display with keyboard.
* 2 displays + 1 printer.

A SALCUZ can connect 1 display with keyboard, 1 display, or 1 printer.

13.4.2 Driver Switching in PTS 6800

By means of a new module (DRSW01) added to DRRT01 or DRLT01, the ASCU4Z and SALCUZ cards may be used in a PTS 6800 system. This module performs the necessary routing of I/O for the appropriate drivers; DRAS01, DRSL01, or DRRT01/LT01 (see figure 12.1, boxes 34, 35, 36, & 37).

If DRSW01 is used, then all calls from device drivers are made to it instead of to DRRT01/LT01. Five calls are possible:-

a. CF A5,OUTPUT
b. CF A15,OUTLIN
c. CF A5,STREG A5
d. CF A15,GETCHR
e. ABL LDREG

Execution of calls c, d, and e is done within DRSW01 in the same way as in DRRT01/LT01. The other two, a and b, involve the checking of channel parameters and are described below.

a. CF A5,OUTPUT

This call is used for normal output from the device driver. DRSW01 checks the character in bits 8-15 of register A2; if it is not /82 or /93 and ASCU4Z's or SALCUZ's are configured, then either DRAS01 or DRSL01 is accessed.

If the character is /82 or /93, then an immediate return is made via A5 and the interrupt routine of the calling device. These characters indicate block start and end for the old remote procedure, and have no meaning for ASCU4Z and SALCUZ.

b. CF A15,OUTLIN

This is a request for output of a character with no special handling, and without returning through the A5 stack. DRAS01 or DRSL01 is accessed immediately in appropriate configurations.

13.4.3 Device Connections to ASCU4Z & SALCUZ

1. Display with Keyboard.

   ASCU4Z - Since this device requires one line for input and one for output, only two may be connected to an ASCU4Z, each occupying a pair of lines. Physically there is only one connection for each device, and this must be to the higher number line in the pair, ie. 2 or 4.
   The logical configuration numbers do not have to be the same as the physical connections; for example, two displays with keyboards could be configured as 2 and 3, leaving 1 and 4 unused.

   SALCUZ - Only one input and one output line are available, so that only a single display with keyboard may be connected.

2. Printer.

   ASCU4Z & SALCUZ - This is connected in the same way as a Display with Keyboard.

3. Display.

   ASCU4Z - Displays can be connected to any of the lines, making a maximum of four devices possible.

   SALCUZ - Only one display can be connected; this is because only a single output line is available.

The first word in the Device Work Table, DWTCHP, contains the channel parameters, as follows:-

Bit 0 - Device without data request when set.

Bits 1 to 3 - Device Address.
1 = Keyboard.
2 = Display Indicator.
3 = Display / Printer.

Bits 4 to 6 - Line Number.
0 = line 1, ASCU4Z 1, 3, or 5 / SALCUZ 1.
1 = line 2, ASCU4Z 1, 3, or 5 / SALCUZ 2.
2 = line 3, ASCU4Z 1, 3, or 5 / SALCUZ 3.
3 = line 4, ASCU4Z 1, 3, or 5 / SALCUZ 4.
4 = line 1, ASCU4Z 2, 4, or 6.
5 = line 2, ASCU4Z 2, 4, or 6.
6 = line 3, ASCU4Z 2, 4, or 6.
7 = line 4, ASCU4Z 2, 4, or 6.

Bit 7 - Input Device if set.

Bits 8 to 11 - Not Used.

Bits 12 to 15 - Channel Index.
0 - /B Reserved for CHLT/CHRT.
/C ASCU4Z 1 or 2.
/D ASCU4Z 3 or 4.
/E ASCU4Z 5 or 6.
/F SALCUZ 1 to 4.

13.4.4 DRAS01 & DRSL01 Channel Work Tables

Each line has a Channel Work Table (CWT) associated with it, laid out as follows:-

CWT for DRAS01

CWTLDW	Device using the line
CWTLON	Last / next output characters
CWTINR	INR instruction
CWTOTR	OTR instruction
CWTCIS	CIO Start instruction
CWTEQ	Queuing DWT
CWTCIH	CIO Halt instruction
CWTSST	SST instruction
CWTTST	Channel status

CWT for DRSL01

CWTLDW	Device using the line
CWTINR	INR instruction
CWTOTR	OTR instruction
CWTCIS	CIO Start output instruction
CWTTP	Timer pointer
CWTEQ	Queuing DWT
CWTCIH	CIO Halt output instruction
CWTSST	SST output instruction
CWTTST	Channel status
CWTCSI	CIO Start input instruction
CWTSSI	SST input instruction
CWTTP2	Timer pointer 2

CWTLDW Pointer to the DWT of the device using the line. Set to zero if the line is free.

CWTLON Left byte is the last output character (CWTLOC); right byte is the next (CWTNOC). CWTNOC is zero if no character is waiting.

CWTEQ Pointer to the DWT of a device which has requested use of the line when it is already occupied.

CWTST Status bits used by the driver.

13.4.5 Output to ASCU4Z

When output is required CWTLDW is checked to see if the line is free. If the line is free it is opened with CIO Start and the character is stored in CWTNOC. Control is then given to the dispatcher, an interrupt is generated, and the character is transferred to the device.

Control passes to the device driver interrupt handler with A1 set to zero, and the handler returns via ABL LDREG and sends another character which is saved in CWTNOC. The dispatcher regains control and when another interrupt occurs CWTNOC is checked to see if another character is waiting for transfer.

If this is the case CWTLOC and CWTNOC are updated and control again passes to the interrupt handler of the device driver. If no character is waiting, or if it is /03, the transfer is terminated by CIO Halt. Only a part of the request will be transferred at a time if all the characters do not reach the communication driver before an interrupt occurs.

After the transfer is finished another interrupt is generated and this must be followed by an SST instruction. Before control is given back to the dispatcher CWTNOC and CWTEQ are checked to see if a new character is waiting for output to the same device, or if another device is queued waiting for the line. This can only happen if the display driver is waiting during indicator output, or vice-versa.
In this case output must be started again with CIO Start. CWTLDW, CWTLOC, CWTNOC, and CWTEQ are updated.

13.4.6 Output to SALCUZ

CWTLDW is checked when output is required to see if the line is free. If it is, a character is transferred to the device using CIO Start, OTR, and CIO Halt, and control passes to the dispatcher. When the transfer is terminated an interrupt is generated and SST is executed, giving line status information.

If bad status is detected (not operable), control is given to the device driver interrupt handler with A1 set to 1. DRSL01 then continuously scans the line and performs recovery when possible.

For good status, control is passed to the interrupt handler with A1 set to zero. Before control is passed a check is made to see if another device is queued for the line, in which case the request is serviced.

13.4.7 Display Indicator Output

Special actions are taken by ASCU4Z and SALCUZ when output to a display indicator is requested. The keyboard where the indicator is located is directly connected to the screen; A character /1C is sent before the requested output and this causes the output to be routed to the keyboard instead of the screen. This is accomplished internally and makes no difference in programming terms.

13.4.8 Input from ASCU4Z & SALCUZ

Interrupts are generated for input characters which are looked after by the interrupt handlers of the individual device drivers. Neither parity nor any other kind of controls are implemented.

13.4.9 Recovery

After program loading each line configured for output is opened and the recovery routines for the devices are executed. Lines configured for input are opened for input.

For SALCUZ recovery is also performed when a line has become operable after being non-operable. DRSL01 transmits /00 twice per second as long as the line is not occupied with normal output.

If a carrier failure is detected on the input line, or it is not operable, an attempt is made to reopen the line every second until it becomes operable again. The device recovery routines are then executed.

Note: Since there is no signal indicating power-up from the terminals, the states of the terminals are lost during a power-off sequence at the terminal end; the keyboard lamps and key-lock positions are not updated at power-on.

14.1 GENERAL

Previous chapters have described the way in which interrupts (LKM requests, completion of I/O action, etc.) are processed by the Monitor. When the processing of an interrupt is complete, control is handed to the Dispatcher, which then determines which application or Monitor tasks are able to proceed. If several tasks are able to proceed, a task is chosen on a 'First-In-First-Out' basis within priority level. Registers A1 to A14 are restored for this task and the task is entered via a RTN instruction.

The RTN instruction automatically enables interrupts to occur. Any interrupt of an equal or lower priority which occurred during the processing of the last interrupt would have been queued. On the RTN instruction being executed this interrupt will take effect immediately and control will again be passed to an interrupt handler.

However, if no interrupt has been queued, the task will begin execution. Execution of the task will continue until another interrupt occurs (which may of course be an LKM instruction executed by the task). When processing of this interrupt has been completed the task will again become a candidate for scheduling, and so on.

A task may exit by issuing an LKM request type 3 (EXIT). The dispatcher will then delete all record of the task and the task will cease to exist. The task can be activated again (from another task) but the register contents will then be undefined except for the contents of registers used for passing parameters.

In order to identify and schedule tasks, each task must have a task identifier and priority level. Application tasks must be assigned a task identifier and priority level during system loading (SYSLOD). Monitor tasks (e.g. data management task, segment loading task) have a predefined task identifier and priority level and need not be specified during system generation. These task identities are preceded by #.

Scheduling of tasks by the LKM processors and Dispatcher is illustrated in figure 14.1.

The following notes refer to the numbers in the diagram:-

1. The dispatcher selects the next task to be dispatched by inspecting the dispatcher queue and taking out the task of the highest priority present which has been the longest time in the queue, i.e. the scheduling is performed in FIFO (First-In-First-Out) basis per level.

2. When a task is activated or restarted it is inserted in the dispatcher queue at its priority level, and within that level, put last in the queue.

3. When a task tries to activate another task which is already active, the request is put in the pending queue (Not the task).

4. When the running task performs an exit, the pending queue is checked, and if there is a pending request for the exiting task, then that task is reactivated (see 2).

5. When the running task issues an I/O LKM (with wait), for example, the task will not be considered for dispatching until the I/O is completed. Meanwhile, the dispatcher selects another task according to 1.

6. When the event that the task was waiting for is completed, the task is inserted in the dispatcher queue according to the principle described in 2.

7. A task may perform the request 'Switch task on same level', which means that the task is put last in the queue within its level, and the next task on the same level is dispatched.

The situation may also arise in 1 that the dispatcher queue is empty, i.e. all tasks are waiting for events to be completed. Then the monitor is 'idling' in the 'idle loop' (priority level 63) waiting for an interrupt to occur. The interrupt handler will finally pass control to the dispatcher, after having queued the requesting task in the dispatcher queue.

In MMU systems, the dispatcher takes care of the loading of the MMU with the contents of the MMU table in TTAB for the task to be dispatched. It also checks that the requested code segment is situated in memory, and if it is not, the dispatcher will request loading of the code segment before dispatching the task. When a task is queued in the dispatcher queue, the contents of the MMU table are saved in the task table (TTAB).

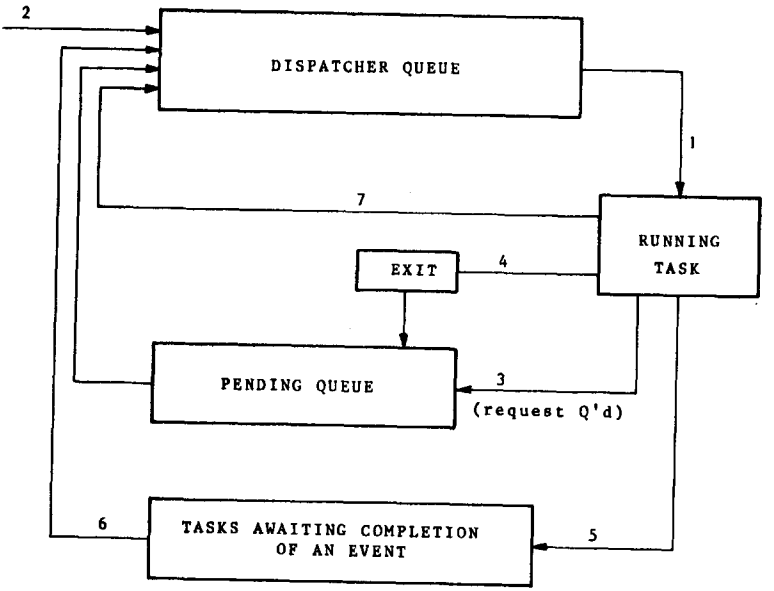
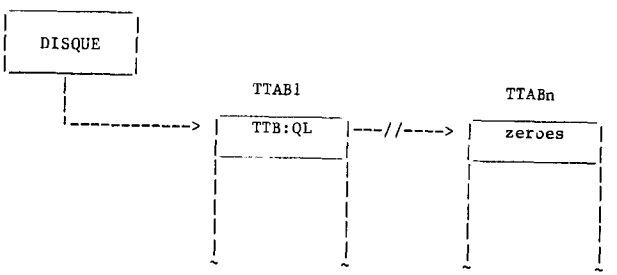


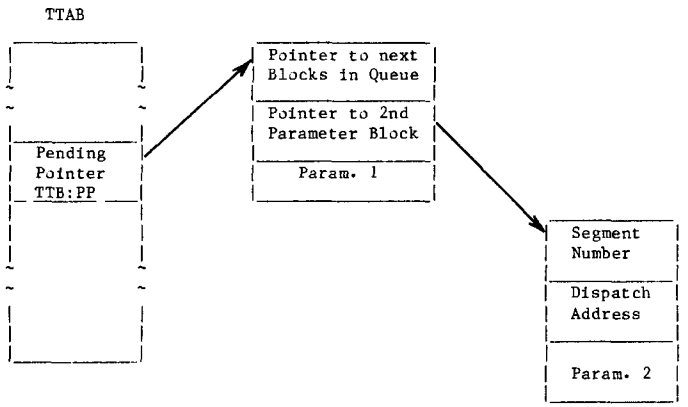
Figure 14.1. Task Scheduling.

14.2 THE DISPATCHER QUEUE



DISQUE is the queue anchor of the dispatcher queue. The contents of this word refer to the first task table (TTAB) in the dispatcher queue. The first word in TTAB is designated TTB:QL and, when it contains zeroes, indicates the end of the dispatcher queue. When the dispatcher queue is empty, word DISQUE contains zero.

14.3 THE PENDING QUEUE



When a task is activated, and the task is already active, i.e. already in the dispatcher queue, then this activation request is put into the pending queue. Six words (2 times three blocks) are obtained from the monitor block pool and pointed to by the word TTB:PP in the task table. These words contain information required to activate the task after an EXIT is performed by that task. When the word TTB:PP contains zero, the pending queue does not have to be checked. Insertions in the pending queue are done on a FIFO basis, except for data management tasks, for which the basis is LIFO.

The meanings of the words are as follows:-

* Pointer to next blocks in queue.
  This word contains a pointer to the next pending request in the queue; end of queue when zero.

* Pointer to 2nd parameter block.
  Pointer to the second block, which contains more information about the pending request.

* Segment number.
  This word contains the segment number in which the activation address (dispatch address) is held.

* Dispatch address.
  This word contains the dispatch address within a segment.

* Parameters 1 and 2.
  Two parameters available to the TOSS monitor.

When an EXIT is performed the pending queue is checked and the blocks released.

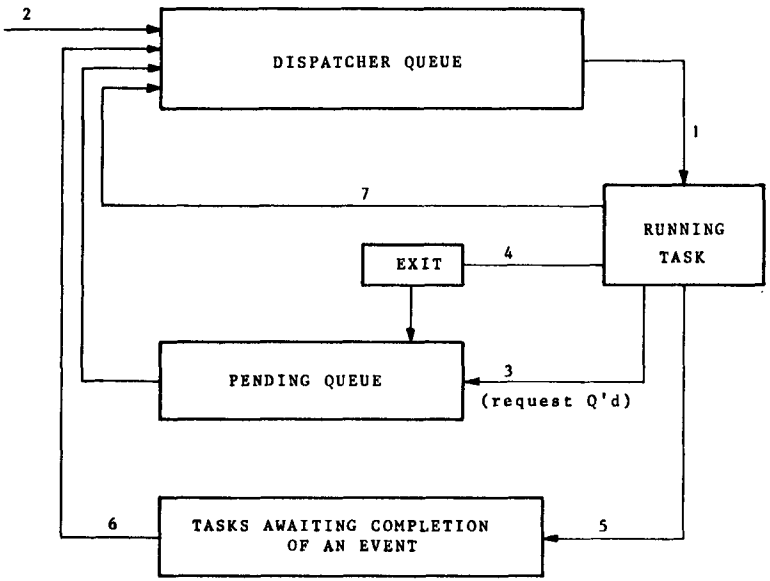
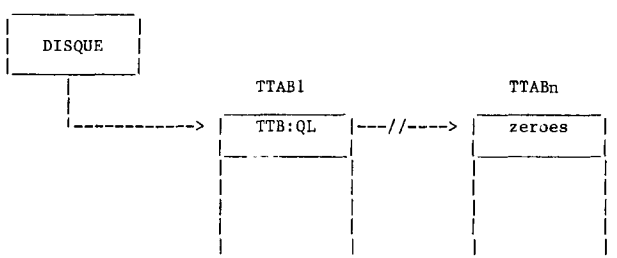


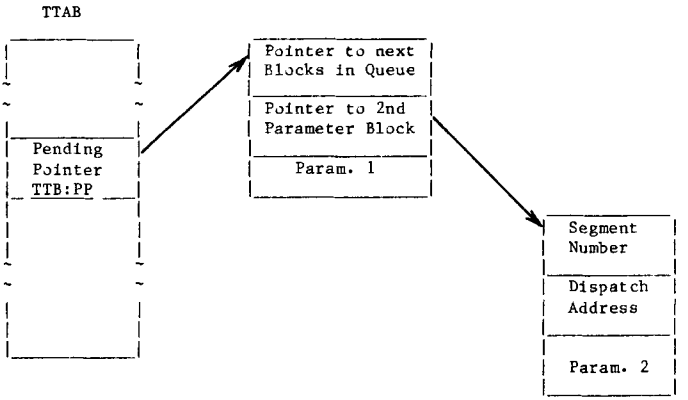
Figure 14.1. Task Scheduling.

14.2 THE DISPATCHER QUEUE



DISQUE is the queue anchor of the dispatcher queue. The contents of this word refer to the first task table (TTAB) in the dispatcher queue. The first word in TTAB is designated TTB:QL and, when it contains zeroes, indicates the end of the dispatcher queue. When the dispatcher queue is empty, word DISQUE contains zero.

14.3 THE PENDING QUEUE



When a task is activated, and the task is already active, i.e. already in the dispatcher queue, then this activation request is put into the pending queue. Six words (2 times three blocks) are obtained from the monitor block pool and pointed to by the word TTB:PP in the task table. These words contain information required to activate the task after an EXIT is performed by that task. When the word TTB:PP contains zero, the pending queue does not have to be checked. Insertions in the pending queue are done on a FIFO basis, except for data management tasks, for which the basis is LIFO.

The meanings of the words are as follows:-

* Pointer to next blocks in queue.
  This word contains a pointer to the next pending request in the queue; end of queue when zero.

* Pointer to 2nd parameter block.
  Pointer to the second block, which contains more information about the pending request.

* Segment number.
  This word contains the segment number in which the activation address (dispatch address) is held.

* Dispatch address.
  This word contains the dispatch address within a segment.

* Parameters 1 and 2.
  Two parameters available to the TOSS monitor.

When an EXIT is performed the pending queue is checked and the blocks released.

15.1 GENERAL

TOSS Monitor Tables exist in a variety of shapes and sizes and are created or deleted under various circumstances. The tables can be divided into three main groups depending on the time of their creation, i.e. tables that are created at:-

* linking time
* loading time
* run time

The tables can also be considered as two other types:-

* Read-only tables (type R).
* Modifiable tables (type M).

Below is a list of some monitor tables and their types:-

* Created at linking time.
  - Device Address Block (DAB), type R.
  - Task Tables (TTAB's) for Monitor Clock task, Load task, File Management Task, and Power failure task; type M.

* Created at loading time.
  - System Control Table (SCT), type R.
  - Task Control Table (TCTAB), type R.
  - Task Table (TTAB), type M.
  - Common Device Table (CDTAB), type R.
  - Device Work Table (DWT), type M.
  - Segment Block Table (SEGTAB), type M.
  - Page Block Table (PAGTAB), type M.
  - Swappable Workblock Table (SWBTAB), type R.

These tables are created by the System loading program, SYSLOD, but for some of the tables prototypes are created at linking time.

* Created at run time.
  - File Descriptor Block (FDB), type M.

This table is created by Data Management at file assignment time and dissolved when the file is closed.

All tables described above, except for the FDB, are tables existing from the time of activation of the first task, and then never dissolved during program execution.

Two tables are created by File Management when a file is opened and dissolved when it is closed. These are the File Work Table (FWT, type M) and the Extent Work Table (EWT, type R). The area used by EWT's can be used for other purposes after it has been released, whereas the FWT areas may only be used for FWT's.

Monitor blocks (3-word size) used dynamically by, for example, the timer routine, could also be regarded as tables created at run time.

There is a pool of monitor blocks from which the different monitor processors can get blocks when they need to save some information temporarily. This could be, for example, when a requested system resource is busy and the request has to be put in a queue.

The requesting task's calling parameters are saved in such a block, from which they can be retrieved when the queued request is taken into execution. The blocks are released and returned to the block pool when the parameters have been taken care of by the resource.

The pointers in TCTAB occur in the order in which tasks were defined during task definition.

There is normally one DWT for each physical device in the system. Device Work Tables are also used for Intertask Communication although they are not in this case related to any physical device.

Each logical device has a file code which is used to select the device for I/O requests. A DWT is assigned to each file code.

For this reason, a table of file codes are included in the TTAB. This table is referred to as the Task Configuration Table and is only accessible from its own task.

Identical terminal devices should preferably have the same file codes.

Some devices are, however, common to all tasks, e.g. discs and cassette drives. These common devices are listed in the Common Device Table, CDTAB, which has the same organization as the configuration table in TTAB.

The file codes for common devices should be different from each other, and from other device file codes.

CDTAB is referenced when a file code is not found in the configuration table of TTAB.

There is no restriction on sharing one device among several tasks, and one device may also be referenced by several file codes.

15.2 SYSTEM TABLE

The monitor module SYSTAB holds the interrupt vectors, the system control table (SCT), the system stack, and the idle loop.

The interrupt vectors are stored from address X'0000' to X'007E'. The corresponding interrupts are defined in the standard interrupt level table in Chapter 7.

15.3 SYSTEM CONTROL TABLE (SCT)

The SCT is the central table in the TOSS monitor (figure 15.1).

byte		
/9E	SCTMSZ	memory size (kB)
/A0	SCTSFA	start of free area (2 words)
/A4	SCTEFA	end of free area (2 words)
/A8	SCTIPL	program loading device (C0,C1,F0,F1,F4,F5,F8,F9) cassette, flex-disk, or disk.
/AA	SCTANO	application number
/AC	SCTADA	application disk sector address (2 words)
/B0	SCTIOE	application restart address
/B2	SCTTCT	TCTAB address
/B4	SCTCDT	CDTAB address
/B6	SCTPAG	PAGTAB address
/B8	SCTSWB	SWBTAB address
/BA	SCTNOP	number of pages
/BC	SCTPSZ	page size (bytes)
/BE	SCTMMC	MMU-table common part entry (index rel.TTAB entry)
/C0	SCTLAC	logical address of common part
/C2	SCTMMP	MMU-table page entry (index rel. to TTAB entry)
/C4	SCTLAP	logical address of pages

SCT (cont)

/C6	SCTNPE	number of page entries
/C8	SCTSTB	system stack base
/CA	SCTOPT	system options
/CC	SCTBUG	debugger address (0 = not included)
/CE	SCTDMT	Data management information
/D0	SCTDMI	DM index record buffer size
/D2	SCTFWT	File Management; FWT address
/D4	SCTNOF	DM/FM; number of files
/D6	SCTNFT	DM/FM; number of files per task
/D8	SCTFWL	FM; FWT length in bytes
/DA	SCTBLK	number of blocks per task
/DC	SCTDCT	DC task in system

Figure 15.1. System Control Table.

SCTMSZ Memory size.
This word contains the memory size in kB, in Hex format.

SCTSFA Start/end free area.

SCTEFA Two words are used to contain the start address of the free area in memory. Word /A0 contains the most significant part of the memory address, and word /A2 the least significant. Two words at address /A4 contain the end address of the free area. This is the part of memory which is used neither by the application nor by the TOSS monitor.

SCTIPL Program loading device.
Contains the codes X'00C0', X'00C1', X'00F0', X'00F1', X'00F4', X'00F5', X'00F8', or X'00F9' for loading from cassette, 6875/6876 disk, 8863 disk, or flexible disk respectively.

SCTANO Application number.
Number as present on the disk when a create file utility was run to create the load file for this application.

SCTADA Application disk sector address.
Physical sector number at which the application file starts.

SCTIOE Application restart address.

SCTTCT TCTAB address.
This word contains a pointer to the Task Control Table.

SCTCDT CDTAB address.
This word contains a pointer to the Common Device Table.

SCTPAG PAGTAB address.
Contains a pointer to the page table, or is zero when paging is not used.

SCTSWB SWBTAB address.
Contains a pointer to the swappable workblock table, or is zero when no swappable workblocks are used.

SCTNOP Number of pages.
The number of pages reserved in memory.

SCTPSZ Page size.
The size of a page in bytes.

SCTMMC MMU table common part entry (index relative to TTAB entry).
This word contains a negative displacement relative to the TTAB entry. It points to a word in the MMU table which precedes TTAB. The entry in the MMU table to which it points contains the start address of the common data area (common to all tasks in the system).

SCTLAC Logical address of common part.
This word points to the logical start address of the common area (code part) in segment zero. This common area starts with the table P:MTAB, followed by P:PIL, etc.

SCTMMP MMU table page entry (index relative to TTAB entry).
This word contains a negative displacement relative to the TTAB entry. It points to a word in the MMU table which precedes TTAB. The entry in the MMU table to which it points contains the start address of the code page.

SCTLAP Logical address of page.
This word contains the logical start address of a code page.

SCTNPE Number of page entries.
This word indicates how many entries in the MMU table are used for a code page.

SCTSTB System stack base.
This word contains the physical start address of the A15 stack.

SCTOPT System options.
Indicates which options are present in this TOSS Monitor.
The list below indicates the meanings of bits when set:-
No bits set = Standard (No MMU, no disk paging).
15 = MMUPAG (MMU used, no disk paging).
14 = DSKPAG (Disk paging and no MMU).
13 = File Management.
12 = Swappable Workblocks.
11 = LKM Load Program included.
10 = LKM Get/Release/Attach page included.

SCTBUG Assembler debugger.
When zero, the assembler debugger is not included in the system, otherwise it contains the address of the start point of the debugger.

SCTDMT Data management.
Bits 15, 14, 13, and 12 of this word are used to indicate which DM tasks are included in this TOSS monitor.
The list below indicates the meanings of bits when set:-
No bits set = DM not included.
15 = DM Task #D.
14 = DM Task #E.
13 = DM Task #F.
12 = DM Task #G.

SCTDMI DM, index record buffer size.
This word contains a value indicating the maximum size of an index record.

SCTFWT FM, pointer to File Work Table (FWT).

SCTNOF DM/FM, number of files.
This word contains a number indicating the maximum number of files present.

SCTNFT DM/FM, number of files per task.
This word contains a number indicating the maximum number of files allowed for a task.

SCTFWL FM, length in bytes of the File Work Table (FWT).

SCTBLK Number of blocks per task.
This word contains a number indicating the number of blocks per task reserved in the monitor blockpool. No blocks are reserved for the load and power failure tasks.

SCTDCT DC task in system.
When zero no DC task is included in the system, otherwise one.

15.4 TASK CONTROL TABLE (TCTAB)

This table, pointed to by SCTTCT in the System Control Table, holds pointers to all Task Tables (TTAB's) in the system, see figure 15.3.

The first word, TABLEN, contains the length of TCTAB in bytes, including the table length word.

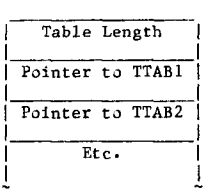


Figure 15.2. Task Control Table.

15.5 TASK TABLE (TTAB)

A set of instructions shared by a number of tasks, together with a task table (TTAB), is unique to a task and completely defines it.

The TTAB is used to describe the task configuration and task status, and also as a save area for registers.

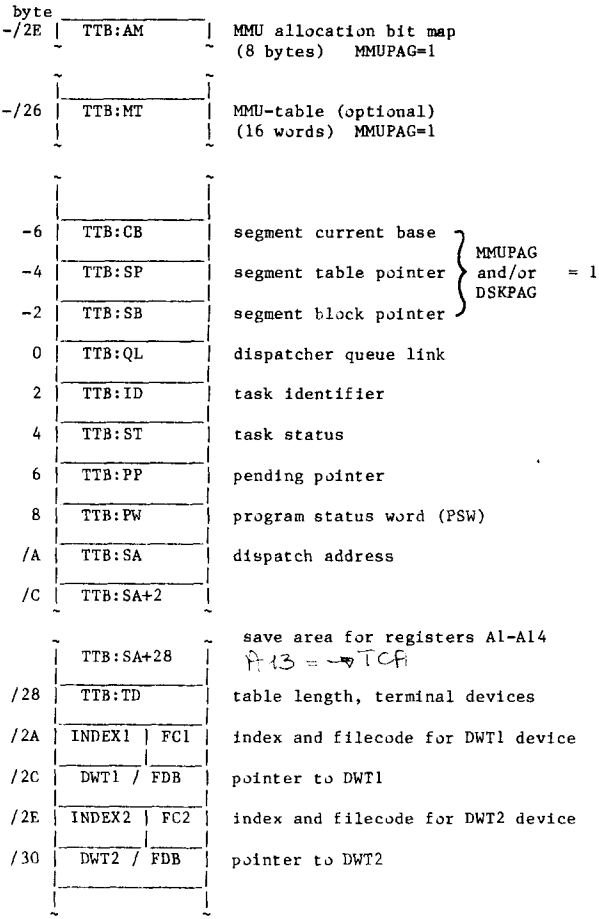


Figure 15.3. Task Table (TTAB).

TTB:AM One bit per kB sub-entry in TTB:MT.
If a bit is set, then the corresponding entry in TTB:MT points to a block of 1kB which is in use.

TTB:MT Included in MMU systems. 16 words holding the task MMU table references to physical core; filled by SYSLOD.

TTB:CB Pointer to a word in the application where the current segment is stored.

TTB:SP Pointer to application's segment table.

TTB:SB Included in MMU and/or disk paging systems. Pointer to the actual segment block in SEGTAB.

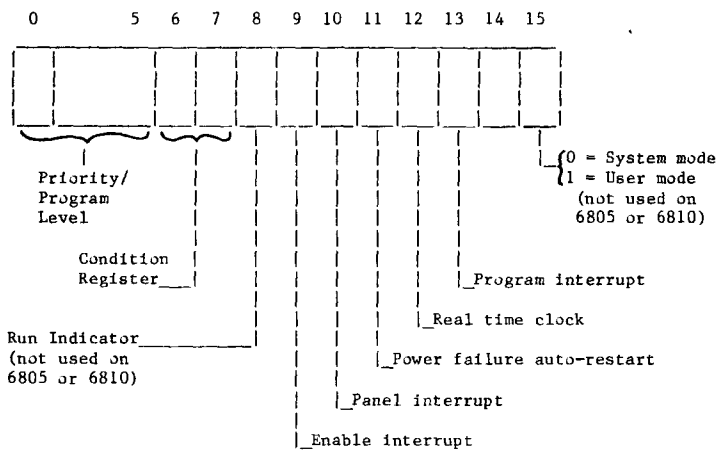
TTB:QL Dispatcher Queue Link.

TTB:ID Task identification.

TTB:ST Bits 0- 7 Reserved.
Bits 8-15 Task priority level.
Level 49-51 = Monitor tasks.
Level 52-62 = Application tasks.
Level 63 = Idle task (wait).

TTB:PP Pointer to pending queue when trying to activate a task which is already active; some parameters have to be saved and the request is inserted in the pending queue. The pending pointer is the queue anchor.

TTB:PW Program Status Word (PSW).



TTB:SA Dispatch address.
This word contains zero if the task is inactive, else the address of the next instruction to be executed for this task. If the task is running it contains the last used dispatch address.

TTB:SA+2 to TTB:SA+28 } Save area.
These words are used to save registers A1 to A14.

In each TTAB a task configuration device table is included, describing the task configuration.
The length of this table is task (configuration) dependent.

TTB:TD Table length; length of the table including this word.

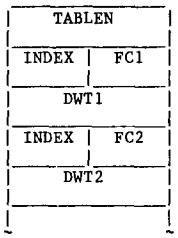
INDEX Used by drivers to select a device dependent part.
Makes it possible for several devices to share the same DWT (e.g. magnetic tapes).

FCx File code of a device.

DWTn Pointer to DWT with file code present in previous word.
Some word pairs are reserved for Data Management when local files are used by this task.
The file code and the FDB pointer are filled by the ASSGN command. One pair per file.

15.6 COMMON DEVICE TABLE

CDTAB



This table is pointed to by SCTCDT in the System Control Table and it contains information about devices which are common to all tasks. It is searched if the filecode is not found in the Terminal Device Table part of TTAB.

TABLEN Table length in bytes, including this word.

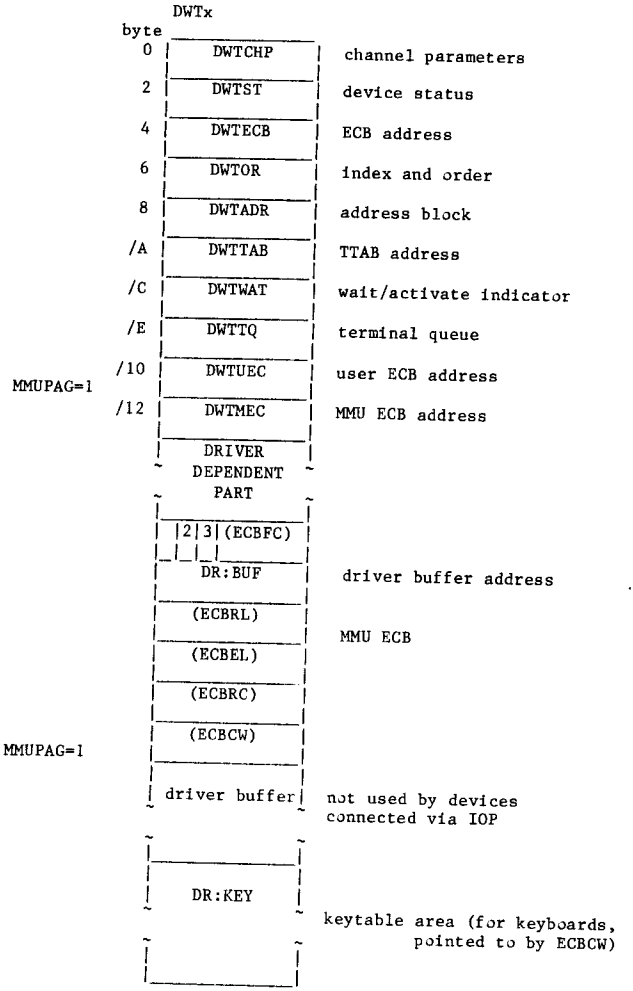
INDEX Used by drivers to select a device. Makes it possible for several devices to share the same DWT (e.g. magnetic tape, flexible disk, etc).

FCx File code of a device.

DWTn Pointer to DWT with file code as present in previous word.

Some word pairs may be reserved for data management when common files are used. The file code and FDB pointer (File Descriptor Block) are filled by the ASSGN command. Each file uses one pair.

15.7 DEVICE WORK TABLE



DWTCHP Channel Parameters for CHLT and CHRT.
Some bits have a standard meaning for these channel units.
For other channel units it is channel unit/device dependent.
Bits with standard meanings are as follows:-

```
Bit 0      : Device dependent;
             0 for Displays.
             1 for Signal Displays.
             TTP's; 0 for TP01 & 02.
                    May be 1 for TP03.
Bits 1-3   : Device address.
Bits 4-6   : CHLT; line number 0-7.
             CHRT; bit 4 set = 2nd terminal on channel.
                   bit 6 set = 2nd channel.
             ASCU4Z's; 1/3, 2/4 on lines 0 to 3.
             SALCUZ's; 1 to 4
Bit 7      : Set = input device.
Bits 8,9   : Printers; 8 set = voucher in.
                       9 set = journal paper in.
Bits 10,11: Keyboards; specifies type.
             Displays; bit 10 = 1 for 6385 or 6386.
                       bit 11 = 0 for 6344 or 6346.
                       bit 11 = 1 for 6351 or 6342.
             Signal Displays; bit 10 = 1 for 6241.
                              bit 11 = 1 for 6232, 34, 36, 71,
                                             6272, 6331.
Bits 12-15: Channel indicator.
             /0  CHLT1
             /1  CHLT2
             /2  CHLT3
             /3  CHLT4
             /4  First channel CHRT1
             /5  Second channel CHRT1
             /6  First channel CHRT2
             /7  Second channel CHRT2
             /8  First  channel CHRT3
             /9  Second channel CHRT3
             /A  First  channel CHRT4
             /B  Second channel CHRT4
             /C  ASCU4Z 1/2
             /D  ASCU4Z 3/4
             /E  Reserved
             /F  SALCUZ 1-4
```

DWTST Status Word.

Bit 0 : Set means device not busy. The bit is checked when module TIO is entered. If the device is busy the request is put into the device queue. If it is not busy, the bit is reset by TIO. When I/O is completed it is set again by module TENDIO.

Bit 1 : reserved.

Bit 2 : Set means device in echo mode. When I/O is performed with echo or with exclusive access for data management, this bit is set in the DWTST and in the DWTST of the echo device. Setting is done by module TIO. Completion of the I/O results in the resetting of this bit in both DWT's by module TENDIO.

Bit 3 : Set means recovery trial (LRC, VRC, or parity error).

Bit 4 : Set means device ATTACHed. This bit is set by the ATTACH function and reset by the DETACH function. It is checked by module TIO.

Bit 5 : Line feed control for printers (top of form). Only after a power off/on situation.

Bit 6 : Keyboard; Set means power off before the request. Printers; Set means status change not allowed. (only after power off/on situation).

Bit 7 : Set means interrupt allowed. The channel unit expects an acknowledgement of a character. When ACK is received the bit is reset.

Bits 8-15 : Device dependent information.

DWTECB ECB Address.
With no MMU present, this word points to the user ECB with which the driver is working.
When MMU is present, two situations are possible:-

* A request is performed in system mode, for example by data management. In this case DWTECB contains the user ECB address (physical).
* A request is performed in user mode. The word DWTUEC contains the user ECB logical address and DWTECB is the same as DWTMEC, which points to the MMU ECB (physical address).

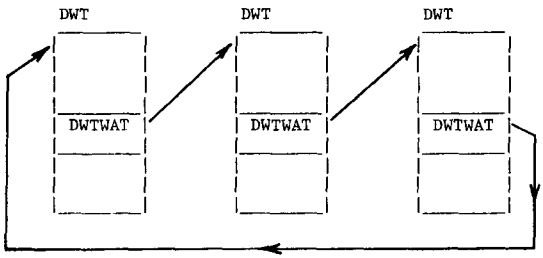
Word DWTECB is also used in recovery procedures to check whether an I/O is running. If no I/O is running the word contains zero.

DWTOR Index and Order.
Holds index and order code. Index is used to process a device dependent part of the driver.

DWTADR Address Block.
Pointer to the device address block (DAB). The DAB contains, among other things, activation, interrupt, and recovery addresses for the driver.

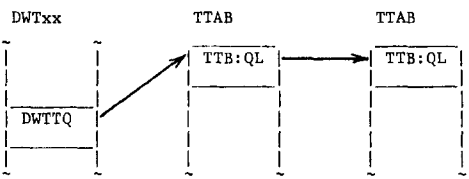
DWTTAB TTAB Address.
Pointer to the task table (TTAB) of the requesting task.

DWTWAT Wait/Activate Indicator.
0 = request in no-wait mode.
1 = request in wait mode.
This word is also used to chain DWT's in case of multiple wait requests or to hold the start address for I/O-and-activate requests. For multiple wait bit 15 is 1, for I/O-and-activate it is zero.



Example of circular DWT chain for multiple wait requests:-
When one I/O is completed, the DWTWAT words are set to zero, the other two I/O's are aborted and another multiple wait request is issued for the other two devices. The chain is then set up for these two devices only.
For CREDIT this is done in the interpreter and is transparent to the user.

DWTTQ Terminal Queue.
When a device is busy, the requesting tasks are queued on the DWT. This queue is the device queue and DWTTQ is the queue anchor. The device busy bit is reset in DWTST.



The word TTB:QL in the TTAB is used for task chaining. The end of the chain is indicated by zeroes in TTB:QL.
The word DWTTQ is also used as a pointer to the output queue. (see section 13.3.5, word CWTEQ in the Channel Work Table).

DWTUEC User ECB Address.
This word contains the logical address of the user ECB when an MMU is configured and the request is done in the user area. (see also DWTECB).

DWTMEC MMU ECB Address.
Contains the physical address of the MMU ECB in the DWT. (see also DWTECB).

ECBFC ECB Filecode.
This word is a copy of the word in the normal ECB, except that bits 2 and 3 are used to form an 18-bit buffer address when I/O is performed via an IOP. This means that devices attached to an IOP need not have an MMU buffer in the system area, since the whole memory can be accessed with an 18-bit address.

DR:BUF Driver Buffer.
This is the MMU buffer in which characters are stored during an I/O operation. Its size is specified at SYSGEN time.

DR:KEY In a keyboard or console typewriter device work table, an area for a keytable is reserved after the MMU ECB. The size of this area (DR:KEY) and the size of the driver buffer (DR:BUF) are specified during system generation (SYSGEN). For a terminal device, the system loading program (SYSLOD) reserves buffers according to the length stated in BUFLEN in the Driver Address Block (DAB).

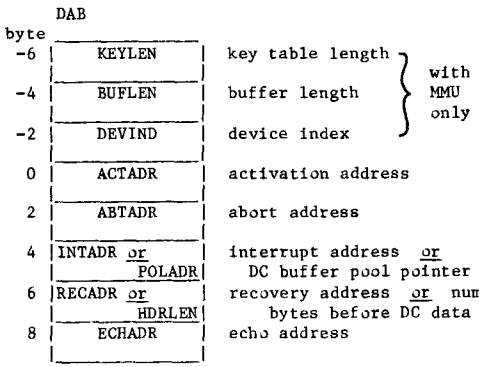
The driver dependent part of the DWT is situated between DWTTQ and the MMU ECB and its length and contents vary from driver to driver. Its contents include timer pointers, special queue anchors, A5 stack, save area for 2 to 3 registers, pointers to echo device DWT (for keyboards), volume name areas (for disks), etc.

Field displacements in the driver defined part are different depending on whether the system is running with MMU or not. SYSGEN performs all necessary updating by setting the conditional assembly parameter MMUPAG to the appropriate value.

15.8 DRIVER ADDRESS BLOCK

The Driver Address Block (pointed to by DWTADR in DWT) has the format shown below.

The three words with negative displacements (KEYLEN, BUFLEN and DEVIND) are used only in systems with MMU.



KEYLEN Keytable Length.
Contains the length of the keytable area (DR:KEY) in bytes.

BUFLEN Buffer Length.
Size of the buffer area (DR:BUF) in bytes. A default value is supplied for each device at SYSGEN if not explicitly defined.

DEVIND Device Index.
Used at I/O requests to select the appropriate routine in TIO or TENDIO for the moving of data and keytables, transferring user buffer address to the IOP, etc. Contents as follows:-

-2 No special action required (e.g. data communications).
0 Data management.
2 Device running on programmed channel.
4 Device running on IOP.
6 Keyboard/Console Typewriter.
8 Intertask communication.
10 Disk Drivers.

When a read or write request has been issued, a check is made that the requested length does not exceed the driver buffer length (BUFLEN). If it does, return code /8008 is set and no I/O is performed. A similar check is performed before transferring the keytable for keyboard/console requests (KEYLEN).

Three drivers (DRLP01, DRCR01, DRFD01) are given device index 2 when running on an IOP. They have their own buffers in the system area and this thus resembles programmed channel running.

The other five words usually point to special routines in the driver, the names being derived from their functions. Not all drivers have the 5 words, it depends on the driver type. When the address is zero, the routine does not exist.

INTADR and RECADR are renamed POLADR and HDRLEN for DC drivers, and have different functions.

ACTADR Activation Address.
Address of the activation routine in the driver.

ABTADR Abort Address.
Address of the abort routine in the driver (abort is device/driver dependent).

INTADR Interrupt Address.
Address of the interrupt routine in the driver.

POLADR For DC drivers only, contains a pointer to the driver's internal buffer pool.

RECADR Recovery Address.
Address of the recovery routine in the driver.

HDRLEN For DC drivers only, this contains the number of bytes which preceed the required data in a buffer.

ECHADR Echo Address.
Address of the echo routine in the driver. This is driver dependent.

15.9 MONITOR BLOCKS

At system start all blocks are held in the monitor block pool. Blocks can be obtained from this pool when monitor processors need a temporary save area for information received from tasks requesting service from busy system resources, or when certain other monitor functions are to be executed. When the requested service is completed, the blocks are released and returned to the pool.

The SCT (System Control Table) contains the word SCTBLK, in which is stored a number indicating how many blocks are reserved per task in the monitor blockpool.
The timer, dispatcher, and pending queues, for instance, all use blocks from the monitor blockpool.

16.1 GENERAL

A Real Time Clock (RTC) is available in PTS systems, control of which is by means of a key switch on the system operator's panel. Key positions are as follows:-

* OFF - RTC stopped.
* ON RTC - RTC running.
* LOCK - RTC running.

Once running, the RTC generates a signal timed from the mains power-supply frequency and it cannot be stopped by a program. The generated signal may be connected to any of the 8 highest priority interrupt levels (0-7) and is it can raise an interrupt every 20ms for 50Hz supplies. The associated interrupt must be cleared using the Reset Internal Interrupt (RIT) instruction and the RTC routine may be used as required within the system.

16.2 CLOCK ROUTINES

The 20ms RTC interrupts are used to update two clocks in the system.

The first clock routine counts five 20ms intervals and schedules the clock task #M every 100ms for dispatching purposes (level 49). After queueing the #M task, the second clock routine, Monitor Clock, is updated to count seconds. This monitor clock routine maintains the time, which can be set by the 'set time' instruction and obtained by the 'get time' instruction. A 24-hour carry is also generated.

When the #M task is dispatched, it updates all timer values in the timer queue pointed to by the word TIMQUE (figure 16.1).

When a timer value becomes zero, a branch to the 'timeout address' is performed. The blocks in the queue are not released.
When the timer value becomes positive, the blocks are released from the timer queue and returned to the monitor blockpool.

Drivers which need timeout functions prepare blocks in the timer queue via a special request. These blocks are then updated by task #M.

The timer queue is pointed to by the word TIMQUE. Two blocks, each of 3 words, are reserved for each timer. The first word points to the next pair of blocks in the queue.

The second pointer points to the block containing the 'timeout address' and two parameters. The third word contains the timeout value (negative). When this value reaches zero, control is passed to the 'timeout address' in the second block. A positive timeout value results in the release of these two blocks.

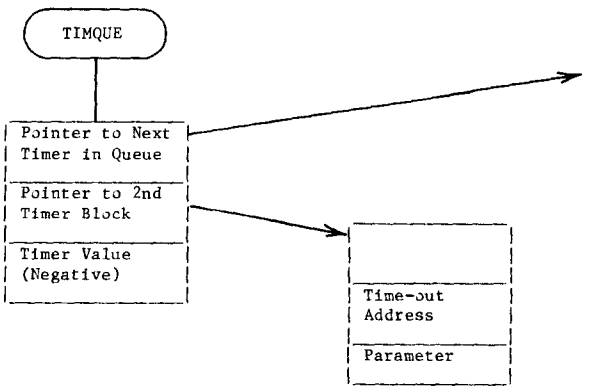


Figure 16.1. Timer Queue.

17.1 GENERAL

This facility provides the processor with the ability to terminate processing in an orderly manner after the detection of a power failure, and to restart and resume processing correctly after the restoration of power.

Apart from separate peripherals, all the system's power supplies, whether within the basic cabinet or extension cabinets, are considered necessary for the correct operation of the system. The failure of any of the supplies will therefore raise the power failure signal.

The power failure signal must be connected to interrupt level 0. When power failure is detected an interrupt is raised and IOP exchanges are inhibited, control of the general purpose bus being given to the CPU.

The associated interrupt routine is executed to save the contents of registers, and if necessary specific areas of MOS memory. Core memory is already protected and thus no loss of data from core occurs even if there is total power failure before completion of the saving routine.

On restoration of power the system restarts and CPU operation continues with the restoration of all registers and areas saved before completing the interrupt routine and returning to the originally interrupted program. The power failure interrupt is reset as necessary by the use of the Reset Internal Interrupt (RIT) instruction.

The power failure signal may also be raised at initial power-on time if the key switch of the system operator's panel is set to the LOCK position. In this position the CPU is started and the restoring routine of the interrupt handler is executed to restart normal operation at the point at which it was suspended.

If the power failure signal is not connected, the CPU will start and remain in the idle state at power-on, or after restoration of power following a failure.

17.2 LIMITS

The power failure interrupt is raised at least 2ms before the voltage drops below the acceptable level. The saving routine should not last more than 2ms.

A power failure interrupt is not raised for detected power losses of less than 5ms. The validity of the contents of a memory location involved in a memory cycle at the time of total failure can not be guaranteed.

17.3 POWER FAILURE / AUTOMATIC RESTART ROUTINE

When the system is loaded the power failure/automatic restart routine is entered. A flag is reset to indicate a system start after IPL and not a power-on situation after power failure.

When a power failure occurs at runtime, a power failure interrupt is raised and the power failure/automatic restart routine is entered. Registers A1 to A8 are saved on the A15 stack and the task #P is scheduled for dispatching on priority level 0. This task saves A15, the value of which will be restored after power-on, and the system halts.

In an MMU system, if a power failure occurs while the system is busy transferring MMU-buffer contents to an application buffer, the transfer is completed before control is given to the #P task.

After power-on, a routine is entered to restore A15 and control is given to routine PFINIT, which starts drivers at their recovery or power-on entries. The addresses are obtained via the power failure table (PFTAB). Control is then passed to the dispatcher and normal processing resumes.

Recovery actions are device dependent and are described in the the Assembler Programmers' Reference Manual, Part 2.

The power failure table (PFTAB) contains the addresses of the driver entries. The length of the table depends on the number of drivers included in the TOSS monitor.

18.1 GENERAL

Memory management in TOSS is available in four different options, depending on the hardware installed. These options are specified at SYSGEN time and their hardware requirements are as follows:-

* MMU Paging.
  - UK01 with MMU; 64 - 128kB memory.
  - 6813 with MMU; 64 - 256kB memory.

* Disk Paging.
  - 6805, 10, 12, 13, or UK01; disk or flexible disk.

* Disk and MMU paging.
  - UK01 with MMU; 64 - 128kB; disk or flexible disk.
  - 6813 with MMU; 64 - 256kb; disk or flexible disk.

* Swappable work blocks.
  - 6805, 10, 12, 13, or UK01; disk or flexible disk.

18.2 MEMORY MANAGEMENT UNIT (MMU)

18.2.1 MMU Structure

The Memory Management Unit is a hardware option which allows memory addressing up to a maximum of 256kB.

The MMU consists of a 16-register table, whose contents can be changed by four special instructions, i.e. Table Load, Table Store, Table Load Register, and Table Store Register.

18.2.2 Address Translation

The CPU operates in either System mode or User mode.

In System mode, memory locations are addressed by using the whole contents of a word, 16 bits.

This allows addressing of the first 64kB of memory and MMU is not used. However, the system is able to extend its addressing by using the extended instructions, which make use of MMU address translation.

Address translation is always performed by the MMU while the CPU operates in User mode.

In User mode, the MMU translates the first four bits of a 16 bit Logical Address into a six-bit physical page address (MMU-registers having previously been loaded). The remaining 12 bits of the Logical Address then select a 4kB slot within the selected page.

It is important to note that the MMU addresses the whole of the memory, not just the area above the 64kB low address space. Pages can equally as well be placed at addresses below 64kB as above.

See figure 18.1 which illustrates MMU addressing of memory.

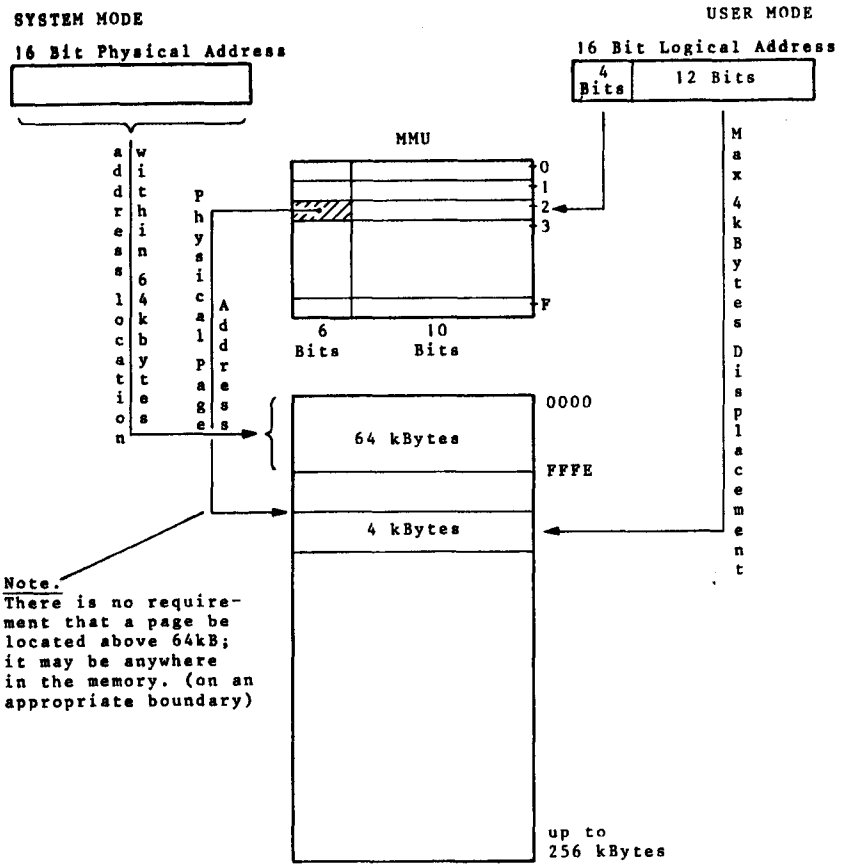


Figure 18.1. MMU Memory Addressing.

18.3 MMU PAGING ONLY

With MMU systems a distinction is made between logical memory and physical memory.

Logical memory is an area addressed in user mode containing data, code pages, etc. These data and code pages are not necessarily contiguous physically.

The transformation of a part of logical memory into physical memory is a function of the MMU.

At any one time 64kB of memory are directly accessible.

The contents of the MMU is called the task window. When CREDIT is used a number of entries in the MMU table are used for data, interpreter, etc. and some entries are left for a code pages. One code page at a time is present in the task window. Which page this is can be found via the segment block address in the task table (TTB:SB in TTAB).

Since all pages are stored in memory when only MMU paging is used, each segment block (SEGBLK) has a corresponding pageblock (PAGBLK). The tables which map these blocks, SEGTAB and PAGTAB, are never changed after program loading. For details of SEGTAB and PAGTAB, see section 18.6.

The code page entry in the MMU table and the segment base in the task control area (T:Axxy for CREDIT) are changed to point to the new code page when a branch is performed from one code page to another. This request is performed without any task switching, which makes it considerably faster than it would be otherwise.

18.4 DISK PAGING, NO MMU (64kB memory)

After allocation of memory for the different tables and work areas, the rest of the memory is divided into code pages. The size of a page is decided at linking time.

The loading of the program into these pages is then controlled by the monitor. Pages are only overwritten between task switches on the same priority level, or on request from the active task. This implies that the number of pages must be at least equal to the number of different task priorities in the system. The loading process is controlled by a special load task running at priority level 49. Application tasks may be running when the load task is executing disk I/O.

The monitor keeps a segment table in memory, SEGTAB, which contains one entry for each segment in the program. Each entry contains information about whether the segment is loaded or not, where it is loaded, PAGTAB, disk address, queue pointers, etc. The new segment address is stored in TTAB and the task control area, T:Axxy.

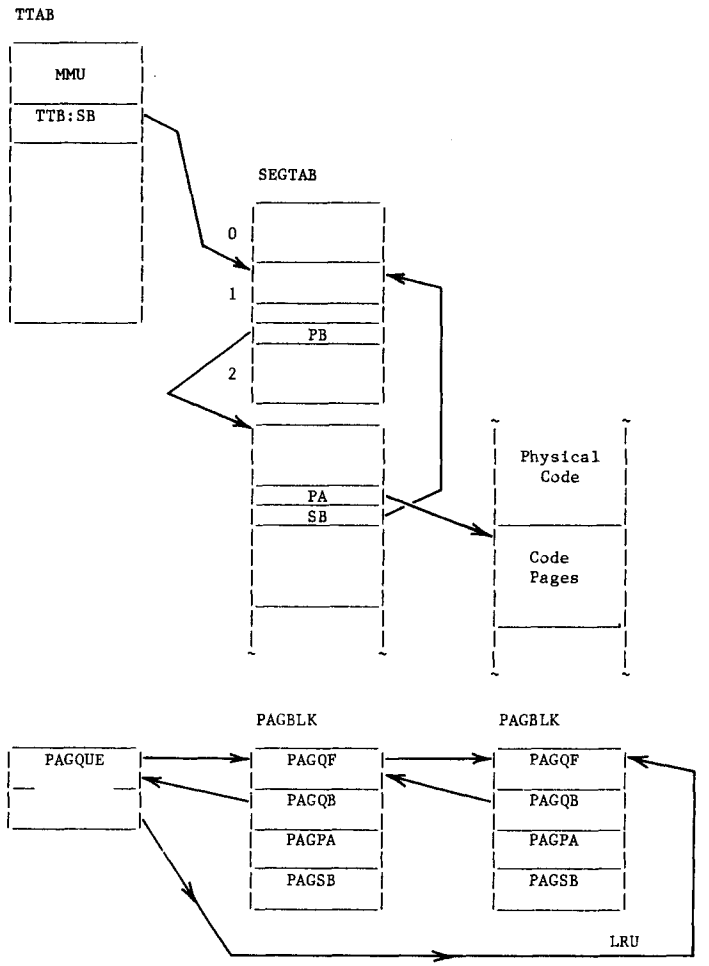


Figure 18.2. Page Queue and Associated Control Blocks.

A page currently being used by a task active in the CPU is called active. All other pages are called inactive, and are placed in a page queue (PAGQUE). When a new segment has to be loaded, space is allocated by taking out one page from this queue using a Least Recently Used (LRU) basis (figure 18.2).

If the requested segment is already present in memory and inactive, then the corresponding page is taken out of the page queue and set active.

The page queue is normally updated each time task switching occurs, which is much more often than segment loading occurs, so that the system can react dynamically to changes in workload conditions.

If the new segment is in the page queue no task switching is performed.

If the new segment has to be loaded from disk, then task switching is necessary. See figure 18.3.

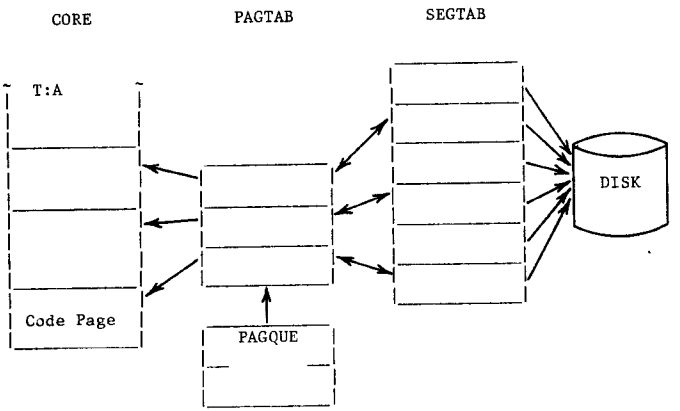


Figure 18.3. Segment Loading from Disk.

18.5 DISK AND MMU PAGING TOGETHER

When both disk paging and extended memory are in use, both previously described methods are combined.

The new segment can either be in extended memory or on disk. If it is in extended memory the loading function is the same as when only MMU was used, and if it is on disk it will be the same as if only disk paging was used.

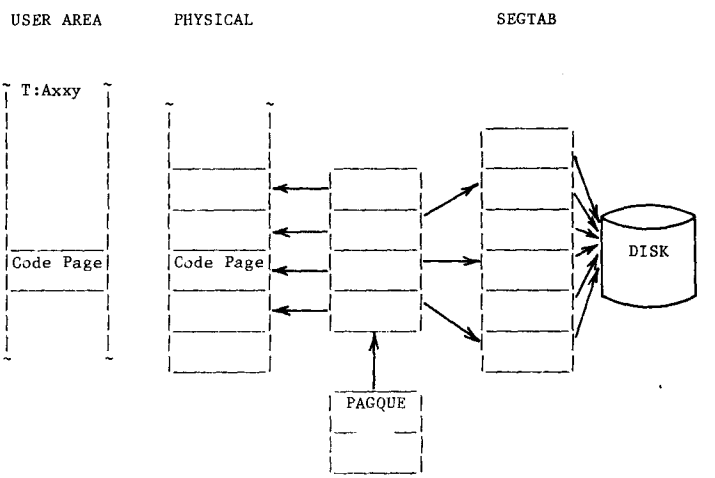


Figure 18.4. Combined Disk and MMU Paging.

18.6 MONITOR TABLES USED BY MEMORY MANAGEMENT

Two tables are central to the memory management system. They are:-

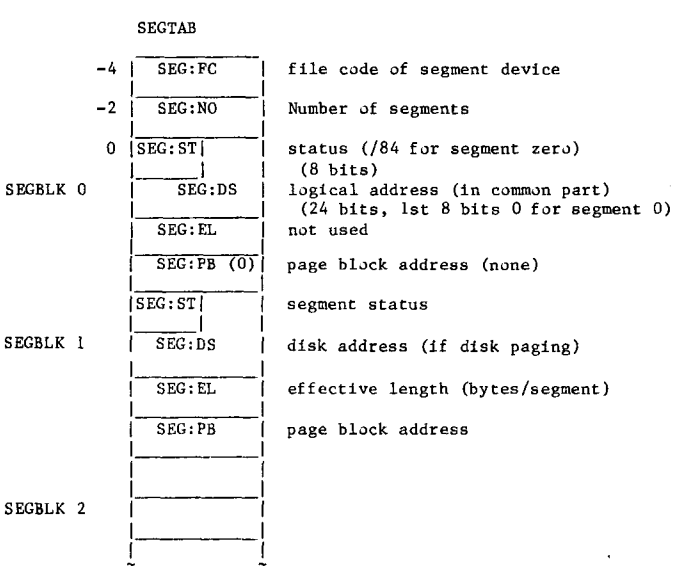
* SEGTAB, the Segment Block Table, which contains segment blocks (SEGBLK's). Each SEGBLK describes, for example, the status and location of a segment.

* PAGTAB, the Page block table, which contains page blocks (PAGBLK's). Each PAGBLK contains, for example, the location of a page in memory (physical address) and a pointer to the corresponding segment block.

Each SEGBLK is related to one segment; SEGBLK 0 belongs to Segment 0, SEGBLK 1 belongs to Segment 1, etc. Each PAGBLK is related to one page in memory; the first one is Page 1, the details of which are defined in PAGBLK 1.

18.6.1 Segment Block Table (SEGTAB)

Each segment is described in a segment block, SEGBLK. All segment blocks are contained in the table SEGTAB which is built by the system loading program, SYSLOD, at configuration time.



SEG:ST Contains the status of the segment. The bits have the following meanings when set:-

Bit 0 : The segment is in memory.
Bit 1 : Loading of this segment is in progress; set and reset by the load task.
Bit 2 : Segment locked, e.g. by the CREDIT debugger; it cannot be overwritten in memory.
Bit 3 : Used to indicate that a task on a higher priority has interrupted this task. The page will not go into the page queue until the lower priority task has returned to it. After that a LKM will cause it to be queued. In this way the page is prevented from moving to different areas of memory.
Bit 4 : Core resident. The segment can not be overwritten in memory.
Bit 5 : Common segment (segment zero).
Bit 6 : Not used.
Bit 7 : Disk I/O error (seek, CRC, or not operable).

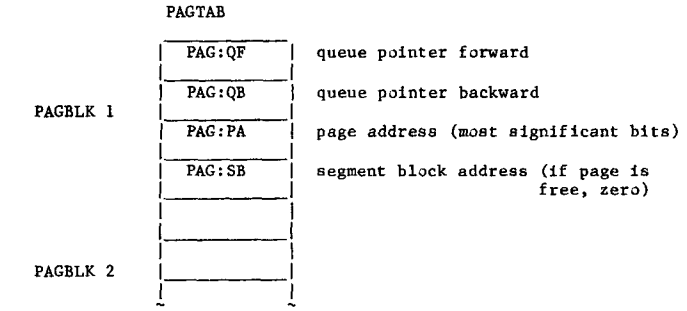
SEG:DS Bits 0 - 7 : Contain zeroes for segment zero;
For other segments bits 1 - 23 contain the disk sector address at which the segment starts (bit 0 is always 0); all zeroes if disk paging is not used. Contains the logical address for segment zero in a disk paging system.

SEG:EL Effective length in bytes of the segment.

SEG:PB Pointer to corresponding page block.

18.6.2 Page Block Table (PAGTAB)

Each page in memory is described in a page block, PAGBLK. All page blocks are contained in the page block table PAGTAB which is built at configuration time by the system loading program, SYSLOD.



PAG:QF Pointer forwards in the page queue, PAGQUE. If the page is not in PAGQUE, this word contains zero.

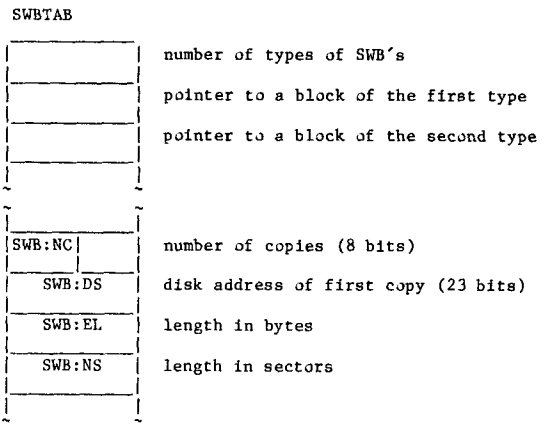
PAG:QB Pointer backwards in the page queue, PAGQUE.

PAG:PA Physical page address in memory; 16 most significant bits.

PAG:SB Pointer to the corresponding segment block. If the page is free, this word contains zero.

18.6.3 Swappable Work Block Table (SWBTAB)

When swappable workblocks are used, the system loading program (SYSLOD) builds tables which have the following layout:-



The swappable workblock type is defined in the application by the 'SWB' declaration.

SWB:NC Number of copies of this type of swappable workblock.

SWB:DS Disk sector address of the first copy of the swappable workblock of this type.

SWB:EL Length of one copy of the swappable workblock of this type in bytes.

SWB:NS Number of sectors occupied by one copy of the swappable workblock of this type.

18.7 I/O HANDLING IN MMU SYSTEMS

The monitor resides in an address space comprising the first 64kB of physical memory. Due to this hardware limitation and the demand for fast response to interrupts, most devices have their own I/O buffer placed in the system area.

At activation or completion of an I/O request, a transfer is made between the user and the device buffer. In most cases this is handled by the I/O initialization and terminating modules (TIO and TENDIO).

The user interface is the same whether the MMU option is used or not, but the device buffers are present only if MMU is implemented.

I/O requests are of several different types, defined by the order and the driver type. When the system itself performs an I/O request, the interface is exactly as in the non-MMU case.

18.7.1 Normal Output Request

This request is used when writing character-by-character, for example on the following devices:-

* GP General Printer.
* TP Teller Printer.
* DY Display.
* TC Cassette.
* LP Line Printer (programmed channel or IOP).
* FD Flexible Disk (programmed channel or IOP).

Each DWT has its own buffer and ECB, and TIO transfers the contents of the user ECB and buffer to the DWT. TENDIO performs the update of the user ECB.

18.7.2 Normal IOP Output Request

This request is used when writing in block mode, for example on:-

* DU Disk.
* MT Magnetic Tape.

TIO assembles the full 18-bit buffer address and the driver transfers it to the IOP. Each DWT has its own ECB to which TIO transfers the user ECB. TENDIO performs the update of the user ECB.

18.7.3 Normal Input Request

This request is used when reading character-wise, for example on:-

* KB Keyboard.
* TC Cassette.
* CR Card Reader.
* FD Floppy Disk (programmed channel or IOP).

Each DWT has its own buffer and ECB, and TIO transfers the contents of the user ECB to the DWT. TENDIO transfers the data from the DWT buffer to the user and updates the user ECB.

18.7.4 Normal IOP Input Request

This request is used when reading in block mode, for example on:-

* DU Disk.
* MT Magnetic Tape.

TIO assembles the full 18-bit buffer address and the driver transfers it to the IOP. Each DWT has its own ECB to which TIO transfers the user ECB. TENDIO performs the update of the user ECB.

18.7.5 Data Communications Input Request

This request is used when reading on:-

* DC Data Communications.

Each DWT has its own ECB to which TIO transfers the user ECB. The driver has a set of buffers and a special monitor routine, DC:MIN, transfers the data from the driver buffer to the user buffer.

18.7.6 Data Communications Output Request

This type of request is used when writing on:-

* DC Data Communications.

The driver keeps one write buffer and, before writing (after poll), a special monitor routine (DC:MOT) is called to transfer the data. Each DWT has its own ECB to which TIO transfers the user ECB. TENDIO performs the update of the user ECB.

18.7.7 Control Requests

This type of request is used when performing a control request (DSCx in CREDIT) on a device. No buffer is needed. Each DWT has its own ECB to which TIO transfers the user ECB. TENDIO performs the update of the user ECB.

19.1 GENERAL

TOSS Data Management functions are implemented as a number of special tasks running on priority level 49.
There are at least two DM tasks per disk driver present in the system.

When a data file has been assigned it will be treated in a similar way to a common I/O device. That is, the file code of the data file is found in CDTAB or TTAB, together with the FWT address. The first part of the FWT has the same layout as a DWT. When a Data Management function is called, control is give to an activation handler (TIODM) within the monitor. This activation handler in its turn activates a DM task for the disk drive.

Each DM request locks all the files it uses and in this way only one user at a time can access a particular file. In other words, it is only when a DM file request is completed that the next DM request for that file can be handled.

19.2 ACTIVATION HANDLER (TIODM)

The activation handler (TIODM) is called in the same way as a driver via an address pointer in the FWT. It checks the order code and can also detect some error conditions, such as exclusive access conflicts and end-of-medium or end-of-file. For sequential and random access methods TIODM also computes the sector number and the relative offset within that sector of the record to be accessed. All the necessary information is stored in the FWT and the FWT address is the only parameter transferred at the final activation of the DM task.

In an indexed file structure only the data file is associated with a file code. Index files are identified internally via a file number. The FWT's of all files in a file structure are locked during a request. External requests to this structure are queued until FWT's are released. Internal DM requests from other DM tasks are not queued.

19.3 DATA MANAGEMENT TASK (DMTASK)

The DM Task contains all the physical I/O routines necessary to perform the various data management functions. Two DM tasks are defined for each disk driver.

DM requests that need data on more than one disk drive cause 'internal DM requests' from one DM task to another.

All I/O requests are made in wait mode. Because there are two DM tasks per driver a request for a drive which already has another request active can start its processing up to the point where it needs the disk, when it will be queued if the drive is still busy. There can thus be an overlap between processing and disk access.

The DM Task makes use of the common sector buffer pool in order to obtain a free sector buffer. If the requested sector is already in memory, this buffer is used and the sector does not need to be read into memory.

Apart from I/O handling, the DM Task also performs the following functions:-

* Moving records to/from user record area from/to sector buffer.
* Updating Exclusive Access buffers and Current Record Number area for the FWT.
* Setting Effective Length, Return Code, and Control Word in the user ECB area.

19.4 MONITOR TABLES

Initially all DM file codes are connected to a special FWT (in TIODM). When a file is opened, a free File Descriptor Block (FWT) is found by File Management, fetched from the FWT pool reserved at SYSGEN time, and the fields in memory are filled in.

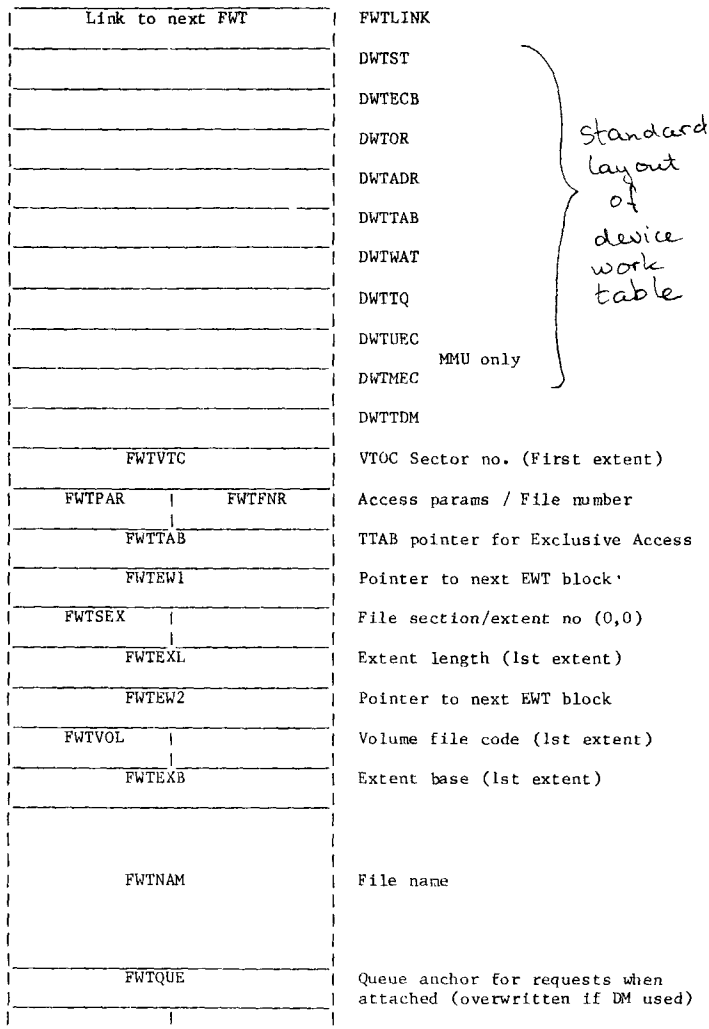
An entry, two words, is made in CDTAB (or TTAB) for the assigned file code. The file codes in CDTAB can be reached by all tasks in the system. Index files do not have file codes.

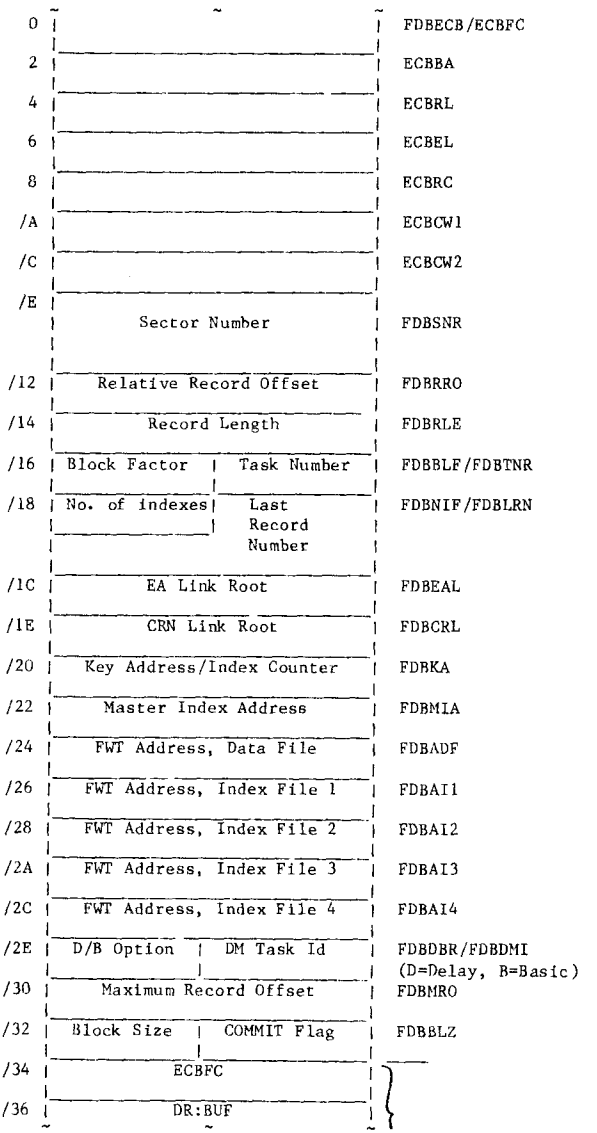
There are two different assign requests and one physical file can therefore be assigned twice with the same file code. The FWT is created at the first request and merely updated if a second request occurs. The CDTAB (or TTAB) is given a separate entry, two words for each request. The first character of the CDTAB/TTAB entry contains an index that identifies the corresponding assign type.

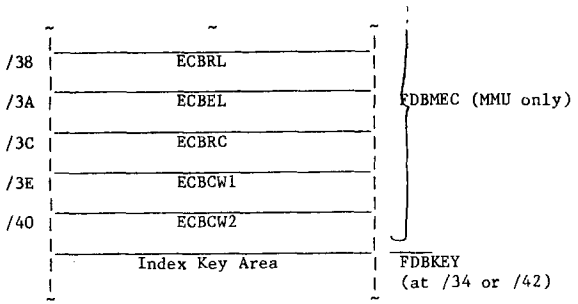
The Monitor Tables are released when a 'Close File' request is issued.

19.4.1 File Work Table

FWT




0
FDBECB/ECBFC
2
ECBBA
4
ECBRL
6
ECBEL
8
ECBRC
/A
ECBCW1
/C
ECBCW2
/E
Sector Number
FDBSNR
/12
Relative Record Offset
FDBRRO
/14
Record Length
FDBRLE
/16
Block Factor
Task Number
FDBBLF/FDBTNR
/18
No. of indexes
Last Record Number
FDBNIF/FDBLRN
/1C
EA Link Root
FDBEAL
/1E
CRN Link Root
FDBCRL
/20
Key Address/Index Counter
FDBKA
/22
Master Index Address
FDBMIA
/24
FWT Address, Data File
FDBADF
/26
FWT Address, Index File 1
FDBAI1
/28
FWT Address, Index File 2
FDBAI2
/2A
FWT Address, Index File 3
FDBAI3
/2C
FWT Address, Index File 4
FDBAI4
/2E
D/B Option
DM Task Id
FDBDBR/FDBDMI
(D=Delay, B=Basic)
/30
Maximum Record Offset
FDBMRO
/32
Block Size
COMMIT Flag
FDBBLZ
/34
ECBFC
/36
DR:BUF



FWTLINK This word links all physical FWT's which are reserved at system generation. Bit 15 (the use bit), if set to 1, indicates that the block is in use.

DWT Standard layout of device work table.

DWTTDM Contains the TTAB address of the calling task.

FWTPAR/FWTFNR /12

NV	B	NC	File Number
0	1	2 7	8 15

NV : 1 indicates 'New volume loaded' has been detected for this file. All requeste for this file code will be rejected withthe appropriate return code. To be able to access this fileagain, all tasks that have opened the file must close itand open it again successfully.

B : 1 indicates that physical write orders must be changed to basic write by File Management.

NC : Number of successful opens performed for file (max 63).

FWTEW1 Extent work table chain

0	EWTLNK	Link to next EWT block
2	EWTVOL	Volume file code
4	EWTEXL	Extent length

0	EWTLNK	Link to next EWT block
2	EWTVOL	Volume file code
4	EWTEXB	Extent base

ECB Layout of normal ECB.
ECBFC is the filecode of the disk (F0, F1, etc.).

FDBSNR Physical sector number; sector content is actually stored in the block buffer.

FDBRRO Relative offset of the record in the block buffer in bytes.

FDBRLE Record length in bytes.

FDBBLF Blocking Factor as specified with the Create file utility.

FDBTNR Number pointing to a word in TCTAB.
The entry contains the pointer to the current task table (TTAB). From TTAB the task identifier can be obtained.

FDBNIF Number of index files (0 when FWT belongs to an index file).

FDBLRN Last record number pointer (starts with one, 3 bytes long).

FDBEAL Exclusive Access link root.

FDBCRL Current Record Number link root.

FDBKA Key address/location in the data record.

FDBMIA Address of master index in memory.

FDBADF FWT address of the data file.

FDBAIn FWT address of index files 1, 2, 3, & 4 respectively.

FDBDBR D - 2 Means delay function included. A write on application level will result in an update in the block buffer. Later the buffer is written to disk. Extra block buffers are required, since one buffer per file is kept until the file is closed.
(Not used for CREDIT).

B - 1 Means basic write, no check is performed.
(Not used for CREDIT).

FDBDMI DM Task Identity.

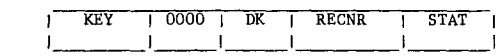
FDBMRO Maximum record offset in block buffer.

FDBBLZ Block Size.

FDBMEC User ECB (MMU only) copied to this area (standard layout).

FDBKEY Contains the record key for indexed access.
Size is defined at SYSGEN time.

19.4.2 Layout of a Record in an Index File



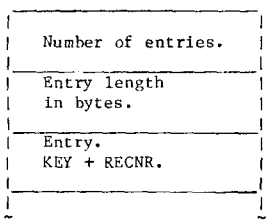
KEY Key = A string of 1 to N bytes, left adjusted and padded with blanks. It is used as a record identification.

DK Duplicate key.
One byte which contains a binary value specifying the minimum number of leading bytes in a key which are identical with the next key in the index file. When the next key is identical, this equals the key length.

RECNR Record number.
Three bytes containing the logical record number of the record in the data file, with corresponding key value.

STAT Status.
One byte indicating whether the record is used or free.
When used the value is X'FF'; when free it is X'00'.

19.4.3 Layout of a Master Index in Memory



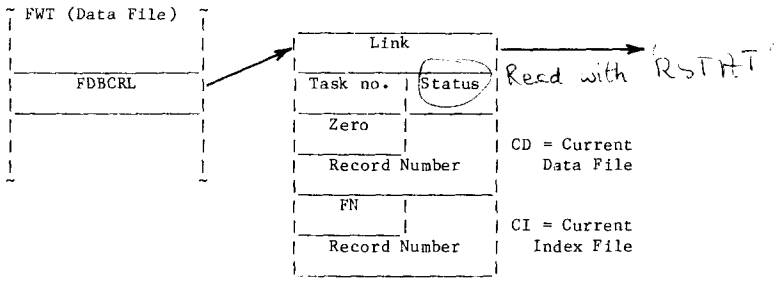
Number of Entries. One word containing a number indicating the entries in the master index.

Entry Length. One word containing a number indicating the number of bytes used per entry for the key and record number.

Entry. Each entry occupies 1-N bytes for the key, plus three bytes for the record number (Record number in index file).
The last entry will contain all ones in the key field.
When only one entry is present, the key value will be all ones, and the record number is set to one.

19.4.4 Current Record Number Handling

FN = File Number of Index File.
Task number is used as an index to TCTAB.



One CRN buffer (6 words) is allocated for each user task per file structure (data file with/without index file). Other data management requests from the same task to the same structure will use the same CRN buffer. The buffer is released after closing the file. When more than one task is using the same file structure, the CRN buffers are linked via the first word in the buffer.

LINK This word contains a pointer to next currency buffer. Zero indicates the end of the chain.

Task no. Contains in one byte a number which is a byte displacement from the second word in TCTAB (ie. the word following the length word) to a word containing the address of the TTAB for which this current record number is valid.

Record nr. Three bytes containing the record number in the data file.

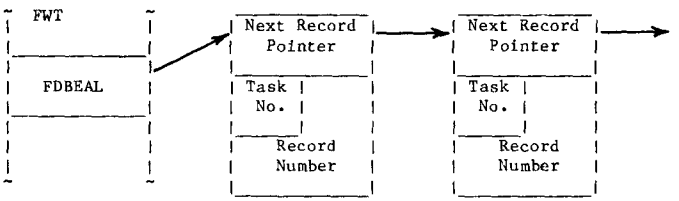
FN File number of the index file.

Record nr. Three bytes containing the record number in the index file.

19.4.5 Exclusive Access

Since files may be in simultaneous use by more than one task, it is important that these tasks are prevented from updating the same record at the same time, or otherwise interfering with each other. Therefore it is possible for a task to request Exclusive Access (EA) to a record or records in one or more files.

One EA buffer (3 words) is allocated for each record of the data file set under exclusive access. The EA buffers are linked via the first word in the buffer. The word FDBEAL (the EA Link Root) in the file descriptor block points to these chained EA buffers.



Whether or not EA is required is specified by the user at the time of issuing a Read Request, and EA is relinquished via Release Exclusive Access, which does not involve any physical I/O. EA is not applicable to Index files.

If an EA request for a record cannot be honoured because that record is already being accessed by a task having EA, then a Record Protected status reply will be returned to the requesting task. This is done in order to avoid deadlock situations. A deadlock may still occur if the requesting task, having received the Record Protected status reply, does not release EA on any other records to which it already has exclusive access.

Next Record Pointer. — Pointer to the next EA block. When zero, the end of the chain has been reached.

Task no. — Contains in one byte a number which is a byte displacement from the second word in TCTAB (ie. the word following the length word) to a word containing the address of the TTAB for which this record is set under exclusive access.

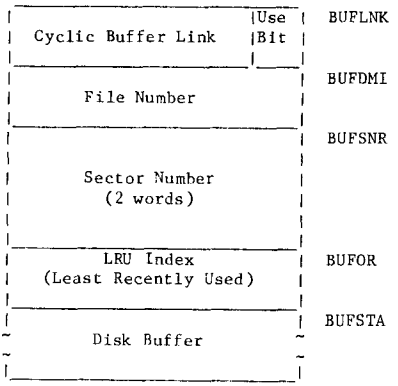
Record Number — Three bytes containing the record numbers in the data file which are under exclusive access.

19.5 BUFFER MANAGEMENT

DM uses a common block buffer pool for all files and indexes. A 'Least Recently Used' algorithm is used to select buffers.

In the DMBUF module, data areas are allocated for these buffers, and routines to get and release a DM buffer are supplied.

By setting a constant (QNBUF), buffers can be allocated in DMBUF by SYSGEN, (2<QNBUF<16). The buffers will be linked together cyclically. Each buffer has a buffer header where the contents of the buffer are described.



The words BUFDMI and BUFSNR define one unique sector. The LRU index is kept in the buffer header to let buffers remain in memory as long as possible in order to minimize physical I/O. The word CURBUF contains a pointer to the current buffer.

BUFLNK This word contains a pointer to the next buffer in the cyclic chain. Bit 15 is used to indicate whether this buffer is in use or not.
Bit 15: 1 = buffer is in use; 0 = buffer is free.

BUFDMI File Number.

BUFSNR Contains the physical sector number, the contents of which are stored in the buffer.

BUFOR This index is incremented by one when a sector is found to be not present in memory. When the buffer is about to be used the index is reset. If a buffer has to be overwritten, the one with the highest LRU index is selected.

BUFSTA Start of the buffer area in which the sector contents is stored.

19.5.1 Get and Release a DM Buffer

As input parameter to the get buffer routine, a unique sector must be defined in registers A1 and A2 in the same way as in BUFDMI and BUFSNR.

If this sector's contents are already in a memory buffer, the buffer address is put in A3, the use bit in BUFLNK set, and a normal return taken.

If the sector searched for is not located in memory the oldest unused buffer is selected via the LRU index.
When this buffer is found, its use bit is set, and a skip return taken (i.e. return to normal return address + 2).

When no unused buffer is available, the system lights SOP lamps 10 and 11 and HALT.

The release buffer routine will reset the use bit in BUFLNK of the buffer address given in A3.

19.6 QUEUEING

There are four points where a queue may arise in Data Management Handling.

19.6.1 File Request Queue

Only one request at a time is handled for a particular file structure. If a file is busy the request is queued in the monitor before the activation handler (TIODM) is called.
Queueing is done using the principle 'First-In-First-Out' per level of calling task (the queue pointer is stored in the FWT, in DWTTQ.
The file request queue is dissolved when the DM request in progress is completed (TENDIO).

19.6.2 Pending Queue for DM Task

When a DM task is activated it may be busy with a job for another file. In that case the activation is put in the ordinary task pending queue (queue pointer is stored in TTAB for the task). The pending queue is dissolved upon an EXIT request for the DM task.

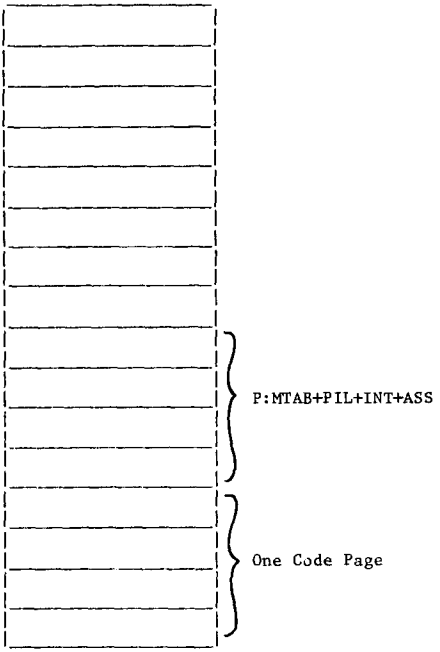
19.6.3 Device Queue for Physical I/O

If the disk drive is busy when a call for physical I/O is made, this request is queued in the monitor (queue pointer is stored in the DWT (DWTTQ) of the data management task. The device queue is dissolved when the running I/O request is completed. The device queue will never contain more than one element, since another DM request will be queued already in the DM task pending queue.

19.6.4 Open Queue

Open and Close orders are executed without overlap. This is accomplished through the DM task pending queue. Only one DM task is allowed to execute these orders.

MMU table (window layout).



The base module containing the address of the interpreter must be a part of all task windows. Moreover the number of entries necessary to point to one page in core must be reserved. The rest of the MMU table entries may be used for the part of the data division used by the task.

The data window of a task may be divided into three main parts.

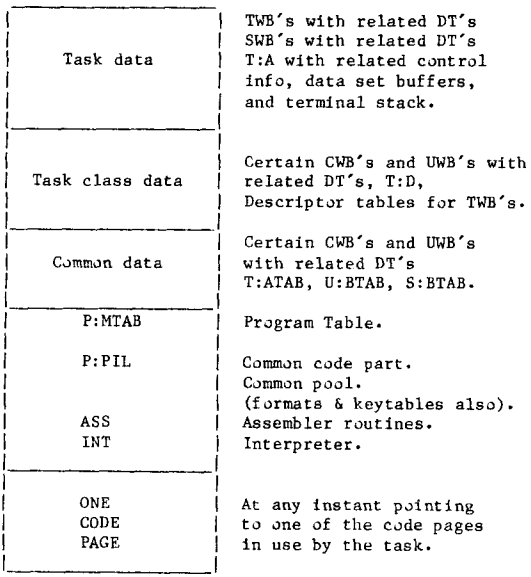
Common Data
Common work blocks used by more than one terminal class belong to this part, together with their descriptor tables. Also, user work blocks accessible from more than one terminal class are present here, together with their descriptor tables.
Certain control tables of common type, such as T:ATAB (task control table) and U:BTAB (user work block control table), belong to this group as well.

Task Class Data
Common and user work blocks (and their descriptor tables) used by only one terminal class are present in this group. The terminal class descriptor table T:D is also present.

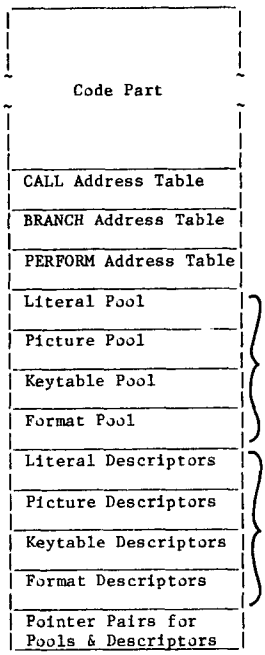
Task Data
Terminal work blocks are allocated to this part of the window as well as the task table (T:A) with related terminal stack, data set buffers and control information.

This means that the task window may differ for each task, which implies that in the total application more then 64kB may be present in the data division. In memory, code and data are allocated consecutively and no memory space is wasted.

Logical Task Window

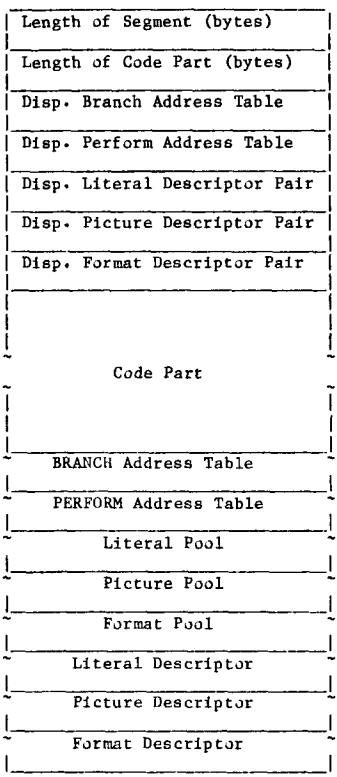


Common Code Part, Segment Zero.



Code Part	Consists of interpretive CREDIT code.
CALL Address Table	The first word contains a value indicating the number of entries in the table; the following words contain the addresses of the Assembler routines.
BRANCH Address Table	The first word contains a value indicating the number of entries in the table. The following words contain the addresses.
PERFORM Address Table	The first word contains a value indicating the number of entries in the table. The following words contain the addresses.
Literal Pool Picture Pool Keytable Pool Format Pool	Each pool contains literals, pictures, keytables, and formats respectively.
Literal Descriptors	A pointer pair; the first points to the descriptor table, the second to the literal pool.
Picture Descriptors	A pointer pair; the first points to the descriptor table, the second to the picture pool.
Keytable Descriptors	A pointer pair; the first points to the descriptor table, the second to the keytable pool.
Format Descriptors	A pointer pair; the first points to the descriptor table, the second to the format pool.

Segment X


Length of Segment (bytes)
Length of Code Part (bytes)
Disp. Branch Address Table
Disp. Perform Address Table
Disp. Literal Descriptor Pair
Disp. Picture Descriptor Pair
Disp. Format Descriptor Pair
Code Part
BRANCH Address Table
PERFORM Address Table
Literal Pool
Picture Pool
Format Pool
Literal Descriptor
Picture Descriptor
Format Descriptor

Length of Segment.	One word containing a value indicating the total size of the segment in bytes.
Length of Code Part.	One word containing a value indicating the total size of the interpretive code part.
Pointer To Branch Address Table.	One word containing the relative address within the segment of the branch address table.
Pointer To Literal Descriptor Pair.	One word containing the relative address within the segment of the literal descriptor.
Pointer To Picture Descriptor Pair.	One word containing the relative address within the segment of the picture descriptor.
Pointer To Format Descriptor Pair.	One word containing the relative address within the segment of the format descriptor.
Code Part.	Actual interpretive code part.
BRANCH & PERFORM Address Table.	The actual branch and perform table are stored here.
Literal, Picture, and Format Pools.	The pools containing the actual literals, pictures, and formats used in this segment.
Literal, Picture, and Format Descriptors.	Two words for each type. The first word contains the rel. address of the descriptor table and the second word the relative address of the pool where the actual values are stored.

24.1 INTRODUCTION

SYSLOD is a system software module which is linked to the TOSS Monitor. It takes care of application loading and monitor and application configuration on the whole range of PTS systems; PTS6805 and PTS6810 with or without overlay, and PTS6820 with MMU and/or overlay.

SYSLOD performs five functions:-

* Loading the application load module.
* Reading the configuration file.
* Monitor configuration.
* Application configuration.
* Starting the System.

24.2 I/O REQUIREMENTS

SYSLOD has its own I/O routines and is independent of drivers contained in the Monitor. Loading can be done from fixed, cartridge or flexible disk, or from cassette. Monitor, application load module, and configuration file must all be loaded from the same input medium type, but need not be on the same volume.

24.3 LOADING PROCEDURE

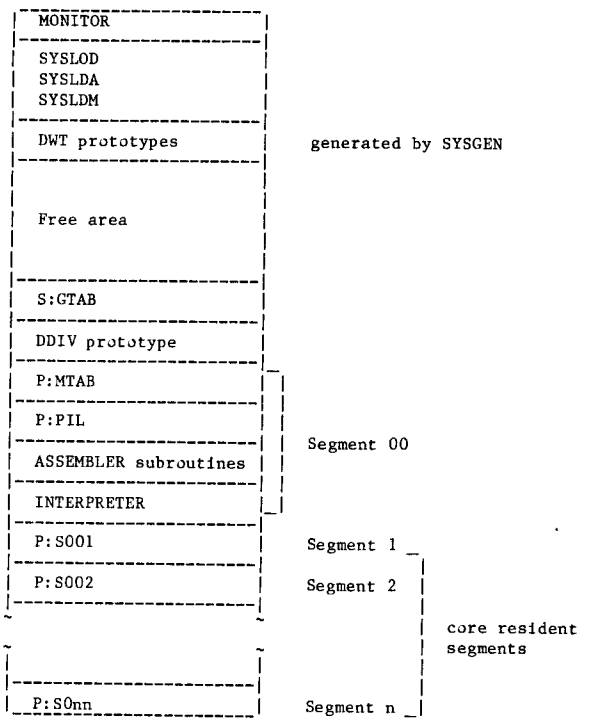
The System loading program consists of three modules. When the Monitor has been loaded by the initial program loader, control is passed to SYSLOD. SYSLOD loads the application and reads the configuration file, SYSLDM then performs monitor configuration and SYSLDA performs application configuration. SYSLDA will queue all the tasks in the dispatcher queue. After that, all the drivers are initialized and control is passed to the DEBUGGER or the Interpreter.

24.4 APPLICATION LOADING

First the application load module as created by the Linkage Editor (LKE) is loaded into core. If loading is from disk and the program is segmented, only segment zero and the other core resident segments are loaded. If loading is from a sequential access medium or if the program is not segmented the entire application is loaded.

The application, segment 0, and the memory-resident segments are placed as high as possible in memory.

24.4.1 Memory Layout after Application Loading

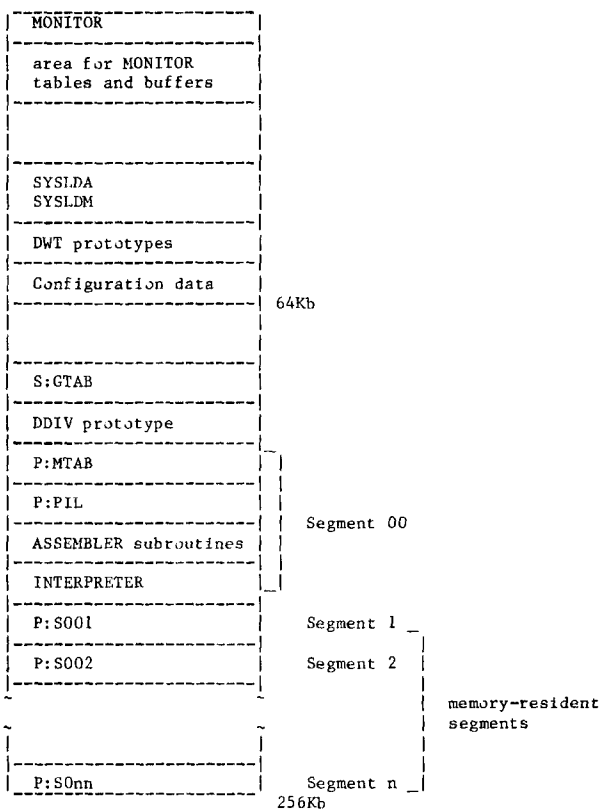


24.5 READING THE CONFIGURATION FILE

Now the configuration file is read into memory and, in a 64Kb machine, placed immediately before the application. With a 256Kb memory, it is placed at address X'FFFE' and lower.

The part of the SYSLOD program which is still needed, and the DWT prototypes, are then placed immediately before the configuration data to make room for monitor tables and buffers.

24.5.1 Resulting Memory Layout



24.6 MONITOR CONFIGURATION

SYSLDM now performs Monitor configuration. Input data for Monitor configuration is the configuration file, the DWT prototypes generated by SYSGEN, and the data arrays SCLASS and TCLASS with information on the special device classes and terminal device classes as specified during the SYSGEN dialogue.
The tables MONTAB and SYSTAB hold the addresses of the run-time tables (configuration tables).

24.6.1 Building Monitor Tables

The following Monitor tables are built:-

* Task Control Table, TCTAB, with pointers to all the TTABs.
* Task Tables, TTAB, with specific information for every task.
* Device Work Tables, DWT, for the terminal devices, special devices and common devices.
* Interrupt Tables for the terminal devices according to the line connections specified in the configuration data.

For a system with a Memory Management Unit (MMU) the TTABs are extended with 16 words to contain the logical MMU addresses connected with this task.
For a segmented application the segment table SEGTAB is built, holding status information, disk sector address, length and load address in memory of each segment.
The corresponding page table PAGTAB is generated which contains the page queue pointers, physical page addresses and the segment block address if the page is used.

24.6.2 Workblocks

If swappable workblocks are defined in the configuration file, each SWB is described in a block SWBBLK which contains information about number of copies of the block, disk address, number of sectors occupied by each copy, and the block length in bytes. Table SWBTAB with pointers to all SWBBLK's is also built.

24.6.3 Buffers

Buffer areas are reserved for data communications, data management, and other devices or functions as required. In a system with MMU, extra I/O buffers in the System area are allocated. These areas are still used by SYSLDM and SYSLDA during configuration. SYSLDM ends by generating the Monitor blocks.

S:GTAB and the DWT prototypes are now no longer needed and may be overwritten. The area occupied by SYSLDM is then released and can be used for application configuration.

24.7 APPLICATION CONFIGURATION

Application configuration is performed by SYSLDA. An auxiliary table with the number of tasks per taskclass is formed to build the terminal control area table T:ATAB. For user workblocks and swappable workblocks, tables are set up according to the number of blocks specified in the configuration data.
If the application contains disk resident segments, memory pages are reserved as read only areas to contain these segments.

24.7.1 Generating the Data Division

SYSLDA and the data division prototype are relocated to make room for the real data division, which can now be built from the DDIV prototype and the configuration data. Workblocks with their descriptor blocks are generated.

Pointers to T:ATAB, U:BTAB, and S:BTAB are updated in P:MTAB. For a system with MMU, task-connected MMU addresses are filled in in TTAB.
If the application contains disk resident segments, memory pages are reserved as read only areas to contain these segments.

Application data can be divided into three types and building of the DDIV is accordingly done in three steps:-

1. The part common to all tasks is generated, the CWB's and UWB's that are used by more than one taskclass.

2. The task class data, CWB's, UWB's, and SWB's that are used within a taskclass.

3. The task local data is generated, the Task Control Area, TCA, the terminal stack, dataset buffers, and TWB's and SWB's.

24.7.2 Generating the Tasks

One task of each taskclass is now completely generated.
Task control areas T:A are copied as many times as indicated in the auxiliary T:ATAB table.
The task identifiers TID are updated in the T:A's and saved in the T:AID table.
Now every task is put in the dispatcher queue and SYSLDA gives control to the module PFINIT to initialize the drivers. A branch to the dispatcher is then performed to schedule the first task.

24.8 ALLOCATION RULES

In a system without MMU, configuration simply consists of making as many copies of each type of task and workblock as specified in the configuration data. If the application is also not segmented, application buffers are allocated following the monitor tables and buffers upwards, over the area still used by SYSLDA.

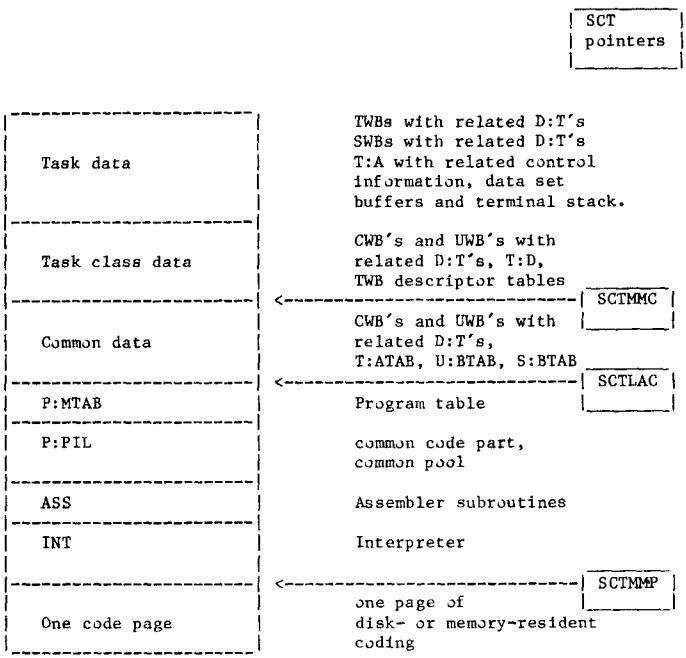
For a segmented application or a system with MMU, application buffers are allocated from the highest free address in the free area downwards. This leaves space for memory pages for the disk resident segments between the application buffers and the monitor tables and buffers.

24.8.1 Task Window

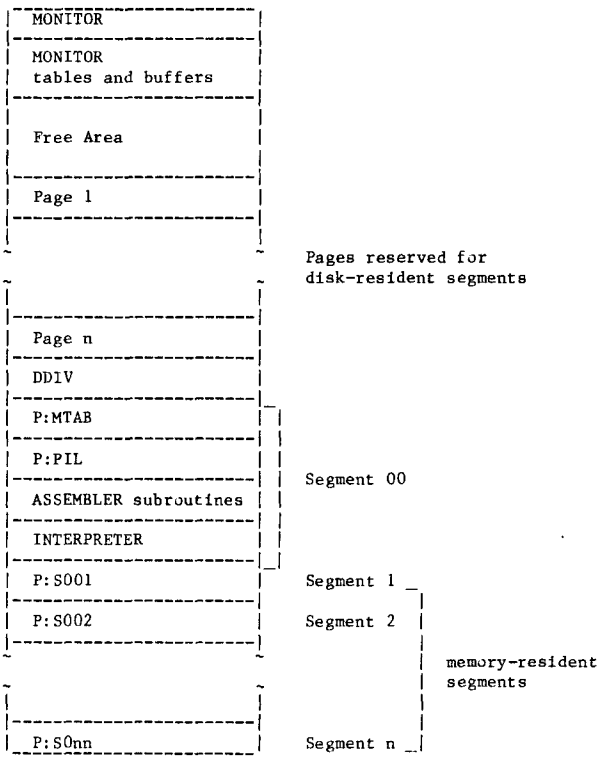
For a system with MMU, configuration is fairly complicated. The MMU table with 16 entries, each needed to address 4K bytes, provides the task with a logical task window of 64K bytes. Segment zero, with the common part of the data division and the task-class common data, P:MTAB, P:PIL, the assembler subroutines and the interpreter, must be a part of all task windows.

This occupies the MMU table to a high degree already. The number of entries necessary to address one page in core must also be reserved. The rest of the MMU table entries may be used for the part of the data division used by the task.

24.8.2 Resulting Logical Task Window

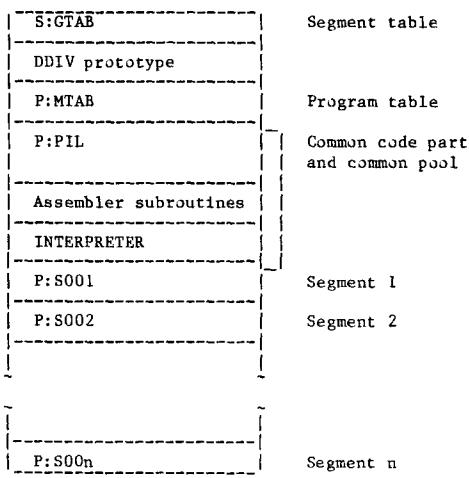


24.9 MEMORY LAYOUT AT END OF APPLICATION CONFIGURATION



24.10 CREDIT APPLICATION IN SECONDARY MEMORY

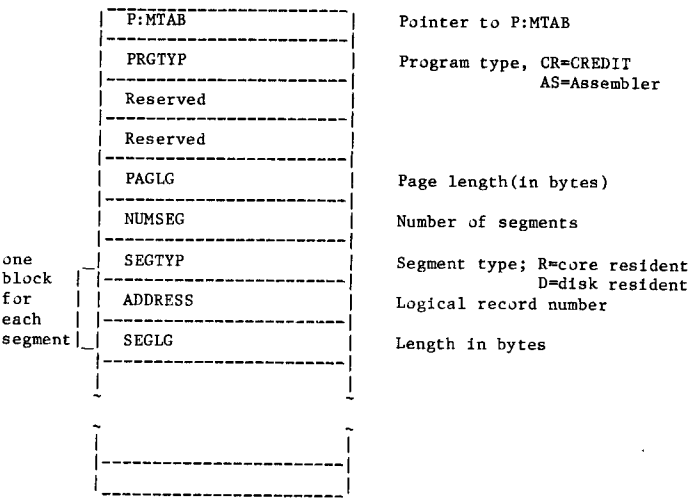
The Application Load Module created by LKE.



24.10.1 Segment Table (S:GTAB)

S:GTAB is the segment table used by SYSLDM and afterwards overwritten.

Each item is two bytes in length.



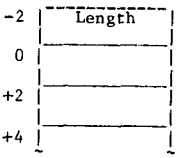
24.10.2 Program Table (P:MTAB)

P:MTAB is the program table used by the interpreter. It contains the logical addresses of the tables that are set up for the application program. Each item is two bytes in length.

Byte		Pointers to:-
0	T:ATAB	Task control area table
2	U:BTAB	User work block control table
4	I:NTPA	System start address (in interpreter)
6	P:BAS	Start of base module (P:PIL)
8	T:BAT	Branch address table
/A	T:CAT	Call address table
/C	T:PAT	Perform address table
/E	T:LIT	Literal pool table
/10		Highest index +1 in T:LIT
/12	T:KEY	Keytable pool table
/14		Highest index +1 in T:KEY
/16	T:PIC	Picture pool table
/18		Highest index +1 in T:PIC
/1A	T:FMT	Format pool table
/1C		Highest index +1 in T:FMT
/1E	P:END	End of base module (P:PIL)
/20	T:AID	Task ID table (for CREDIT debugger)
/22	OPTION	SYSTEM option (SCOPT)
/24	LITADR	Literal addressing mode
/26	ADRMOD	Data addressing mode
/28	S:BTAB	Swappable workblock control table
/2A	5 words reserved for Assembler Debugger.	

T:ATAB Task control area table (contains logical addresses).

Layout of table T:ATAB.



Length is the table length in bytes, including this word. The rest of the table contains pointers to task control areas, ie. T:Axxy's.

U:BTAB User workblock control table. Contains pointer to U:BTAB.

I:NTPA System start address.
Contains start address, from the interpreter.

P:BAS Start of base module.
Start address of the interpretive code module P:PIL.

T:BAT Pointer to branch address table.

T:CAT Pointer to call table.

T:PAT Pointer to perform table.

T:LIT Literal pool table.
This word contains a pointer to another pointer pair, in segment zero, which consists of one pointer to the data descriptor table for literal constants, the second pointer points to the actual pool.
The descriptor table consists of two words for each literal constant in the pool.
The layout of the descriptor table is the same as for data item (see D:zz0).

Highest index in T:LIT.
A value which indicates the highest index in the descriptor table of segment zero.

T:KEY Keytable pool table.
This word contains a pointer to another pointer pair in segment zero, which consists of one pointer to the data descriptor table for keytables, the second pointer points to the actual pool.
The descriptor table consists of two words for every keytable in the pool.

Highest index in T:KEY.
A value which indicates the highest index in the descriptor table of segment zero.

T:PIC This word contains a pointer to another pointer pair in segment zero, which consists of one pointer to the data descriptor table for the pictures, the second pointer points to the actual pool.
The descriptor table consists of two words for each picture definition. The layout of the descriptor table is the same as for data items (see D.zz0).

Highest index in T:PIC.
A value which indicates the highest index in the descriptor table of segment zero.

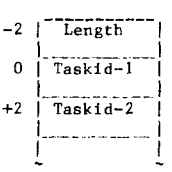
T:FMT Format pool table.
This word contains a pointer to another pointer pair in segment zero, which consists of one pointer to the data descriptor table for formats, the second pointer points to the actual pool. The descriptor table consists of two words for each format list definition. The layout of the descriptor table is the same as for data items (sww D:zz0).

Highest index in T:FMT.
A value which indicates the highest index in the descriptor table of segment zero.

P:END End of base module.
End address of the interpretive code module P:PIL.

T:AID Task identification table.
This word contains a pointer to a table in which all task identifiers are stored; only used by the CREDIT debugger.

Layout of T:AID.



Length is the table length in bytes including this word.
Following words contain the id's of the user tasks.

OPTION System Options.
A value indicating the system options.
0 - Standard (No MMU, no disk paging).
1 - MMU system.
2 - Disk paging system.
3 - MMU and disk paging system.

LITADR Literal addressing mode.
Two bytes are used to indicate the addressing mode.
X'1111' means one byte addressing for literal constants, keytables, pictures, and formats.
The value is derived from the LITADR option in CREDIT.

ADRMOD Addressing mode.
A value 1 or 2 specifying which addressing mode is used.
This is derived from the ADRMOD option in CREDIT.

S:BTAB Swappable Work Block Table.
Pointer to the swappable work block table.

The use of the SYSGEN utility is described in the DOS 6800 System Software Reference Manual (M11).

The update file (file code /CA) which is created by running the SYSGEN utility contains commands for updating source modules present in TOSSWORK. The updates are related to the questions answered.

To see which adaptations are made for a driver, the update file can be listed and the corresponding updates searched for.

Assigning file code /E0 to the update file results in execution of the commands, and all source modules of the generated monitor are kept in the current user (userid). The monitor will contain all initial task tables (TTAB), CDTAB's, etc. which will be input for the system loading program SYSLOD.

The terminal device classes, created with SYSGEN, are stored in the module TCLASS for terminal devices, and in module SCLASS for special devices, for use by SYSLOD

These modules contain two words per device.
The first word contains an index value in the most significant byte, the least significant byte contains the filecode.
The second word contains a pointer to the device work table.
SYSLOD will use these classes to configure other tasks in the monitor.

By setting the conditional assembly parameter DIFPRI to 'EQU 1' in TDISP, it is possible to have tasks running at different priorities in a disk paging system.

/CB (on disk): printout of the conversation.

26.1 SOP LAMPS

When a system is halted due to a serious system error, an error code is displayed on the SOP lamps. This code should be noted and can be used later as an aid in analyzing the dump. The following lamps may be lit (lamp 1 is leftmost on the SOP panel):-

SOP LAMP NUMBERS (x = lit)

7	8	9	10	11	
			x	x	No currency buffer available.
		x		x	Illegal interrupt.
		x	x	x	Stack overflow.
	x			x	Instruction not accepted. *
	x		x	x	No blocks available.
	x	x		x	Invalid instruction (trap).
	x	x	x	x	Requested LKM processor not in monitor.
x				x	Data management (SYSGEN) error.

* SST, OTR, or INR not accepted due to a hardware error.

However, in some cases the RUN lamp is off and the SOP lamps give no indication of the cause of the problem. If this occurs, it is useful to take the approach outlined in section 26.3.

8-9-11 Timque, Curr. buffers, Lst-link, excl. access, pending queue (loop!).

26.2 ACTIONS BEFORE TAKING A DUMP OF MEMORY

Before executing the dump utility DMPGEF or DMPGEN, created under DOS6800, it is necessary to save some memory words because these utilities overwrite the first part of memory.

When dumping on flexible disk, the utility DMPGEF will overwrite memory locations X'0000' to X'021A' if the flexible disk drive is connected to an IOP. If the drive uses the programmed channel, memory locations X'0000' to X'027E' are overwritten. The DMPGEN utility, used to dump on cassette, will overwrite memory locations X'0000' to X'00FE'.

In order to save memory words it is necessary to connect a full panel, which is used to select the relevant memory locations. The contents of these selected locations must be noted as displayed on the lamps of the full panel.

It is useful to save the following words from SYSTAB (system table):-

X'00A0' - Start of free area.
X'00A2'
X'00A4' - End of free area.
X'00A6'
X'00B0' - PAGTAB address. B6
~~X'00B2' - SEGTAB address.~~
X'00B4' - SWBTAB address. B8
X'00C8' - TCTAB address. B2
X'00CA' - CDTAB address. B4
X'0100' } - System Stack, addressed via A15.
X'01FE' }

The PSW may be saved via the console panel.

26.3 MEMORY DUMP

Various situations can cause the system to come to a halt. An error indication may be given on the SOP lamps, but sometimes this is not so, when the RUN lamp is off.

It is important to know whether the problem concerns only one work station, or if all are involved, and to localise the problem as much as possible.

26.3.1 Problem with a Single Workstation

1. Find the pointer SCTTCT at /C8 (B2) in the System Control Table, which addresses the TCTAB.

2. From TCTAB can be found the address of the TTAB corresponding to the problem workstation.

3. Note the dispatch address and the contents of registers A7, A8, A13, and A14 which are stored in the save area of the TTAB.

A8 points to the ECB. Bit 0 in the ECB is used as an indication of I/O operation status. If it is 1, then the I/O has completed and the dispatch address points to the next instruction to be executed. If it is 0, I/O is not finished and the dispatch address is the last one used.

The DWT for the device can be found by scanning the TTAB device dependent part for the filecode contained in the ECB.

For CREDIT, A13 points to the Task Control Area, T:Axx0, which contains the CIA. CIA is the logical address of the last CREDIT instruction which was executed (i.e. Current Instruction Address).

26.3.2 Problem Workstation Unknown

1. Find TCTAB via the SCT as in step 1 above.

2. From TCTAB, locate all TTAB's and thus the status of each task.

3. Check I/O status for each task as in step 3 above, and similarly establish the halt point instructions from the CIA.

26.3.3 Problem with Multiple Workstations

Check the contents of the word PRUN. If not zero, establish I/O status and CIA's as above. If it is zero, it is likely that all tasks are waiting for completion of I/O.

If it appears that the system is hanging up because of a Data Management task, you will probably need technical assistance.

26.3.4 Reading the Dump

To aid in interpreting a memory dump (printed using PRDUMP), it is useful to have available the following:-

* The monitor source listings.

* The linkage editor map of the monitor (output from SYSGEN).

* The linkage editor map of the application.

Note that the LKE map of the monitor is not valid for modules SYSLOD, SYSLDA, SYSLDM, and subsequent prototype module locations, since they are not used after system loading, or are reallocated then.

With MMU systems, addressing in User Mode is done via the MMU, and this must be allowed for when reading the dump.

Some tables and keywords are important to find out the state of the system when the dump was made. See next page for details.

Occurs in linklist monitor

Keywords

PRUN This word contains the TTAB address for the running task. When zero the system is in the idle loop, or all tasks are waiting for completion of I/O.

INTSAV This word contains the last interrupt. It is the contents of the instruction counter (P) from the driver which received the interrupt.

DISQUE The dispatcher queue pointer, pointing to the first task (TTAB) in the queue. When zero, the dispatcher queue is empty.

TIMQUE Pointer to the timer queue. When zero, no timers are set.

PAGQUE In a paging system, two words pointing to the beginning and end of the page queue (inactive pages, but present in memory).

FREQUE A pointer to the first free block (3 words). When zero, blockpool is empty, no blocks available.

Tables

T:AxxO Task control area for a CREDIT task.

TCTAB Contains pointers to all TTAB's.

TTAB This table gives the current task status. In the save area is stored the contents of register A13, which points to T:AxxO. A7 contains the order code, A8 the ECB address, and A0 the dispatch address.

DWT Includes current device status.

CWT This table includes the current control unit status.

26.4 ADDRESS TRANSLATION VIA MMU

In systems with more than 64kB of memory, addresses are translated via the MMU when the system is executing instructions in User mode.

The memory is divided into four 64kB sections, which are addressable when using the CREDIT debugger by using letters as follows:-

'S' - First 64kB (the system area).
'X' - Second 64kB.
'Y' - Third 64kB.
'Z' - Fourth 64kB.

A 16-bit address may be translated for user mode in the following way when reading a dump (see also figure 26.1 on the next page):-

Consider the address X'B68A' for example.

The most significant tetrad (in this case 'B') of the 16-bit address specifies the entry in the MMU table (on top of TTAB). The contents of entry B in the MMU table is in this case X'9000'.

The two most significant bits give the memory section, 2, and the next four bits form the most significant tetrad of the 16-bit displacement in the addressed section. The three other, less significant tetrads of the original address (X'68A') must now be added, resulting in a displacement of X'468A' from the start of section 2.

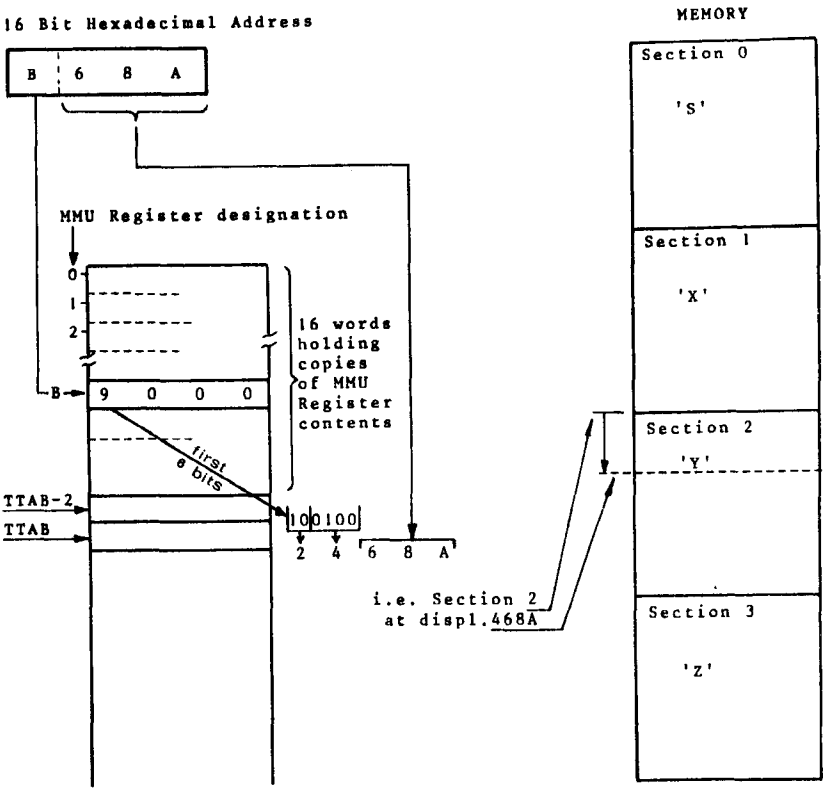


Figure 26.1. MMU Address Translation.

CHAPTER 27

REFERENCE

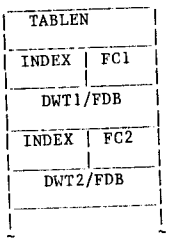
```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                                       *
*    Under the right hand box at the top of each reference page is      *
*                                                                       *
*    printed the number of the section in the appropriate chapter       *
*                                                                       *
*    where the relevant subject is discussed.                           *
*                                                                       *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

CDTAB

COMMON DEVICE TABLE

CDTAB

15.6



CRN

CURRENT RECORD NUMBER POINTER

CRN

19.4.4

FDB (Data File)
FDBCRL

Link	
Task no.	Status
Zero	
Record Number	
FN	
Record Number	

CD = Current Data File

CI = Current Index File

CWTLTy — CHANNEL WORK TABLE LOCAL TERMINALS — CWTLTy

13.3.5

CWTLTy

y = 1-4

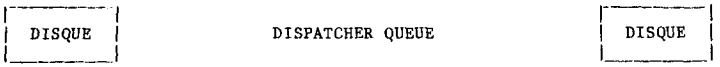
CWTLDW	Last output DWT
CWTLOW	Last output word
CWTITA	Address to interrupt table
CWTINR	INR - instruction
CWTOTR	OTR - instruction
CWTCIS	CIO - start
CWTRTC	Retransmission counter
CWTEQ	Queue first terminal on channel
CWTADD	NAK - accumulator
	Retransmission fault accumulator
	Undefined interrupt accumulator
ACKTIM	Pointer time-out accumulator

DAB — DRIVER ADDRESS BLOCK — DAB

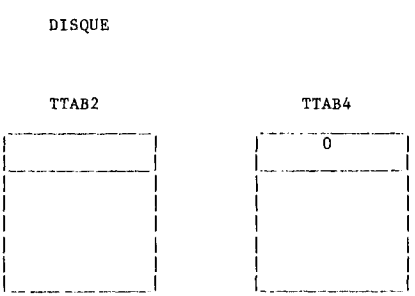
15.8

DAB

byte		
		only with MMU
-6	KEYLEN	key table length
-4	BUFLEN	buffer length
-2	DEVIND	device index
0	ACTADR	activation address
2	ABTADR	abort address
4	INTADR	interrupt address
6	RECADR	recovery address
8	ECHADR	echo address



14.2

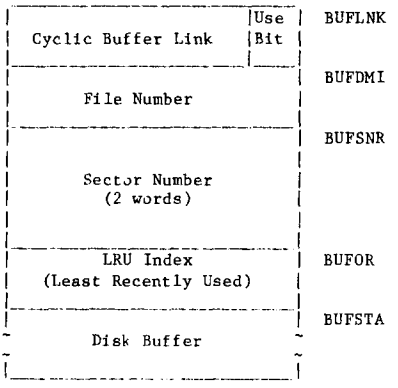


DMBUF

DATA MANAGEMENT BUFFER

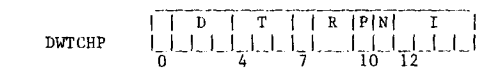
DMBUF

19.5



DRDI01	SIGNAL DISPLAYS AND KEYBOARD LAMPS	DRDI01

Display characteristics are defined in the first word of the DWT, DWTCHP, which is layed out as follows:-



The bits in this word contain the following:-

bit 0 : Set to 1.

D (bits 1 to 3) : Device address on selector unit.

T (bits 4 to 6) : Terminal address on CHLT OR CHRT.

bit 7 : Set to zero.

R (bits 8 & 9) : Reserved.

P (bit 10) : Set to 1 for PTS 6241.

N (bit 11) : 0 for PTS 6241, 6242, or 6233.
1 for PTS 6232, 6234, 6236, 6271, 6272, or 6331.

1 for output to BCR.

I (bits 12 to 15) : Channel unit Index.
/0 = First CHLT.
/2 = Second CHLT.
/4 = First CHRT.
/6 = Second CHRT.

Another word in the DWT is required by the driver. It is DWTSD at displacement .
Display information is preset into this word as follows:-

/2010 for PTS 6233, 6241, or 6242.
/0000 for PTS 6232, 6236, 6261, 6271, 6271, or 6331.

For a PTS 6241 two further words are required, following DWTSD, as follows

DWTSD : /2010
/3F4F
/5F6F

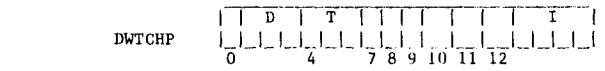
DRDI01 (continued) DRDI01

If order /39 is included in the driver, another two extra words are required in the DWT, following the above mentioned words, as follows:-

```
DWTSD  :  /0000
          /3F4F
          /5F6F  | if order /07, else zeroes.
          /0000
          /0000
```

DRDY01	VIDEO & PLASMA DISPLAYS	DRDY01

Display characteristics are defined in the first word of the DWT, DWTCHP, which is layed out as follows:-



The bits in this word contain the following:-

Bit 0 : Zero.
D (Bits 1-3) : Device address on selector unit.
T (Bits 4-6) : Terminal address on CHLT/CHRT.
Bit 7 : Zero.
Bit 8 : Low intensity mode (6344 only).
Bit 9 : Underline mode (6344 only).
Bit 10 : Set to 1 in PTS 6385 or 6386, else 0.
Bit 11 : Set to 0 if PTS 6344 or 6386.
Set to 1 if PTS 6351 or 6342.
I (Bits 12-15) : Index indicating channel unit.
0 = first CHLT
2 = second CHLT
4 = first channel, first CHRT
6 = second channel, first CHRT
8 = first channel, second CHRT
A = second channel, second CHRT.

The number of display columns is contained in the byte DWTPOS at displacement on the DWT.

The number of display lines is contained in the byte DWTLIN at displacement on the DWT.

DRGP01 GENERAL TERMINAL PRINTER DRGP01

In one word of each DWT, printer characteristics are defined. This word has the following format

Bit 0 : Zero
Bit 1-3 : Device address on selector unit
Bit 4-6 : Terminal address on CHLT/CHRT
Bit 7 : Zero

Bit 8-11 : Reserved
Bit 12-15 : Index indicating channel unit

0 = first CHLT
2 = second CHTL
4 = first CHRT
6 = second CHRT

DRKB01	KEYBOARD	DRKB01

Keyboard characteristics are defined in the DWT as follows:-

DWTCHP: (WORD 0)
- bit 10 = 0 Characters within /20 - /5F are accepted for Standard Read.
- bit 10 = 1 Characters within /20 - /7F are accepted for Standard Read.
- bit 11 = 0 Characters within /30 - /39 and /70 - /79 are accepted for Numeric Read.
- bit 11 = 1 Characters within /30 - /79 are accepted for Numeric Read.
- bit 9 = 1 For keyboards/BCRs with time-out.
- bit 7 = 1 Indicating input device.

DWTST: Bit 8 - 15. Device address times 2 for keyboard (WORD 1) with '8-bit' setting.

DWTKEY: (WORD) Code for the special keys KBEOR, KBCLR, KBBSP and KBMZ
If code conversion is used the converted code should be used.

DWTECH: (WORD) DWT-address of echo-device.

DWTTP: (WORD) Timer-indicator.

DWTCOD: (WORD) Address to code conversion table. Set to zero if no conversion.

The circular input buffer is also situated in the DWT at displacement . Its length must be the same in all DWT's.

Keyboard parameters are held in fields of the DWT as follows:-

DWTKEY: (WORD ?) Codes for special keys
KBCLR, KBBSP, KBMZ2, KBMZ3.
Note that converted code should be used.

DWTECH: (WORD) DWT-address of echo-device. Set to zero means no echo device.

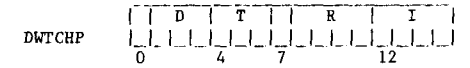
DWTTP: (WORD) Timer-indicator. Set to zero if no timing wanted on this keyboard.

DWTCOD: (WORD) Address to conversion address table CTABXX.

The circular input buffer is also placed in DWT. Its length should be the same in all DWT's. It starts at displacement ???????.

DRTP01 | TELLER TERMINAL PRINTER | DRTP01

Printer characteristics are defined in the first word of the DWT, DWTCHP, which is layed out as follows:-



The bits in this word have the following meanings:-

Bit 0 : Set to zero.

D (Bits 1 to 3) : Device address on selector unit.

T (Bits 4 to 6) : Terminal address on CHLT or CHRT.

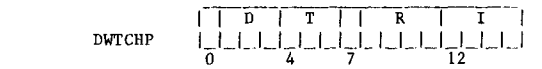
Bit 7 : Set to 0 if on CHLT.
Set to 1 if on CHRT.

R (Bits 8 to 11): Reserved.

Bits 12 to 15) : Channel unit Index.
Bit 12 set - first CHLT.
Bit 13 set - second CHLT.
Bit 14 set - first CHRT.
Bit 15 set - second CHRT.

DRTP02 TELLER TERMINAL PRINTER DRTP02

Printer characteristics are defined in the first word of the DWT, DWTCHP, which is layed out as follows:-



The bits in this word have the following meanings:-

Bit 0 : Set to zero.

D (Bits 1 to 3) : Device address on selector unit.

T (Bits 4 to 6) : Terminal address on CHLT or CHRT.

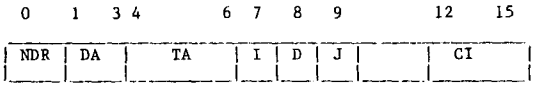
Bit 7 : Set to 0.

R (Bits 8 to 11): Reserved.

Bits 12 to 15 : Channel unit Index.
/0 - first CHLT.
/2 - second CHLT.
/4 - first channel on first CHRT.
/6 - second channel on first CHRT.
/8 - first channel on second CHRT.
/A - second channel on second CHRT.

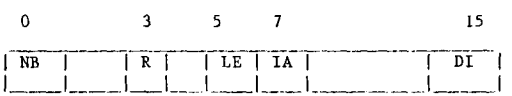
DRTP03	TELLER TERMINAL PRINTER	DRTP03

Word 0 DWTCHP (Channel parameters)



NDR: =1 No Data Request
DA: =2 Device Address
TA: Terminal Address on CHLT (CHRT)
I: =1 Input message allowed
D: Document inserted
J: Journal paper in
CI: Channel Index

Word 1 DWTST (Status)



NB: Not Busy
R: Recovery indicator
LE: Line feed Executed
IA: Interrupt Allowed
DI: Device Index (0=journal, 1=document)

Note: In the numbering of the following words, the number preceding the slash applies to systems without MMU, and the number following the slash to systems with MMU.

Word 8/10 DWTOTQ (Output queue link)

DWTOTQ contains the address to the device table of device queing for output via CHLT (CHRT).

WORD 9-11/11-13 DWT A3-A5 (Save area for A3-A5)

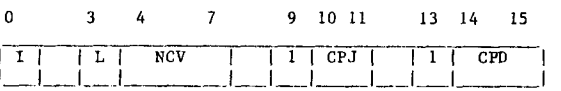
Word 12-15/14-17 DWTSB1-SB2 (A5 stack one-two level)

Word 16/18 DWTTP (Timer pointer)
DWTTP is used for time supervision of CHLT (CHRT).

Word 17/19 DWTTP (Printer parameters)
DWTPP are the actual printer parameters.

DRTP03 (continued) DRTP03

DWTPP

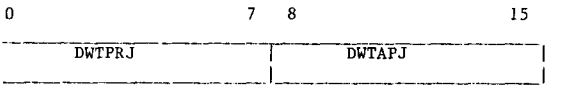


Bit 0 indicates that the printer and the document parameters have been set up after program loading.

L: Lower Case Indicator
NCV: National Character Variation
CPJ: Character Pitch for Journal
CPD: Character Pitch for Document

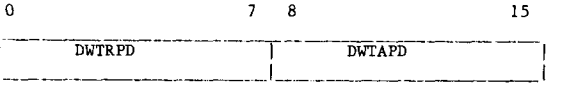
See Driver Description in M06/2 for details of above fields.

Word 18/20 DWTPPJ (Print Position Journal)



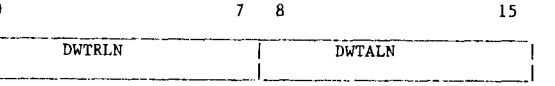
DWTRPJ: Requested number of print positions for journal.
DWTAPJ: Actual print position for document.

Word 19/21 DWTPPD (Print Position Document)



DWTRPD: Requested number of print position for document.
DWTAPD: Actual print position for document.

Word 20/22 DWTLN (Line Number)



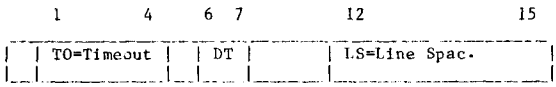
DWTRLN: Requested line number.
DWTALN: Actual line number.

DRTP03 (concluded) DRTP03

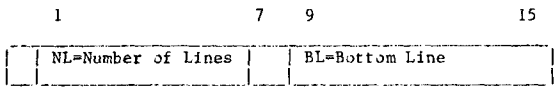
Word 21-25/23-27 DWTDP 1-5 (Document parameters)

These words are set up at system generation: They may be changed at run time by use of the order /27. See driver description in M06/2 for fuller details.

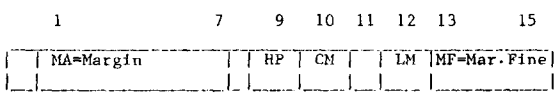
DWTDP1



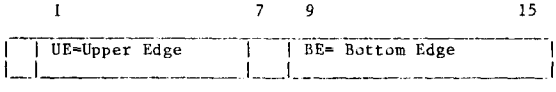
DWTDP2



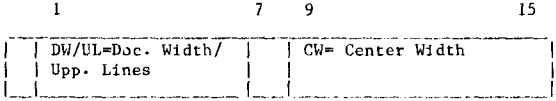
DWTDP3



DWTDP4



DWTDP5



DRTW01 CONSOLE TYPEWRITER DRTW01

Typewriter parameters are held in fields of the DWT as follows:-

DWTKEY: (WORD ?) This holds the codes for the special keys TWEOR, TWCLR, & TWBSP. Default codes for the keys are respectively /OD,/5E, & /5F.

DWTTP: (WORD ?) The Timer Indicator. Set to zero if no timing is wanted on this keyboard. If timing is not n the system then this word may be left out.

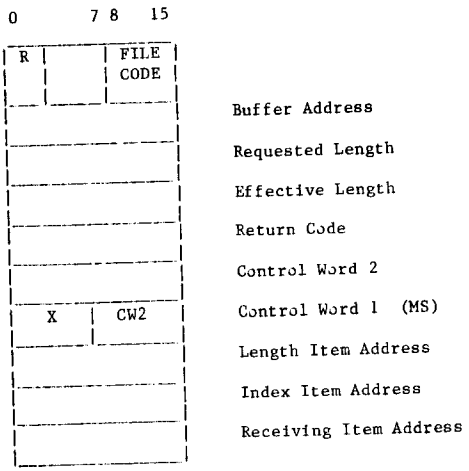
DSCB

DATA SET CONTROL BLOCK

DSCB

20.1

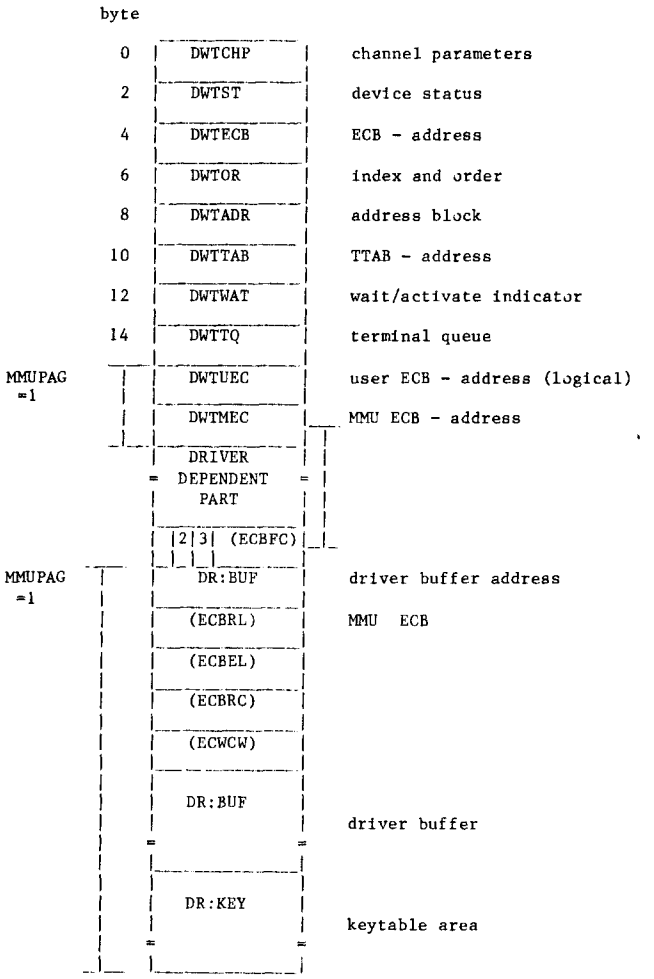
DSCB



DWT — DEVICE WORK TABLE — DWT

15.7

DWTx



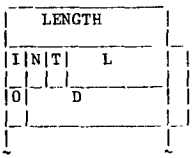
| D:zzz0 |

DESCRIPTOR TABLE

| D:zzz0 |

20.2

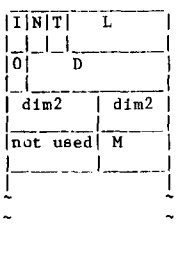
D:zzz0



LENGTH Length of table in bytes

Two words per single data item

I, bit 0, Zero
N, bit 1, Zero
T, bit 2, 3, type of data item 00=string
10=binary
11=decimal
L, bit 4-15, Length of the data item workblock
D displacement of the data item in the workblock



4 words used for an array definition

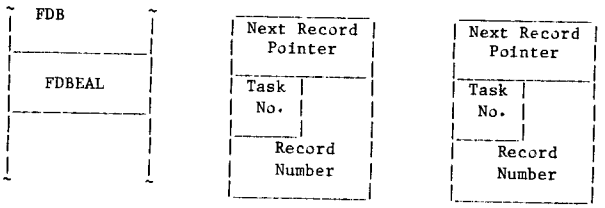
I, bit 0, one
N, bit 1, zero means one dimensional array
one means two dimensional array
T, bit 2, 3, type of array 00 = string
10 = binary
11 = decimal
L, bit 4-15, length of an array element.

EAL

EXCLUSIVE ACCESS LINK

EAL

19.4.5

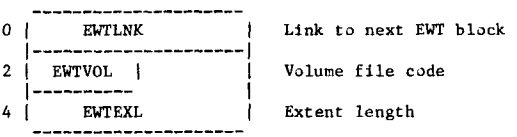


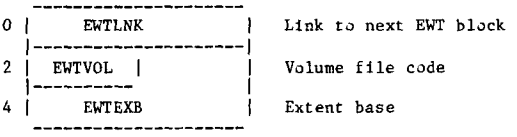
EWT

EXTENT WORK TABLE

EWT

19.4.6





FDB

FILE DESCRIPTOR BLOCK

FDB

19.4.1

FDB

	Link to next FDB	FDBLINK
		DWTST
		DWTECB
		DWTOR
		DWTADR
		DWTTAB
		DWTWAT
		DWTTQ
		DWTUEC
		MMU only
		DWTMEC
		DWTTDM
	FWT (see 19.4.6)	
0		FDBECB/ECBFC
2		ECBBA
4		ECBRL
6		ECBEL
8		ECBRC
/A		ECBCW1
/C		ECBCW2

/E	Sector Number	FDBSNR
/12	Relative Record Offset	FDBRRO
/14	Record Length	FDBRLE
/16	Block Factor | Task Number	FDBBLF/FDBTNR
/18	No. of indexes | Last Record Number	FDBNIF/FDBLRN
/1C	EA Link Root	FDBEAL
/1E	CRN Link Root	FDBCRL
/20	Key Address/Index Counter	FDBKA
/22	Master Index Address	FDBMIA
/24	FDB Address, Data File	FDBADF
/26	FDB Address, Index File 1	FDBAI1
/28	FDB Address, Index File 2	FDBAI2
/2A	FDB Address, Index File 3	FDBAI3
/2C	FDB Address, Index File 4	FDBAI4
/2E	D/B Option | DM Task Id	FDBDBR/FDBDMI (D=Delay, B=Basic)
/30	Maximum Record Offset	FDBMRO
/32	Block Size | COMMIT Flag	FDBBLZ
/34	ECBFC	
/36	DR:BUF	
/38	ECBRL	
/3A	ECBEL	FDBMEC (MMU only)
/3C	ECBRC	
/3E	ECBCW1	
/40	ECBCW2	
	Index Key Area	FDBKEY (at /34 or /42)

FWT

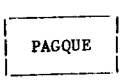
FILE WORK TABLE

FWT

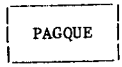
19.4.6

FWT

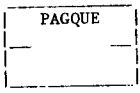
0	FWTLNK	Link to next FWT
2	DWTST	
4	DWTECB	
6	DWTOR	
8	DWTADR	
/A	DWTTAB	
/C	DWTWAT	
/E	DWTTQ	
/10	FWTVTC	VTOC Sector no. (First extent)
/12	FWTPAR | FWTFNR	Access params / File number
/14	FWTTAB	TTAB pointer for Exclusive Access
/16	FWTEW1	Pointer to next EWT block
/18	FWTSEX	File section/extent no (0,0)
/1A	FWTEXL	Extent length (1st extent)
/1C	FWTEW2	Pointer to next EWT block
/1E	FWTVOL	Volume file code (1st extent)
/20	FWTEXB	Extent base (1st extent)
/22		
/24	FWTNAM	File name
/26		
/28		
/2A	FWTQUE	Queue anchor for requests when attached (overwritten when DM is in use)

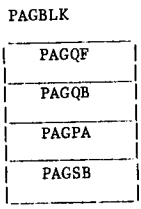


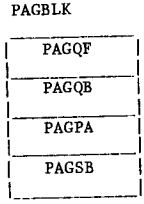
PAGE QUEUE



18.4







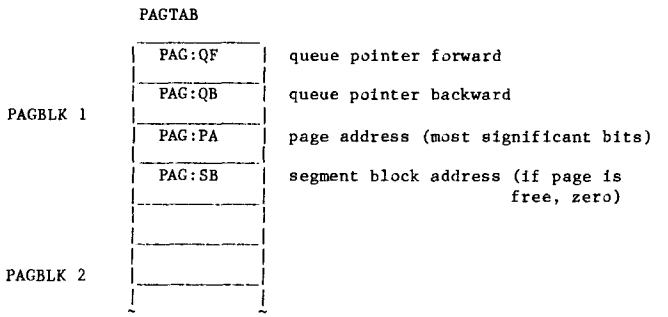
LRU

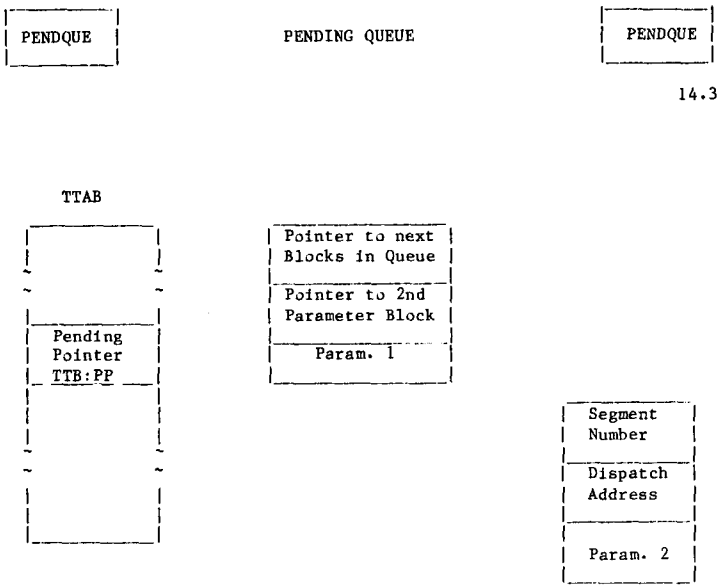
PAGTAB

PAGE TABLE

PAGTAB

18.6.2




PENDQUE
PENDING QUEUE
PENDQUE
TTAB
Pending Pointer TTB:PP
Pointer to next Blocks in Queue
Pointer to 2nd Parameter Block
Param. 1
Segment Number
Dispatch Address
Param. 2

14.3

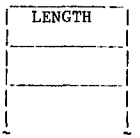
PFTAB

POWER FAILURE TABLE

PFTAB

17.3

PFTAB

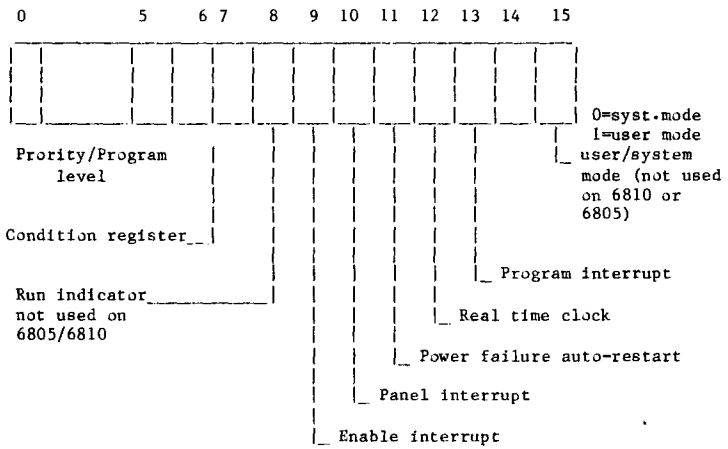


PSW

PROGRAM STATUS WORD

PSW

4.3.4



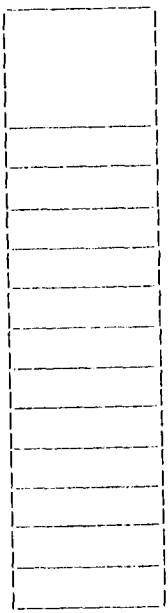
P:MTAB PROGRAM TABLE P:MTAB

24.11

Each item is two bytes. Pointers to:-

byte		
0	T:ATAB	Task control area table
2	U:BTAB	User work block control table
4	I:NTPA	System start address (in interpreter)
6	P:BAS	Start a base module (P:PIL)
8	T:BAT	Branch address table
A	T:CAT	Call address table
C	T:PAT	Perform address table
E	T:LIT	Literal point pointer
10		Highest index +1 in T:LITs
12	T:KEY	Key table pointer
14		Highest index +1 in T:KEY
16	T:PIC	Picture pointer
18		Highest index +1 in T:PIC
1A	T:FMT	Format pointer
1C		Highest index +1 in T:FMT
1E	P:END	End of base module (P:PIL)
20	T:AID	Task ID table (for credit debugger)
22	OPTION	SYSTEM option (SCTOPT)
24	LITADR	Literal addressing mode
26	ADRMOD	Data adressing mode
28	S:BTAB	Swappable workblock control table
		5 words used by Assembler debugger.

P:PIL COMMON CODE PART P:PIL



SCT — SYSTEM CONTROL TABLE — SCT

15.3

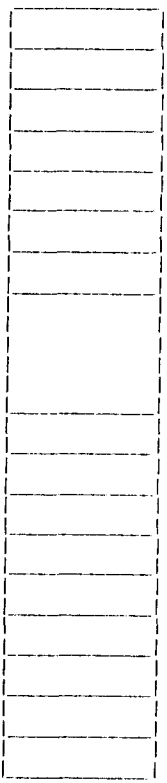
byte		
/9E	SCTMSZ	memory size (kB)
/A0	SCTSFA	start of free area (2 words)
/A4	SCTEFA	end of free area (2 words)
/A8	SCTIPL	program loading device (C0,C1,F0,F1,F4,F5,F8,F9) cassette, flex-disk, or disk.
/AA	SCTANO	application number
/AC	SCTADA	application disk sector address (2 words)
/B0	SCTIOE	application restart address
/B2	SCTTCT	TCTAB address
/B4	SCTCDT	CDTAB address
/B6	SCTPAG	PAGTAB address
/B8	SCTSWB	SWBTAB address
/BA	SCTNOP	number of pages
/BC	SCTPSZ	page size (bytes)
/BE	SCTMMC	MMU-table common part entry (index rel.TTAB entry)
/C0	SCTLAC	logical address of common part
/C2	SCTMMP	MMU-table page entry (index rel. to TTAB entry)
/C4	SCTLAP	logical address of pages
/C6	SCTNPE	number of page entries

/C8	SCTSTB	system stack base
/CA	SCTOPT	system options
/CC	SCTBUG	debugger address (0 = not included)
/CE	SCTDMT	Data management information
/D0	SCTDMI	DM index record buffer size
/D2	SCTFWT	File Management; FWT address
/D4	SCTNOF	DM/FM; number of files
/D6	SCTNFT	DM/FM; number of files per task
/D8	SCTFWL	FM; FWT length in bytes
/DA	SCTBLK	number of blocks per task
/DC	SCTDCT	DC task in system

SEGMENT

LAYOUT OF SEGMENT X

SEGMENT



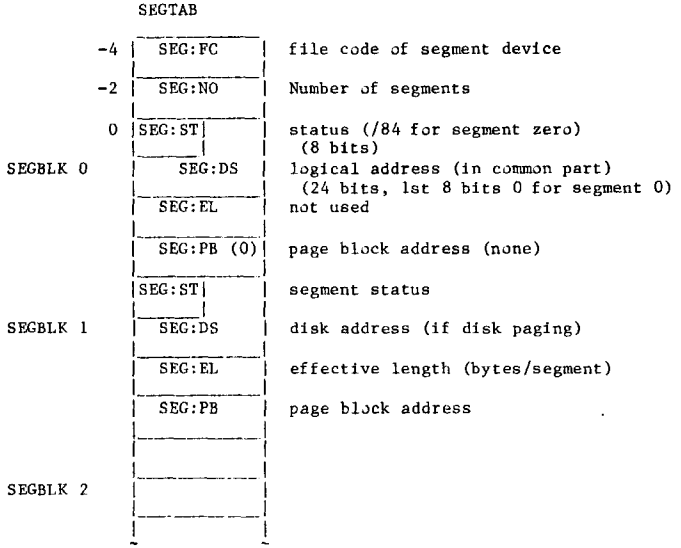
The literal pool descriptor pair etc. consists of two words. The first word points to the corresponding descriptor table and the second word points to the pool itself.

SEGTAB

SEGMENT TABLE

SEGTAB

18.6.1

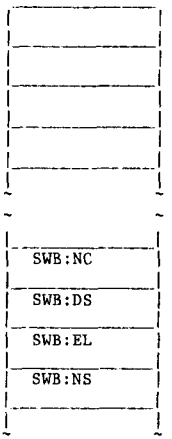


SWBTAB

SWAPPABLE WORKBLOCK TABLE

SWBTAB

18.6.3



Number of types of SWB's

Pointer to a block of the first type

Pointer to a block of the second type

Number of copies

Disk address of first copy

Length in bytes

Number of sectors

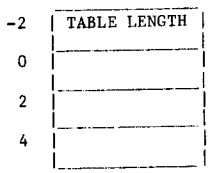
S:BTAB

SWAPPABLE WORKBLOCK CONTROL TABLE

S:BTAB

20.5

S:BTAB



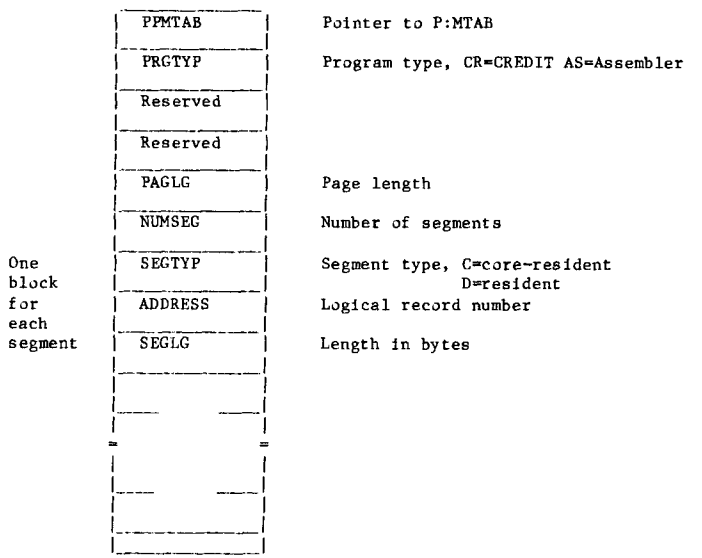
S:GTAB

SEGMENT TABLE

S:GTAB

24.10.1

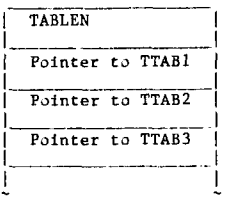
Each item occupies two bytes.

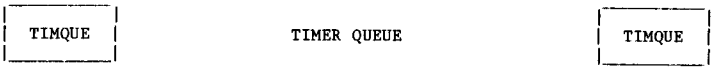


TCTAB	TASK CONTROL TABLE	TCTAB

15.4

TCTAB





16.2

TIMQUE

Pointer To Next Timer In Queue
Pointer To 2nd Timer Block
Timer Value (Negative)

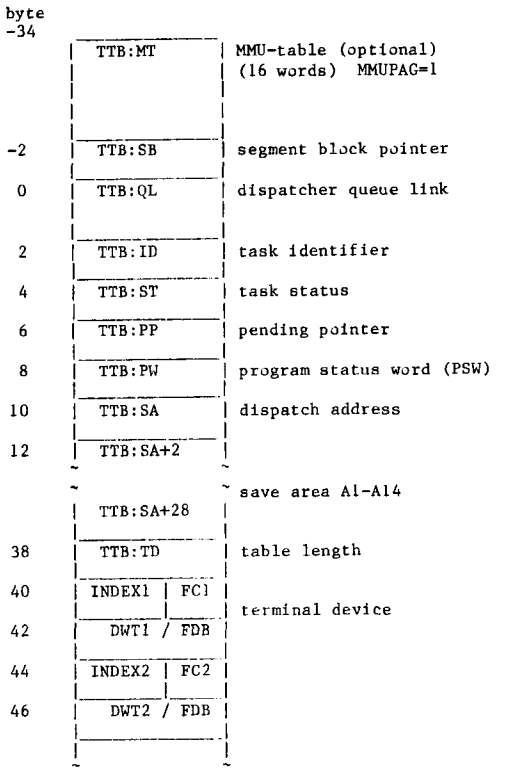
Time Out Address
Parameter

TTAB

TASK TABLE

TTAB

15.5

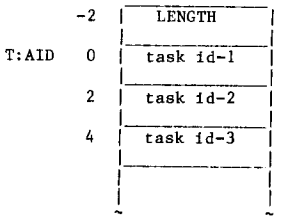


T:AID

TASK IDENTIFICATION TABLE

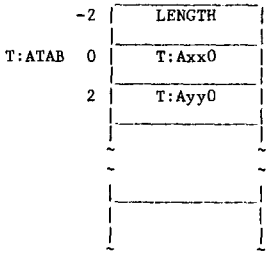
T:AID

24.10.2



T:ATAB	TERMINAL CONTROL AREA TABLE	T:ATAB

24.10.2

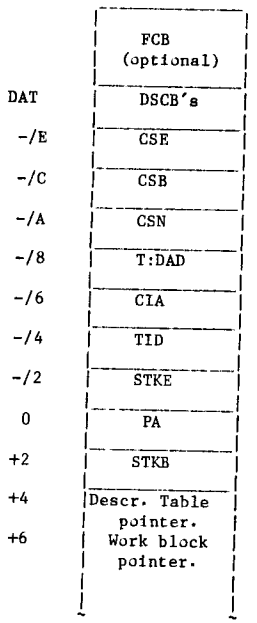


T:Axxy

TASK CONTROL AREA

T:Axxy

20.1



Format Control Block (33 words)

10 words for each data set

Current segment end

Current segment base

Current segment number

Pointer to task descr. table (T:Dxxx)

Current instruction address

Task identifier

Stack end pointer

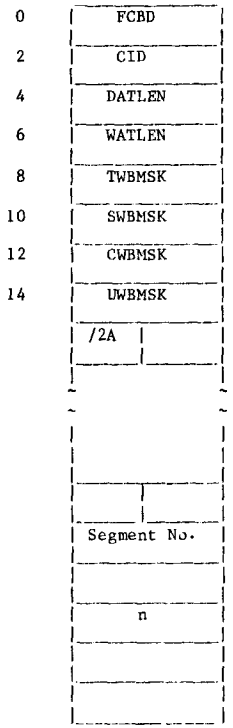
Current stack pointer

Stack base pointer

WAT Working storage allocation table pointers

T:Dxxy	TERMINAL CLASS DESCRIPTOR TABLE	T:Dxxy

20.3



Displacement of FCB in T:A

Terminal class identifier

Number of entries in DAT

Number of entries in WAT

Mask for TWB's

Mask for SWB's — one bit (=1) per WAT entry

Mask for CWB's — per type

Mask for UWB's

Dummy block table. One byte for each entry in WAT.
If DWB the corresponding workblockindex is put here.
If not DWB, the byte is zero.
Length of table is 2 to 16 bytes.

Re-entry point, 1st word is segment number (=FFFF if not in existance)
2nd word is logical address in segment

n is number of start points
1st start point. 1st word is segment number, 2nd word is logical address in segment.
2nd start point, etc.

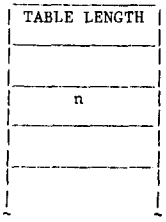
U:BTAB

USER WORKBLOCK TABLE

U:BTAB

20.4

U:BTAB



Address of first block

Number of blocks

Length of a block