

ksm

general description



CATS 35

RCSL: 52-AA188
pp: 1 : 33
Author: Fritz Krogh-
Jespersen
Edited: February, 1975

CATS 35
General Description

Keywords: RC 3500, CATS 35, Communication, Terminals, Network,
Software, General Description, Front-end.

Abstract: This paper is a general description of the Communication and
Terminal System implemented on RC 3500. RC 3500 runs as a
node in a generalized network, simultaneously servicing many
computers and many types of terminals.

A/S REGNECENTRALEN
3-9, Guldsmædgade
DK-8000 Aarhus C.

CONTENTS

Section

PREFACE	1
INTRODUCTION	2
OVERALL DESCRIPTION	3
Concepts	3.1
Design Objectives	3.2
System Structure	3.3
Primary Centre	3.3.1
Secondary Centres	3.3.2
PROTOCOL AND MESSAGE FORMATS	4
Message Format	4.1
Protocol	4.2
IMPLEMENTATION	5
Version 1 and the Design Objectives	5.1
Line Administration	5.2
Buffer Administration	5.3
Deadlock Prevention	5.4
EXAMPLE	6
AVAILABLE MODULES	7
Primary Centres	7.1
Secondary Centres	7.2
Miscellaneous	7.3

pp. 1 : 33.

This paper is a general description of CATS 35 (vers. 1), i.e. Communi-
And Terminal System implemented on the RC 3500 General Purpose
Controller.

The paper is written for people who want to get a basic knowledge of
CATS 35, the ideas and concepts behind it and an impression of the
system possibilities. Those who need a deeper understanding of the modules
in CATS 35 (normally people who are going to implement new modules)
should, after reading the present paper, continue reading the relevant
module descriptions.

CATS 35 was designed by Flemming Hermansen and Fritz Krogh-Jespersen
and implemented by Flemming Hermansen, Henning Jakobsen, Fritz Krogh-
Jespersen, Bo Bagger Laursen, Inger Møller, and Poul Vestereng, all RC
employees.

The designwork started in November, 1973, and the first installation was
running in production environment on July 2, 1974.

This first installation was at Jydsk Telefon-Aktieselskab (JTAS) who ordered
the system and thereby initiated the development project. The following
JTAS employees participated in several ways in the project: Folmer Eller-
bæk, Kristian Hyllested, John Krarup, and Kjeld Schmidt-Møller.

A communication system is a system which transfers information from one location to another. The kind of communication system we are dealing with in this paper is a system where the information originates from terminals and computers and is destined for terminals and computers.

Such a system may be divided into three levels (see fig. 2.1). Level 3 is the system containing the computers and the terminals, level 2 contains a logical network tying computers and terminals together, and level 1 contains the transmission lines and transmission facilities used by the communication system.

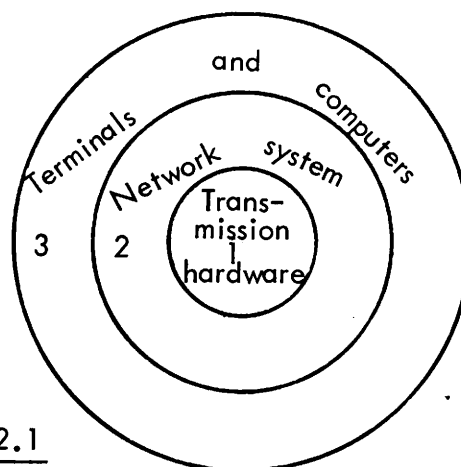


Fig. 2.1

The computers in level 3 are (in respect to communication) concerned with jobs like:

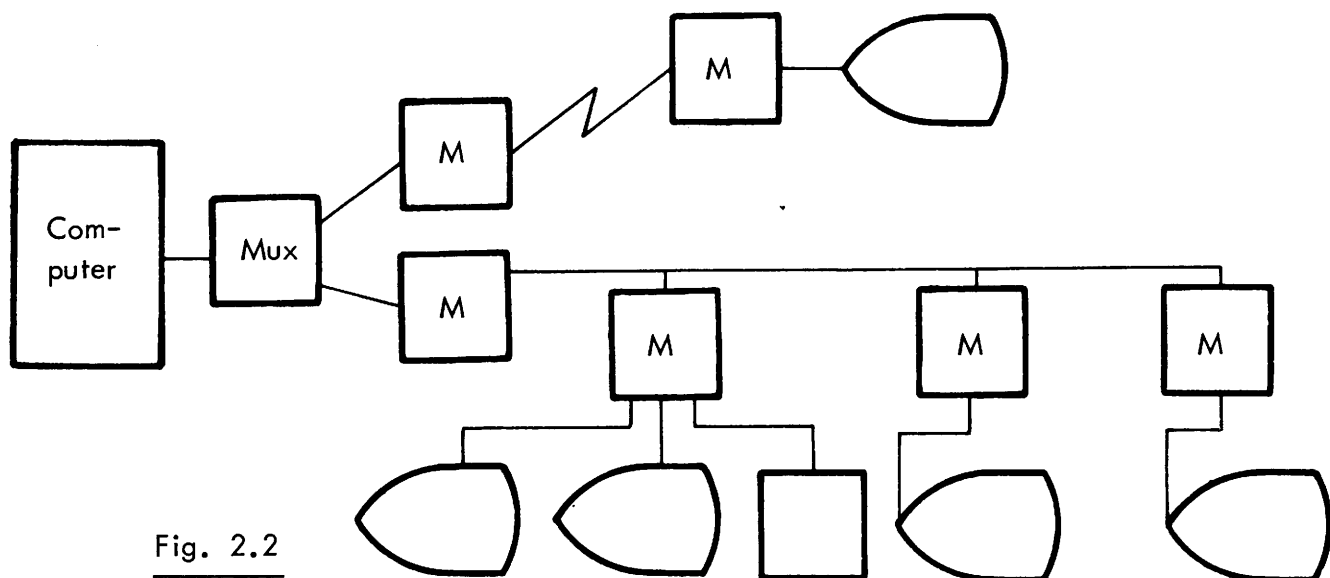
- a) checking a terminal's right to access the computer (login),
- b) receiving data from terminals and after processing of the data delivering of output data to the terminals,
- c) logging of input and output data with the purpose to make it possible to reprocess or retransmit data in case of severe errors in the network or in the computer.

The terminals in level 3 are the places where data from the users are entered to the communication system, and where data to the users are delivered from the communication system. The user should normally not be aware of the existence of level 1 and 2 as he just enters data and receives output.

The hardware in level 1 consists of telephone lines, telex lines, radio links, modems, central exchanges etc. and are the physical components used to transmit information from one location to another.

In this paper we will not focus on level 1 and level 3 except for some remarks where it is necessary to point out interesting properties in relation to level 2 - the network system.

In very simple communication systems there might be a direct connection between level 1 and level 3, for instance this is true in most systems containing only one computer (fig. 2.2).



In such a system the computer knows all the terminals and supervises the traffic flow on the lines (level 1) directly via a multiplexer or an equivalent device.

In more complicated communication systems containing many computers and terminals (maybe of different makes), you must have a network system which makes it possible to supervise the traffic flow independent of a special computer and to communicate between all (or at least many of) the participants in the system. This means that it is possible to share lines etc., thereby saving money and often also achieving a higher transfer rate and shorter response times. Furthermore, it is possible to use the same terminal in connection with different computers and to simplify the communication software in the computers.

A network system with these facilities is what is discussed in this paper. It may have the general structure shown in fig. 2.3.

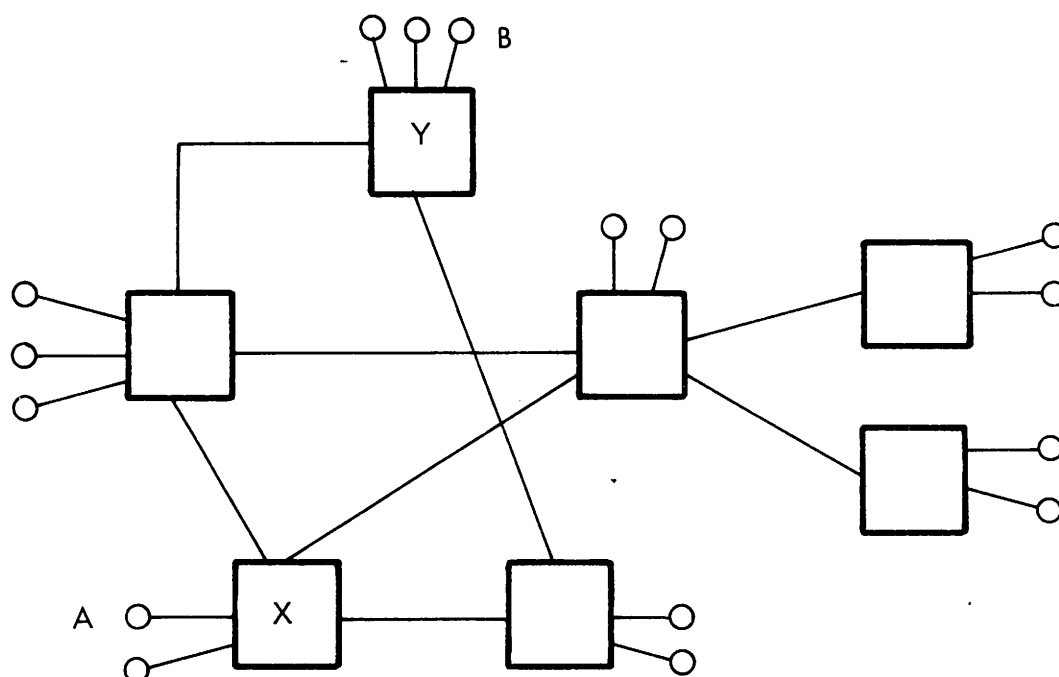


Fig. 2.3

The circles symbolize terminals and computers while the squares symbolize "concentrators" which are the network components taking care of the transfer from one circle to another.

Assume that the terminal A wants to send data to the computer B. The data are then entered to the "concentrator" X and transferred via the communication hardware to the "concentrator" Y from where it is delivered to the computer B. The route, the data from A to B follow, is determined by the network system and may vary from time to time.

Concepts.

3.1

To facilitate the description of CATS 35, some basic concepts used in the following must be defined.

Node. A node is a computer containing a primary centre. It may furthermore contain one or more secondary centres.

Primary centre. A primary centre is the part of the software in a node which performs routing of messages, connection to communication lines, error recovery and network supervision.

Secondary centre. A secondary centre is the part of the software in a node which adapts external communication protocols, formats, code sets etc. to the standardized CATS 35 format and code set, and may execute functions of different types for users.

Process. A process is defined as the sequential execution of an ordinary program possibly containing loops, conditional branches, procedure calls, etc.

Host. A host is a computer performing jobs for users. It is connected via a secondary centre to a primary centre. A node may besides its other functions act as a host.

When a terminal and a host communicate, it is in fact a process (a program) in the host and a process driving the terminal which communicate. The secondary centre (which is a process) connecting a host to CATS 35 is from CATS 35's point of view the host itself (you may call it a host adaptor), and the communication takes place between this process and the process driving the terminal. This last process is from CATS 35's point of view acting precisely as a host, so we are always dealing with communication between hosts.

Comname. Every program module (process) in CATS 35 which shall be able to communicate with other modules via primary centres must have one or more unique comnames. Let us as an example consider a secondary centre adapting more than one terminal. Here the secondary centre must have a comname for each device.

Originator. The originator is the program module from which a specific message originates. The comname of the module is called the originator name.

Destinator. The destinator is the program module for which a specific message is destined. The comname of the module is called the destinator name.

Message. A message is a block of data to be transmitted between two program modules in CATS 35. The first part of the message contains the originator name and the destinator name.

Line. A line is the communication line connecting two neighbouring nodes.

Link. A link is the logical connection between an originator and a destinator.

It is very important to distinguish between the link and the physical connections between the originator and the destinator. Some systems use the link concept to ensure that only a limited number of communications occurs at the same time. It is hereby possible to achieve a more effective traffic flow, a better utilization of the communication systems resources, and prevent the possibility of deadlock.

Linktable. The linktable is a table in each node containing the destinator name (and some other information) of every destinator who can be "reached" through that particular node.

Path. A path is the actual succession of nodes, through which a message passes on its way to the destinator.

The design objectives of CATS 35 are:

- a. It shall be possible for every computer/terminal to communicate with every computer/terminal, i.e. all program modules are potential originators/destinators.
- b. Computers of different makes shall be able to use a common CATS 35 system.
- c. It shall be possible to connect terminals of arbitrary types to a CATS 35 system.
- d. Computers shall be unaware of the actual make of terminals, i.e. a computer can for example assume that a terminal is a synchronous block-oriented CRT while it in fact is a TTY.
- e. Computers need not know the actual topology of the network.
- f. An arbitrary topology of the network shall be allowed.
- g. The communication software in hosts need not be changed when using CATS 35.
- h. CATS 35 shall be able to handle messages with respect to priority.
- i. CATS 35 shall guarantee that messages are not duplicated irrespective of errors of any kind in the network.
- j. CATS 35 shall not guarantee that messages are not lost in case of errors in the network, i.e. a host must be able to reenter a message to CATS 35.
- k. CATS 35 shall perform conversion of transmission speed and transmission mode (conf. d).
- l. CATS 35 shall be a store- and forward-system.

- m. CATS 35 shall be able to utilize different transmission speeds on the lines within the same network.
- n. Network status reporting may be performed centralized or decentralized.
- o. Message rerouting and network reconfiguration shall be performed automatically and decentralized.
- p. It shall not be possible for an originator to monopolize the system by entering a large number of messages with high speed.

System Structure.

3.3

The structure of a CATS 35 network system is shown in fig. 3.1.

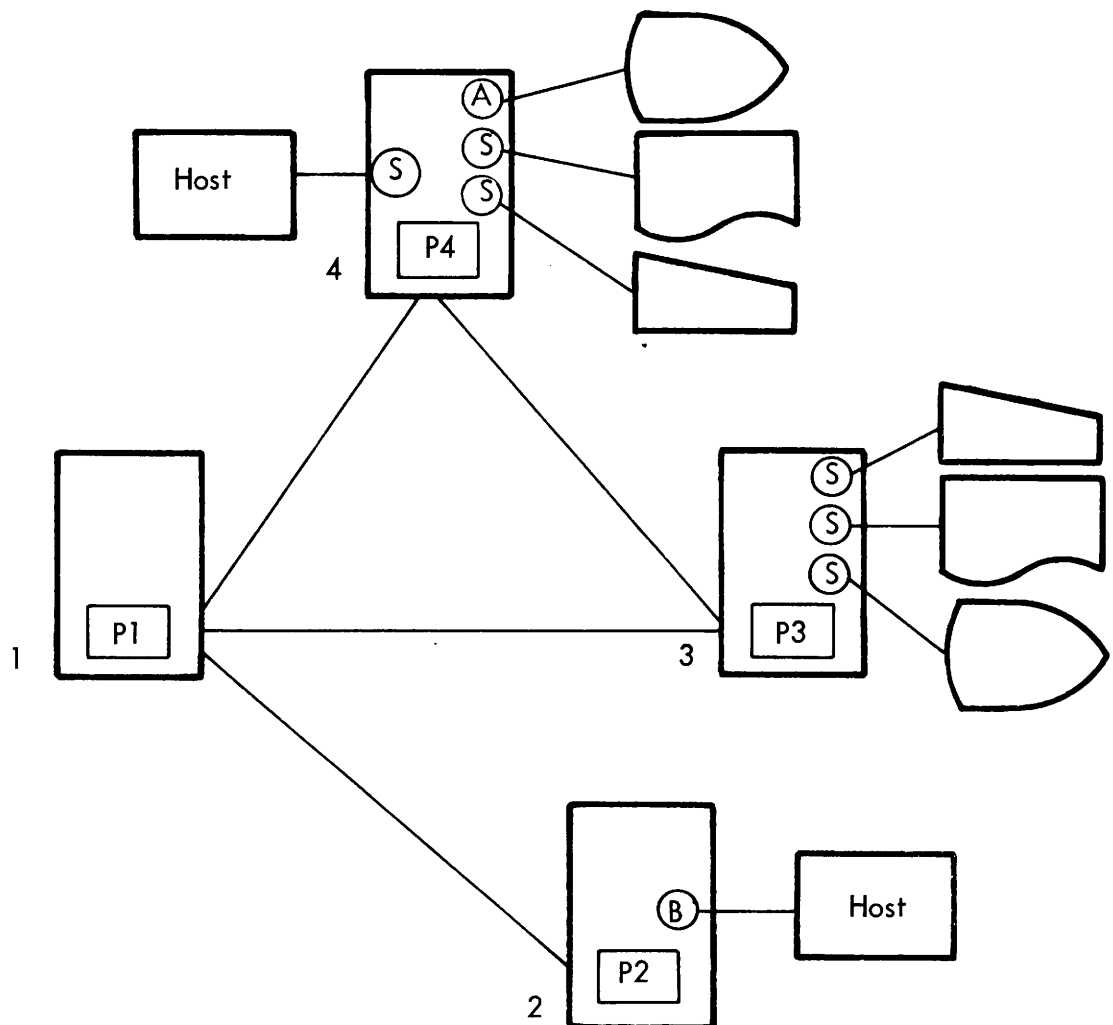


Fig. 3.1

In the figure is shown four nodes, each of them containing a primary centre (P). Node 1 acts only as a node in the network (possibly to provide alternative pathes), while node 2 contains a secondary centre (B) adapting a host to CATS 35, i.e. node 2 acts as a front-end computer to the host. Node 3 contains some secondary centres driving terminals, i.e. the node acts as one or more hosts (depending on the relations between the secondary centres) to the terminals and other hosts in the system. Node 4 contains secondary centres adapting both a host and some terminals.

If the secondary centre A which services a CRT, wants to transfer data to the secondary centre B which services a host, it will deliver the data structured as a message (i.e. containing the destinator and the originator names B and A respectively) to the primary centre P4. The primary centre will then check if the link exists and send the message on one of the outgoing lines to the next primary centre along the path. This center will repeat the linkcheck and finally the message will arrive at primary centre P2, from where it is delivered to B. B on its hand will then deliver the message to the host. The path a message follows is not known by a primary centre, because the centre only knows the next node on the path.

Primary Centre.

3.3.1

In each node exists a primary centre which is constituted by the modules shown in fig. 3.2.

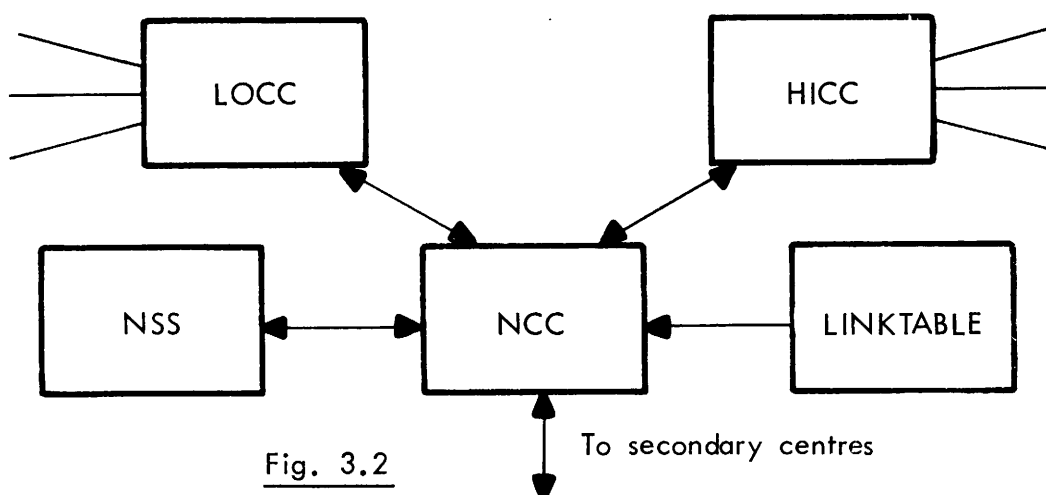


Fig. 3.2

NCC - Network Communication Controller - is the heart of a primary centre. All messages in a node pass through this module which by table-lookup in LINKTABLE distributes them to the next module on their way to the destination.

NSS - Network Status Supervisor - is a module which receives messages concerning line status and messages which cannot be delivered. Reconfiguration of LINKTABLE in case of line errors is also done by NSS.

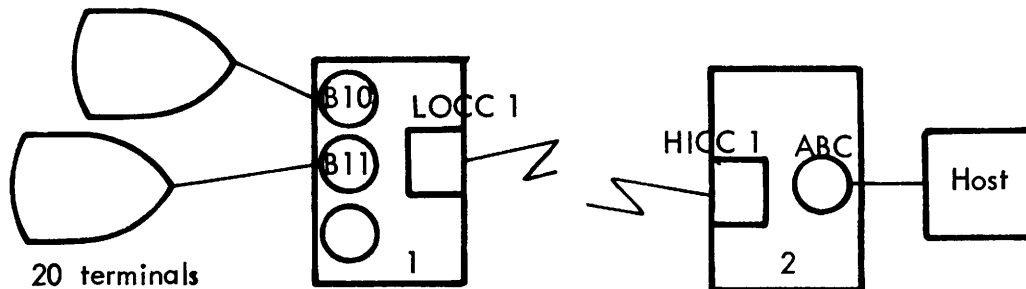
HICC - High Level Communication Controller - is the program module which is responsible for the polling of a neighbouring node and checking of messages exchanged with neighbouring nodes. Furthermore, it prioritizes the lines which it serves.

LOCC - Low Level Communication Controller - is a module which is a slave of a HICC in a neighbouring node, and services the "low" end of the line.

All messages (also between two modules in the same node) are passed through NCC. This means that they must have a standard format, which is described in chapter 4.

Furthermore, it means that a module which can receive messages must have an entry in the LINKTABLE, not only in its "own" primary centre, but also in all nodes through which messages to that module may pass. So the destination name must be present at all nodes along the path. If the destination name is not present, NCC will try to return the message to the originator, but if no link exists to him (i.e. there is no entry in linktable concerning the originator) it is delivered to NSS for further treatment. This means that it is possible to route messages, if a link between originator and destination exists, but if you want to communicate in the opposite direction, it is not possible because the link is unidirectional. So in all normal communications you will have two links going in opposite directions.

The link is simply represented by the comname which must be unique in the system. But many originators may use the same destinator name (i.e. they share the same link). As an example consider 20 terminals located at the same or different nodes. The terminals shall all communicate with the same host, so that they can share the same link to the host, but must have their unique link from the host (see fig. 3.3).



LINKTABLE in 1

ABC, LOCC 1
B 10, B 10
B 11, B 11
B 12, B 12
.
B 29, B 29

LINKTABLE in 2

ABC, ABC
B 10, HICC 1
B 11, HICC 1
B 12, HICC 1
.
B 29, HICC 1

Fig. 3.3

When a message to the host is passed through NCC in node 1, NCC will lookup ABC and direct the message to LOCC 1, and when it is passed through the NCC in node 2, it is delivered to ABC. When a message to a terminal (say B 12) is passed through NCC in node 2, it will be directed to HICC 1, and when it is processed by NCC in node 2, it will be delivered to B 12.

The NSS-module may via a secondary centre be connected to an operator's console, where line status changes and message delivery errors may be reported. It is also possible to let the NSS transmit its reports to a master centre simply by delivering messages to NCC with the master centre

(a secondary centre) as destinator. From the operator's console it is possible to reconfigure the LINKTABLE thereby changing the routing scheme. This reconfiguration can also be done automatically by NSS when it receives error messages from the HICC and/or LOCC or from some remote NSS.

The lines are oriented in a way so that there is always only one master (HICC) on each line, but a node may be master in respect to some lines and slave in respect to other lines. So you can have a primary centre containing both a HICC and a LOCC or containing only a HICC or a LOCC. This depends on the actual topology of the network. Let us as an example use the network shown in fig. 3.1. Here P4 contains only HICC, i.e. it is master to the lines P4,P1 and P4,P3. P3 contains for instance only LOCC meaning that it is slave for the lines P3,P4 and P3, P1. Now let P1 be slave to the lines P1,P4 and P1,P2 and master to the line P1,P3, so P1 contains both HICC and LOCC. Finally, we deduce that P2 contains only HICC. The concept of master and slave are important, because you can only have one to manage a common facility (the line), for instance in case of a line error. Then you have to ensure that the two modules at the ends of the line do not try to establish connection at the same time (the contention problem). Furthermore, it is possible for HICC (the master) to regulate the traffic flow on the lines dependent on the available buffers and computer power.

Secondary Centres.

3.3.2

The connection between the primary centres and the hosts is done via the secondary centres (which from the primary centres look like the hosts themselves). The function of the secondary centres is very widespread depending on the type of the device (terminal/computer) the secondary centre services. Let us therefore consider two typical examples.

Example 1

The first example is a secondary centre servicing a TTY-compatible terminal. This secondary centre will receive messages from NCC and deliver messages to NCC. Upon reception of a message the secondary centre must check if the originator is someone it wants to get messages from. If not, it may either discard the message or return it with delivery failure code. The message is contained in a buffer originally belonging to HICC or LOCC, and this means that the secondary centre has to empty the buffer within a limited time to avoid that the communication through the node will suffer from insufficient buffer space. Therefore, the secondary centre has to get rid of the buffer content (the data in the message) as soon as possible, either by writing it out on the TTY or copying it to an internal buffer, so that the empty HICC/LOCC buffer can be returned.

During the writing to the TTY some code conversions may be performed, and of course an adaption to the line speed must take place. After the writing some kind of acknowledgment may be sent to the originator telling him that he may transmit the next message.

When characters are typed they must be assembled in an internal buffer until an end of message, for example an end of line, is encountered. Then the secondary centre must restructure the data to form a message (may be at the same time make code conversions), i.e. setting up the originator and destinator in front of the data, and finally deliver it to NCC for transmission. After that the secondary centre may discard all typed characters, until a message acknowledge is received from the destinator.

Example 2

The other example is a secondary centre servicing a host via a communication line. Here the secondary centre must sustain the communication protocol to the host, i.e. send acknowledges, rejects etc.

When the secondary centre receives a message from NCC, it is queued until either a specific poll for that originator or a general poll is received. Then the message can be transmitted to the host, maybe after code conversion and reconfiguration of header etc. The comnames used

in CATS 35 have no relationship at all to the addressing method of the host. So it is the responsibility of the secondary centre to perform these conversions.

When the host transmits data to the secondary centre, the data are converted and restructured to form a message which include translation of the host's terminal addressing to a comname. Then the message can be delivered to NCC for transmission and an acknowledge can be transmitted to the host.

Also in this type of secondary centre it is necessary that buffers belonging to other modules (HICC/LOCC) are emptied and returned as quickly as possible. It can therefore be necessary to discard messages, if the host cannot accept input for longer periods.

The communication protocol used on the lines is a slightly modified subset of the ISO Recommendation 1745. This implies that only characters in the interval 0 to 127 decimal (not including transmission control characters) may be transmitted in the data portion of a message.

Message Format.

4.1

The messages which NCC can handle have the following format:

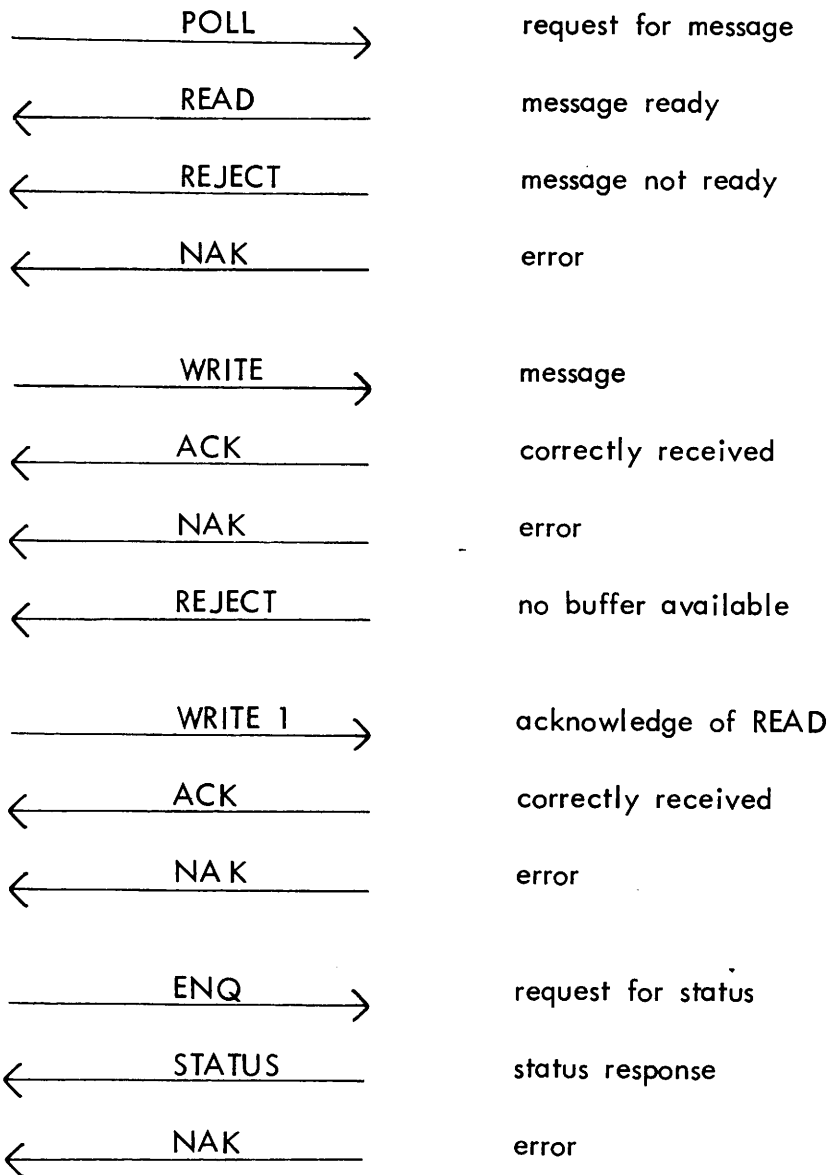
{MS} {destinatorname} {originatorname} {type} {TS} {data} {ME}

where:

MS	is the ISO character <u>SOH</u>
destinatorname	consists of 3 letters and/or digits
originatorname	consists of 3 letters and/or digits
type	X , normal transaction N , naked transaction (delivery failure) S , supervisory message
TS	is the ISO character <u>STX</u>
ME	is the ISO character <u>ETX</u>
data	Many secondary centres use the first character in data to tell the destinator about end of file, interrupt etc.

HICC

LOCC



Every WRITE and WRITE 1 contains a synchronization character A or B which is returned by LOCC in every data transfer. It is used by HICC to control if the last write message was correctly received by LOCC in case of line errors etc.

POLL	has the format:	$P \left\{ \frac{A}{B} \right\} \underline{\text{ENQ}}$
WRITE	has the format:	$\left\{ \frac{A}{B} \right\} \left\{ \text{Message} \right\}$
ENQ	has the format:	$E \left\{ \frac{A}{B} \right\} \underline{\text{ENQ}}$
WRITE 1	has the format:	$\left\{ \frac{A}{B} \right\} \underline{\text{ACK}}$
ACK	has the format:	$A \left\{ \frac{A}{B} \right\} \underline{\text{ACK}}$
STATUS	has the format:	$S \left\{ \frac{A}{B} \right\} \underline{\text{ACK}}$
NAK	has the format:	$N \left\{ \frac{A}{B} \right\} \underline{\text{NAK}}$
REJECT	has the format:	$R \left\{ \frac{A}{B} \right\} \underline{\text{NAK}}$
READ	has the format:	$\left\{ \frac{A}{B} \right\} \left\{ \text{Message} \right\}$

The underlined symbols represent the corresponding ISO characters

On the transmission lines all blocks are preceded by 3-4 synchronization characters, and all blocks which include a message are terminated by a block check character.

CATS 35 is as previously mentioned implemented on the RC 3500 mini-computer. This computer is a 16 bit machine with memory from 8K bytes to 64K bytes. The computer has an extremely fast interrupt response because each I/O channel (up to 32 channels) is connected to its own set of working registers. One of the registers in the set is the instruction counter, and each time the CPU has finished the execution of the current instruction, it selects via a priority selector the set of working registers with the highest priority. Then it takes the instruction counter of that set as current instruction counter. This means that the response time to an interrupt is practically zero, so in this way it is possible to service devices which have a high synchronous data rate, without the possibility of data overrun.

This feature is extremely important in a communication system where you have to service 10-12 lines, each with a transfer rate of 48K bits per second.

The software runs under control of MONITOR 1 which makes it possible to use reentrant program modules with respect to a priority dependent on the desired service level for the different modules. The hardware interrupt system is not under control of the monitor, so program modules communicating direct with devices are not scheduled by the monitor. These modules are called drivers. The communication between two program modules in the same machine is done via the monitor by means of semaphores.

Version 1 and the Design Objectives.

5.1

This section describes to what extent the version 1 of CATS 35 fulfills the design objectives mentioned in section 3.2.

- a. Totally fulfilled. It is only limited by the actual set up of the LINKTABLES. Of course, an originator must know the name of the destinator.
- b. Totally fulfilled. This depends only on the actual programming of the corresponding secondary centres.
- c. As b.
- d. As b.
- e. Totally fulfilled.
- f. Not fulfilled. In version 1 only a tree-structure of the network is allowed. This implies that no alternate pathes to a destinator exist and that no rerouting therefore is possible (except for messages with delivery failure).
- g. Totally fulfilled. Often it may be convenient to change the communication software to take full advantages of the power of CATS 35.
- h. Not fulfilled. Priority scheduling is not performed by the primary centre, but may be performed by the individual secondary centres.
- i. Totally fulfilled.
- j. Totally fulfilled. The retransmission may take place from the host or from the corresponding secondary centre.
- k. As b.
- l. Totally fulfilled.
- m. Totally fulfilled. A group of lines running at the same speed is serviced by one HICC-module, and you may have more HICC-modules in one node.
- n. Totally fulfilled. Only dependent on the set up of the LINKTABLES.
- o. Not fulfilled.
- p. Totally fulfilled. The secondary centres must communicate on a transaction-oriented basis, i.e. if originator A sends a message to

destinator B, he should await an answer from B before entering next message to B. It is possible to communicate in an n-buffered way, as long as every message corresponds to an answer.

Line Administration.

5.2

The lines are supervised by the HICC-module, i.e. a node may supervise all, some, or none of the lines connecting it to other nodes, depending on the presence of HICC-and LOCC-modules in the node. In version 1 the HICC-modules must be placed in such a way that no more than one HICC has connection to a node. A convenient example is shown in fig. 5.1.

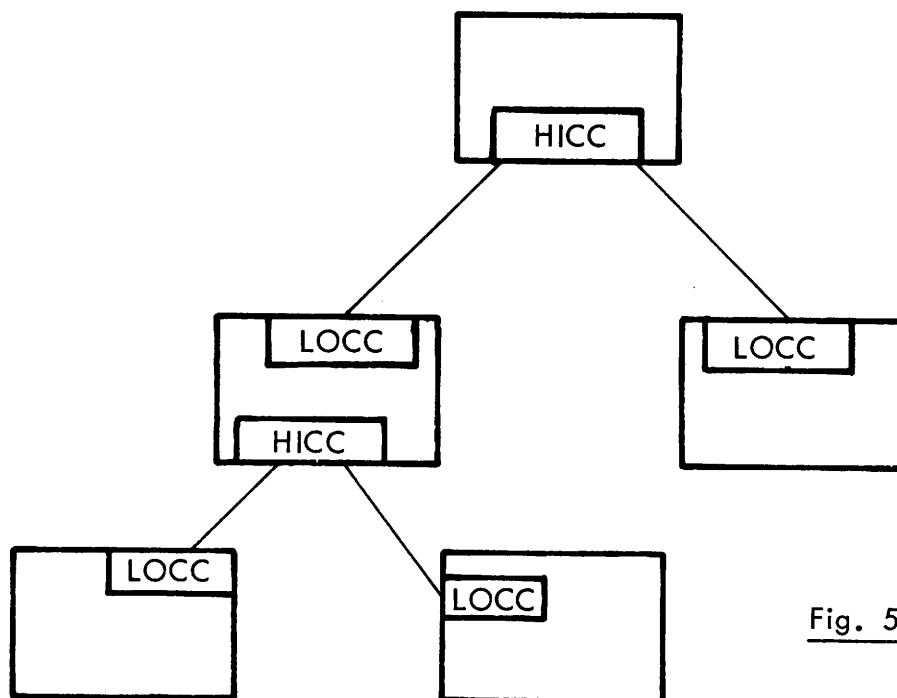


Fig. 5.1

This is due to the fact that HICC will service output before input, so that you may fill up a node, if two HICCs have lots of output to that node.

The transmission between HICC and LOCC follows the protocol shown in section 4.2. On all data blocks HICC may besides the listed answers get no answer, meaning that LOCC has not received the data block or

that the answer has not reached the HICC. In these cases a timeout occurs, and then HICC will try to establish connection by sending ENQ. As HICC supplies every WRITE and WRITE 1 with a block number (alternating A and B), it is possible for HICC by the status to determine which blocks have been correctly received by LOCC.

If a READ-message is in error, HICC simply repeats the POLL. In case of repeated transmission errors (or timeouts) the line is declared down and HICC will return pending messages to NCC for rerouting, while a status message is sent to NSS. HICC will now give the line a very low priority, so that attempts to establish connection (ENQ) and the timeouts will not degrade the performance of other lines.

While a line is active, HICC has to send POLLS with regular intervals; if not, LOCC will declare the line down and return pending messages while making a status message to NSS. This method has been chosen to prevent that messages can be pending at LOCC for infinite time, thereby blocking the buffers for other use. If too many consecutive datablocks are in error, or if no transmission at all occurs on the line, LOCC will also declare the line down.

When a line is in down-state, neither HICC nor LOCC will accept messages for it. HICC declares the line up after a successful enquiry and starts polling the line. LOCC waits for some complete protocols to be finished correctly before it declares the line up, but it may accept write-messages from HICC when line-state is down.

Buffer Administration.

5.3

A very simple buffer administration has been adapted. All buffers are preallocated in the following way. During start up of a node NCC distributes standard buffers to the other modules according to a buffer distribution list. A standard buffer has a fixed size (assembly option) and is

initialized with pointers, owner etc. by a "reserver" routine. Modules which need buffers of a special size may request them at start up time direct at the "reserver". If a module needs space for tables or communication with a host, it may request this at start up time at the "reserver".

With this allocation scheme every buffer has its special owner, and they do all have the same format. When a buffer passes through NCC, it is followed by a status word telling whether the buffer is full or empty. If empty, NCC returns the buffer to the owner, if full, NCC will look upon the destinator name and make its table lookup.

So when a module receives a message in a buffer which is not one of its own, it must remove the message from the buffer and return the buffer to NCC with empty status. It is not allowed to a module to fill data into a buffer belonging to another module, because it is then unpredictable at what time the buffer will be back at the owner.

The HICC and the LOCC will only read messages into buffers of their own, then distribute them to other modules via NCC and finally get them back with empty status. Correspondingly, a secondary centre will use one of its own buffers to contain a message from say a terminal, then send it to HICC/LOCC via NCC, where the buffer is queued waiting for transmission. After successful transmission the buffer is returned to NCC with empty status.

This way of administrating the buffers is very little complicated, but has as result that buffers must only stay temporarily at a "not-owner" module, i.e. they must be returned in one way or another within finite time, maybe by discarding the message (setting empty status) or by setting up delivery failure status.

Deadlock Prevention.

5.4

Due to the buffer preallocation and the rule of fast releasing of buffers, no deadlock can occur inside a node. Let us then consider a deadlock involving more nodes (fig. 5.2).

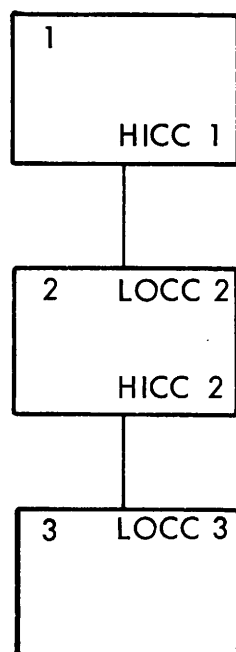


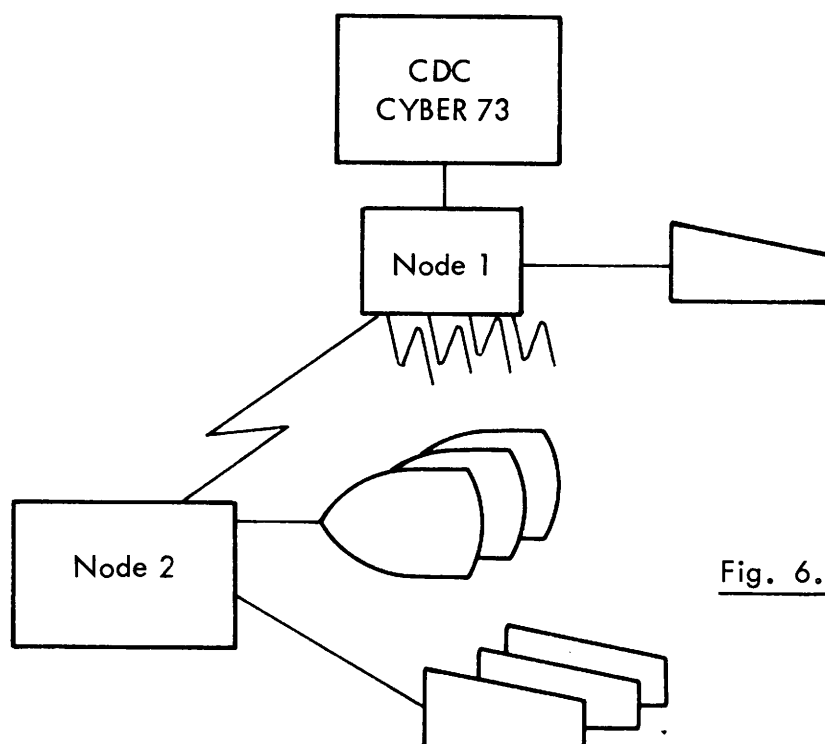
Fig. 5.2

A deadlock may occur when all of the HICC/LOCC buffers are full and pending for transmission. I.e. HICC 1's buffers are at some modules in node 1, LOCC 2's buffers at HICC 2, HICC 2's buffers at LOCC 2, and LOCC 3's buffers at some modules in node 3. Now no transmission can occur (although there are output messages at all transmission modules), because no empty buffers are available.

This situation is not a real deadlock, because it may resolve itself when LOCC 3 gets empty buffers, thereby making it possible to the HICC 2 to service its queue, thereby releasing a buffer for LOCC 2 etc. But remark that HICC prioritizes output over input, which means that it can take quite a while before the situation is resolved, but the system is in fact overloaded. As no one can monopolize the network, all queues at HICC will be emptied sooner or later, so that HICC will resume polling.

When LOCC has not been polled for some time, it defines the line as down and returns all pending messages to NCC with delivery failure status, where the message type is changed to N and the message routed back to the originator. An N-type message with delivery failure is by NCC sent to NSS, where the buffer is emptied.

As an example of CATS 35, let us consider part of the system installed at JTAS, and shown in fig. 6.1.



The distance between node 1 and node 2 is about 110 km, and the line is a full-duplex line transmitting at 48K bps. To node 2 are connected some CRTs and a local printer, later an extension with a computer is possible. Node 1 services 5 lines at 48K bps., and the connection to the CDC-CYBER is directly via a CYBER data channel with a maximum transfer rate of 10M bps. (900K words of 12 bits). A later extension will connect node 1 to two more computers used for a special enquiry system.

The modules in node 1 are shown in fig. 6.2.

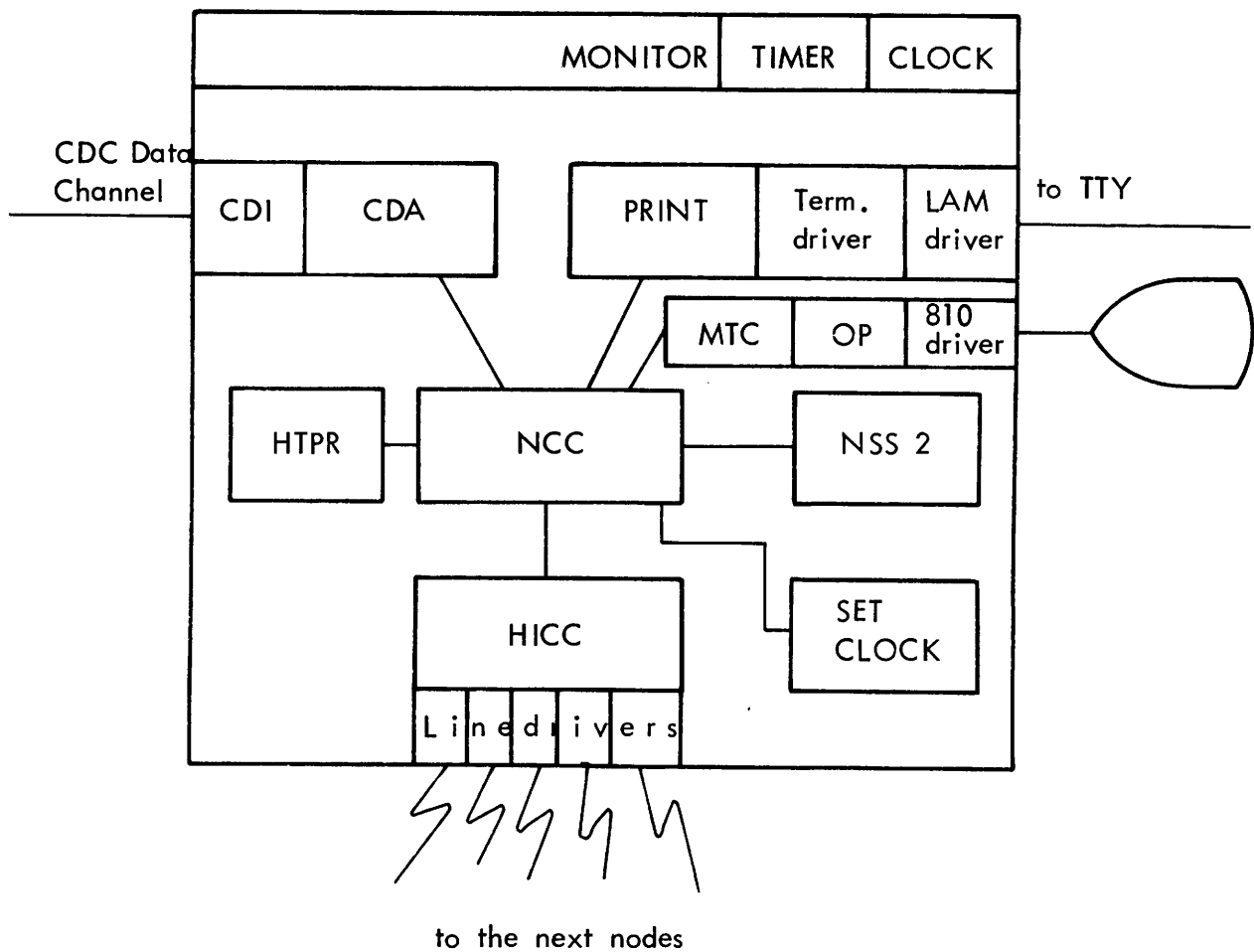


Fig. 6.2

CDI is a driver for the hardware interface connecting RC 3500 to the CDC data channel. Together with the CDA (Control Data Adaptor) module it controls the data transfers over the channel and supervises the transfer protocol. This protocol is developed especially for this purpose, and its main component is that the CDC computer (in fact a special program in a peripheral processor) requests a status from CDA about 10 times a second. In the status CDA tells, if it is ready to receive data from the CDC and/or it has data ready for the CDC. If data are ready, a read or write protocol is entered. CDA furthermore converts the data from the CDC (which are 6-bits characters with escape facilities) to ISO or from ISO to the internal CDC code set. Finally, CDA supervises the CDC in that way that if the CDC for a period does not ask for data or status at all, it is declared down and pending messages returned to NCC.

HPR is a module, the function of which is to copy messages from the CDC destined for the TTY to an internal buffer, thereby releasing the message buffer.

PRINT is a module which receives messages from other modules and re-formats them for printing. In this case the printing takes place on a TTY via a terminal driver and a LAM driver (Low Speed Aynchronous Multiplexer driver).

The TTY is used for status reporting etc. Some of the messages may come from the CDC computer and some from NSS 2. NSS 2 (Network Status Supervisor) is the module which receives messages concerning lines going up and down, message delivery failures, and CDC-CYBER breakdown. It forms from that information new messages which are sent to the PRINT module, after a time stamp has been added. The time stamp comes from the CLOCK-module, which delivers time, date, and weekday. The clock may be synchronized via the secondary centre SETCLOCK, which can receive the time date information from the CDC-CYBER or from some other source.

Connected to the node is an RC 810 - CRT, which is used as a head terminal for the realtime system in the CYBER. The CRT is driven via an 810 driver by the OP-module, which makes some code translations etc. when the SEND-key is pushed before requesting a buffer at MTC. When a message is received from MTC, it is code converted before it is written to the CRT.

The MTC (Multiple Terminal Controller) has as its primary function to be an administrator of a few (possibly only one) buffers to be shared by many terminals. You may have many OP-modules or other types of modules servicing terminals using the same MTC, thereby saving buffer space. Furthermore, MTC may administrate a queue of print buffers, used when a CRT wants to make hardcopy.

The modules in node 2 are shown in fig. 6.3.

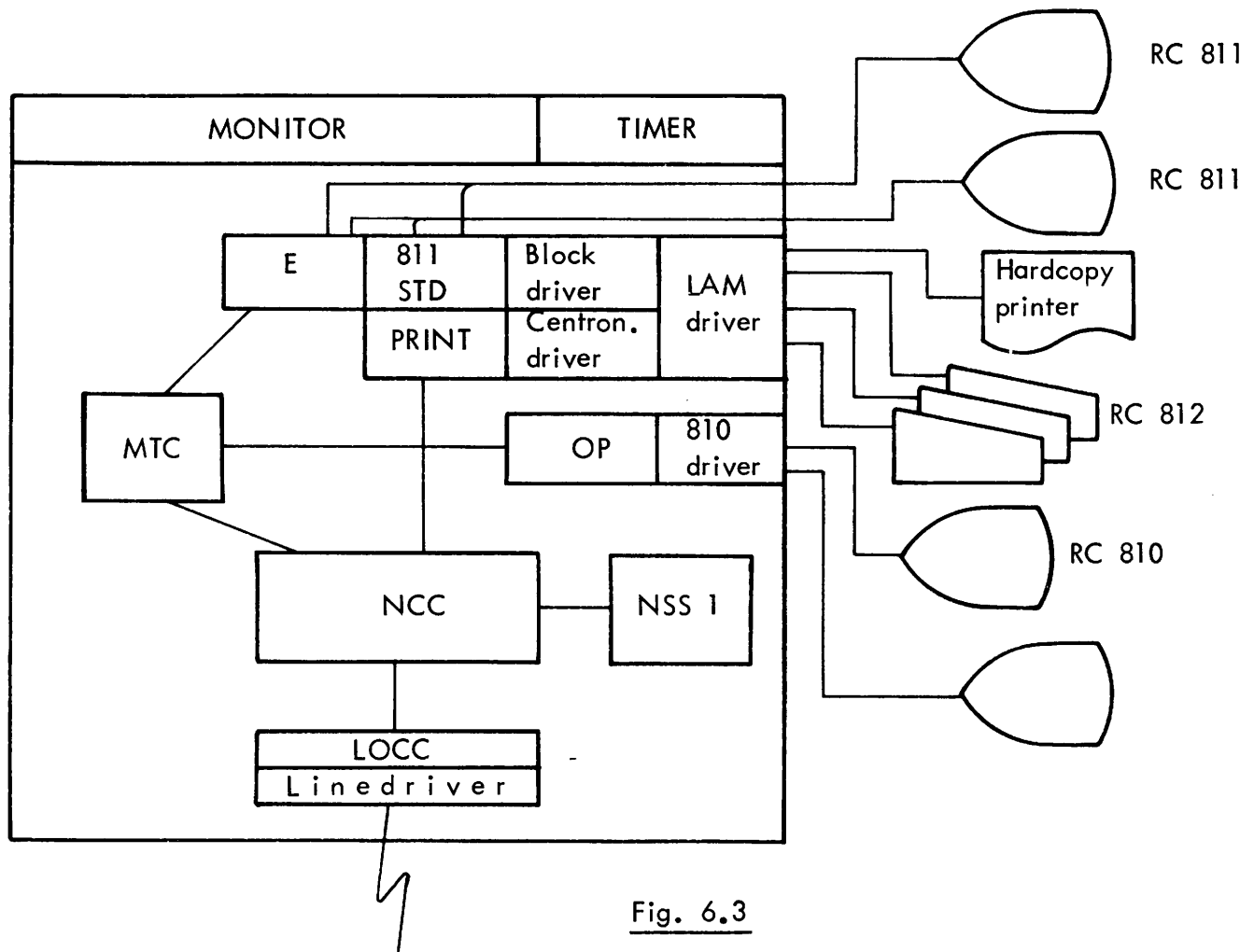


Fig. 6.3

Here MTC services many terminals via the OP-modules (reentrant) and the E-modules (reentrant). The E-module makes use of R 811 STD module (standard module for RC 811 CRT-s), a Block driver, and a LAM driver to service the RC 812 keyboards and R 811 displays in the way JTAS wants. That means for example setting up clichés on demand from the CDC-CYBER, checking the input in the open fields on the CRT etc. The RC 811 CRT is refreshed automatically from RC 3500 memory.

PRINT uses here a Centronics driver to print messages on a Centronics 306 printer. The messages contain hardcopies sent from E via MTC and NCC, so in fact the printer could be on another node in the network.

NSS 1 is a very simple version of NSS 2, because it only sets the empty status in all messages which it receives.

As you can see from this example, CATS 35 contains a lot of modules, each with a specific function. The connection between secondary centres are always via an NCC which means that the originator does not know (and does not want to know) whether the destinator is in the same node or not. It is all depending on the contents of the LINKTABLE.

This section contains only the modules which are CATS 35 modules.
Standard modules and drivers are not included.

Primary Centres.

7.1

RCSL: 52-AA194	NCC 1	Network Communications Controller
RCSL: 52-AA190	HICC 1	High Level Communications Controller
RCSL: 52-AA191	LOCC 1	Low Level Communications Controller
RCSL: 52-AA200	NSS 1	Network Status Supervisor (No reporting)
RCSL: 52-AA201	NSS 2	Network Status Supervisor
RCSL: 52-AA202	HSSI	Driver for the 48K bps. High Speed Synchronous Interface
RCSL: 52-AA203	BSC	Driver for the 48K bps. Block Synchronous Controller

Secondary Centres.

7.2

RCSL: 52-AA193	MTC 1	Multiple Terminal Controller
RCSL: 52-AA198	CDI 1	Driver for the CDC-channel interface
RCSL: 52-AA199	CDA 1	Control Data Adaptor
RCSL: 52-AA204	PRINT 1	Printer module for serial printers
RCSL: 52-AA205	TEST	Debugging module
RCSL: 52-AA217	HTPR 1	Head Terminal Printer Module
RCSL: 52-AA234	SETCLOCK	Setting of system time and date
RCSL: 52-AA257	UTC 1	User Terminal Controller (TTY compatible devices)

RCSL: 52-AA258	SDC 1	Special Device Controller
RCSL: 52-AA259	BA 1	Buffer Administrator
RCSL: 52-AA260	RCA 1	RC 4000 Adaptor
RCSL: 52-AA261	MSSI 4000	Driver for the Medium Speed Synchronous Interface (RC 4000 formats)
RCSL: 52-AA314	IBM CSC	Driver for the Character-informed Synchronous Controller (IBM-BSC format)
RCSL: 52-AA315	IBM BSC	Formatter for IBM-BSC format
RCSL: 52-AA316	RC 3271	IBM Adaptor (3271 CU Simulator)
RCSL: 52-AA317	RC 3277	IBM 3277 Display Station Simulator

Miscellaneous.

7.3

RCSL: 52-207	Installation Handbook
--------------	-----------------------

RCSL: 52-AA206	Utility Routines
----------------	------------------