X.25 co-processor functional design level 1 and 2

Klip, A.

Award date:
1985

Eindhoven University of Technology
Department of Electrical Engineering
Group of Digital Systems (EB)

X.25 CO-PROCESSOR

FUNCTIONAL DESIGN LEVEL 1 & 2

by A. Klip

Graduation report by A.Klip

Supervisor : Prof. Ir. A. Heetman
Coach : Ir. M.P.J. Stevens

Eindhoven, The Netherlands
August 1985

Summary

This report describes the functional design of a X.25 co-processor for data communication as recommended by the CCITT in X.25. The design of hardware and software of level 1 and 2 will be discussed. Level 3 and testing of the co-processor can be found in other reports from the Digital Systems Group of the Eindhoven University of Technology.

Contents

Preface

A study has been made of the X.25 protocol together with a design how this protocol can be implemented in a single chip co-processor. Level 3, the communication with the host processor and the data structure in the hosts memory have been studied by H.P.M.J. Schenkelaars.

This report contains the level 2 and level 1 analysis and the functional design of the hard- and software. The co-processor has been partitioned into several finite state machines, a number of shift registers and several processors, each for a special task in the X.25 protocol.

The chip has been designed for application in a DTE (Data Terminal Equipment) as well in a DCE (Data Circuit-terminating Equipment). The initialization procedures and the communication with higher levels will also be discussed. A decomposition of the hardware has resulted in quite simple descriptions of the parts of the chip. The chip is designed to work with an Intel or a Motorola bus, with 8 or 16 bits data structures and for a maximum bit rate of 64 kbit/sec.

The software for the host for the level 4 data handling is under study by E.P.M. Bakker. The test procedures and hardware to be integrated into the chip will be discussed by H. van Ooijen. All reports from the Eindhoven University of Technology.

The aim of this study was to design a X.25 co-processor that would not require large printed circuit boards with expensive software. The chips on the market are considered as only implementing parts of the 3 level deep X.25 protocol. This chip should contain all levels. The layer structure of X.25 should be maintained throughout the design. As a result of the study it seems feasible to develop such a single chip X.25 co-processor.

1 X.25 recommendation

The need of a standard protocol concerning data communication between computers has resulted in a recommendation of the CCITT. This X.25 recommendation has adopted the lower 3 layers of the 7 layer ISO (International Standard Organisation) model of Open System Interconnection. Open does not refer to a particular implementation. It simply indicates that the system supports standards which enable these systems to exchange information with all sorts of other users. The recommendation specifies the interfacing between DTE (Data Terminating Equipment) and DCE (Data Circuit-terminating Equipment). More generally speaking the DTE is the computer of a subscriber, or the intelligent terminal, and the DCE the network of the telecommunications company. Mostly the DCE is the first node in the data communications network.

X.25 gives rules for the lowest 3 layers of the ISO model. For the lowest layer parts of the CCITT recommendation X.21 have been used. The protocols described are about communication set-up and disconnection, flow control and error handling and (if possible) recovery. Because the hardware and software for these protocols are quite large, about 2 eurocards and 100k software, a single-chip solution would be very welcome to designers for datacommunication equipment. In this chapter an outline of the functions to be performed and the global architecture chosen are discussed.

1.1 Layers

The lowest layer of the ISO-OSI model is the Physical layer, also called the physical level. This layer defines the electrical and mechanical connection of the link between DTE and DCE. X.25 gives the paragraphs of recommendation X.21 which are supported for this layer. X.21 bis is a standard which is allowed for an interim period of time, and some administrations may offer a DCE/DTE interface in accordance with this. The design of the chip supports X.21 because this is the recommended protocol. The function of level 1 is to make an electrical connection between level 2 and the network (mostly present in the form of a modem). The working of the physical layer receiver and transmitter will be described in chapter 2.

The second layer is the Data-link layer. This layer concerns the frame level of the data communication. Data bits from higher levels are to be transported between DCE and DTE in such a way that an error free link is reached. To accomplish this, the level 3 data has been packed in frames in such a way that errors occurred during transport over the communication lines will be detected. Then the level 2 protocol has several ways to ask for retransmission. Or the logical link an be reset by transmitting a special purpose frame. Level 2 also offers a logical connection between level 3 of the DCE and level 3 of the DTE and uses transport via the levels 1, the modems and the network.

The third layer is the Network layer. On this so-called packet level, the data from a subscriber will be multiplexed with data packets from other subscribers. Over one logical link there can be more than 4000 logical channels. The host computer only has to allocate space for the receive buffers in the shared memory and to send a request to make a virtual call to an other computer or terminal. Level 3 then opens and closes the logical channels on level 3. It also controls the data flow of each logical channel separately. Error recovery is done by initializing and clearing of a single channel, or of all channels. Level 3 offers the host an opportunity to send a high priority byte of information which passes by the normal flow control.

Figure 1 shows how the principle of datacommunication with the X.25 protocol works. Higher levels present their data to level 3 which opens a logical channel and makes a virtual call to another station. The information from level 3, data or channel management information, are data frames for level 2. Level 2 converts these level 3 packets into level 2 frames and transmits them over the one logical link via the physical layer. At the destination on the other side of the network, the frames will be received and if they are not level 2 link management frames but level 3 information, the packets will be unpacked and if they are correct, passed to level 3 which distributes them to the several channels.

1.2 Data structure

Figure 2 shows how level 2 frames and level 3 packets are constructed. Level 4 data is presented by the host processor to the X.25 co-processor in a data format of 128 octets. Level 3 of the co-processor adds a packet type identifier (8 bit) and a logic channel number (8 bits). Preceding these there is a general format identifier/logical channel group

number. In a report written by H.P.M.J. Schenkelaars packing and unpacking of the level 4 data by level 3 has been described.

Sequencing and distribution of the packets are controlled by the header. Besides that a non-data packet can be sent for instance for setting-up a call or disconnecting one. The level 3 processing is very complex, so level 3 has its own sub-processors.

Level 3 packets have to be transferred between DCE and DTE, over a single line (logically seen), and free of errors. For this reason level 2 has its own sequence control protocol (for a single line) and an error detection mechanism. The level 3 data, 128 bytes + 3 info bytes, is therefore enveloped by 6 octets of level 2. First of all there is a leading flag, indicating that the data-, or control-frames is starting. After the flag there is an address, indicating if the frame is a command or response, and a control byte for the link maintenance and sequence control. If the frame is not a control frame, the so-called U or S frame, the level 3 packet follows. After the level 3 data an error detecting code, generated by level 2, will be transmitted and a closing flag. This last flag indicates that the transmission of the frame has ended. It can also be the leading flag of a new frame. By the way, the chip can support the sharing of a flag shared as trailing and leading flag by two frames. The level 2 bits are fed into level 1 and transmitted without change. So level 1 adds no extra data to the level 2 frame. The operations of level 1 and 2 will be explained in this report, level 3 can be found in the report of H.P.M.J. Schenkelaars. The level 3 report also explains the communication with a host-processor in an Intel or Motorola environment.

1.3 Chip outline

A decomposition of the X.25 protocol gives the different functions to be performed.

For the receiver these are:

frame in:	- handles the X.21 protocol procedures for physical link set-up.
frame disassembler:	- disasembles (de-multiplexes a frame into:

address
control
information if present
Frame Check Sequence
- handles data transparency.

Packet disassembler:
- disassembles a packet into GFID/LCGN (General Format Identifier/ Logical Channel Group Number)
LCN (Logic Channel Number)
PT (Packet Type)
Data for higher levels

(P)VC de-multiplexer:
- puts the data and additional housekeeping information of the various channels in their corresponding channel descriptors.
- generates housekeeping information to keep track of the state of that channel.

(P)VC control:
- location to put and read data and control information of a single channel (channel descriptor)

For the transmitting part of the co-processor these functions are:

(P)VC control:
- location to put and read data and control information about that channel (channel descriptor).

(P)VC multiplex:
- reads housekeeping information and data and initializes the corresponding registers in the packet-assemble module and handles the state of each channel

packet assembler:
- assembles the header of the packet by shifting the octets out of their registers, like
GFID/LCGN
LCN
PT
- fetches data out of memory to

transmit them

frame assembler:
- assembles a frame by shifting out the registers containing:
  flags
  address
  control byte
  information
- generate a FCS and shift it out.
- handles the transparency of the transmission. This is done by the zero-inserter

frame output:
- handles the X.21 protocol.

In figure 3 the functional decomposition is recognizable in the main architecture of the X.25 co-processor. The level 1 transmitter and receiver are clearly recognizable. The level 2 functions handled in the low level receiver and transmitter if they are simple enough. Controlling the standard functions of the low level 2 machines is done by high level 2. This is a microprogrammed controller which manages the logical link. Receiver and transmitter use the same microprocessor architecture within high level 2, and a set of communication registers to allow receiver/transmitter communication and concurrent program execution. The communication between low and high level 2 is through a number of 4 bit registers (bitwise accessible from low level 2 and 4 bit parallel from high level 2).

Data for level 3 is passed to the low level 3 receiver where the low level functions like demultiplexing are performed. Via a DMA unit the received data is stored in memory. The Bus interface handles the Intel or Motorola bus. The low level transmitting functions are similarly done by several machines in the low level 3 transmitter. Data and control information is coming from high level 3, and from the host. High level 3 controls the low level 3 machines, level 2 and lower. Data Handling and communication information in the main memory is an important part of the design and is discussed in the level 3 report.

Level 3 uses an 8 bit databus structure for internal communication with the low level 3 machines and the high level 3 provisions. For communication with the host system a 16 bit address and 16 bit data structure has been chosen, with a possible multiplex depending on the package. A 40

pens DIL package has more limits than a 64 pin grid package.

The high level 3 manager communicates with the host via a special command and response area in shared memory, and generates an attention signal to the host. The communication between level 2 and level 3 goes via 4 bit registers for level 3, accessed as 8 bit registers for level 3. Based on the wider databus of level 3. Two 4 bit registers for level 2 are seen as one 8 bit level 3 register. The data is exanged in these four bit as 3 bit data and 1 flagbit (indicating a change). The communication is strictly controlled to prevent a write action while the other processor is reading or writing. For the different flags, an attention is generated. This attention is checked and processed by high level 3 if its microprogram allows it. Beside high level 2 - high level 3 communication, there is also a command/status exange between high level 2 and low level 3 via a set of 4 bit registers. Level 3 communication will be discussed in chapter 6. Chapter 3 and 4 describes low/high level 2.

X.25 LOGICAL CHANNELS

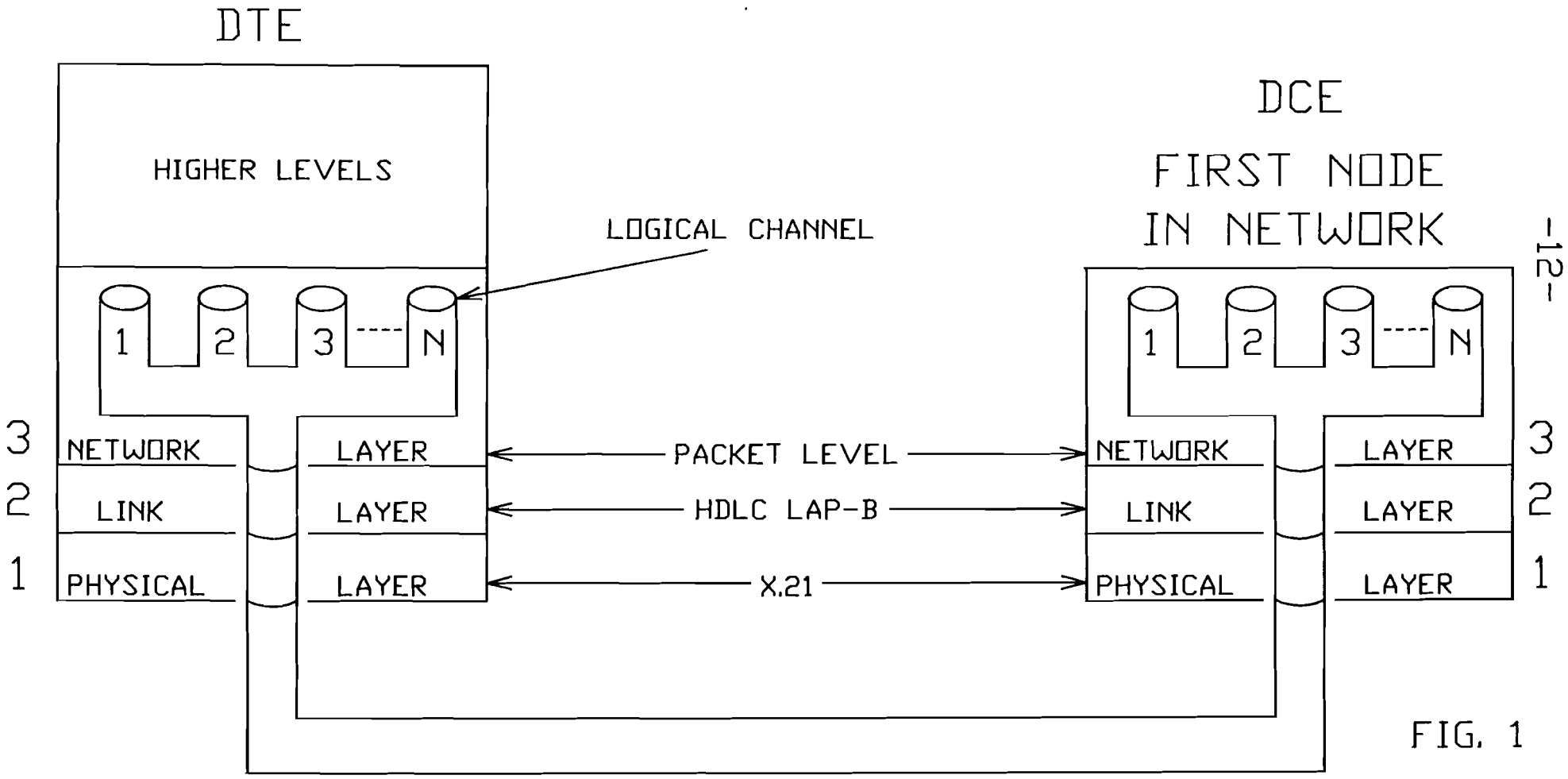


FIG. 1

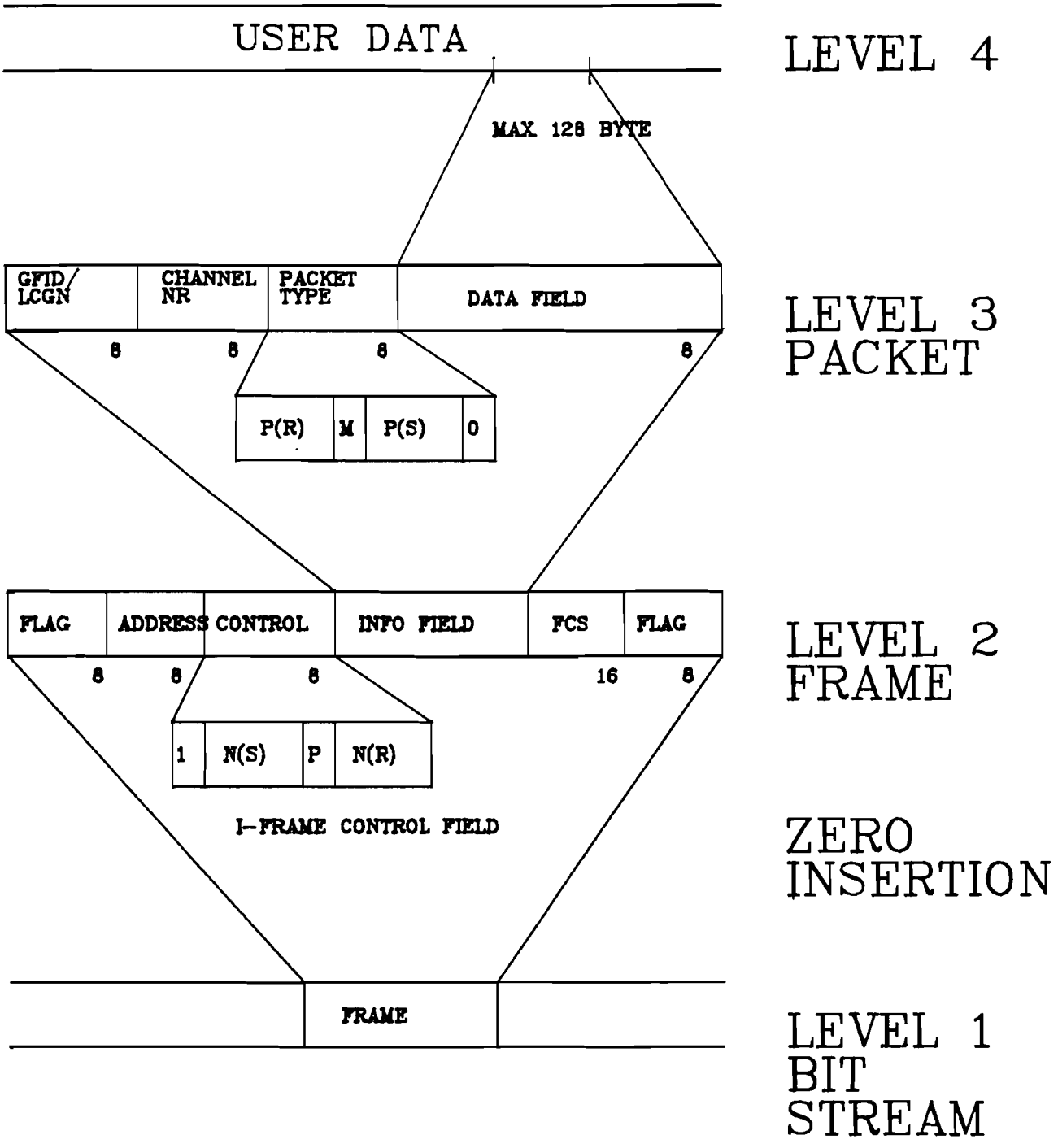

USER DATA
LEVEL 4
MAX 128 BYTE
GFID/ LCGN
CHANNEL NR
PACKET TYPE
DATA FIELD
8
8
8
8
LEVEL 3 PACKET
P(R)
M
P(S)
0
FLAG
ADDRESS
CONTROL
INFO FIELD
FCS
FLAG
8
8
8
16
8
LEVEL 2 FRAME
1
N(S)
P
N(R)
I-FRAME CONTROL FIELD
ZERO INSERTION
FRAME
LEVEL 1 BIT STREAM

FIG. 2

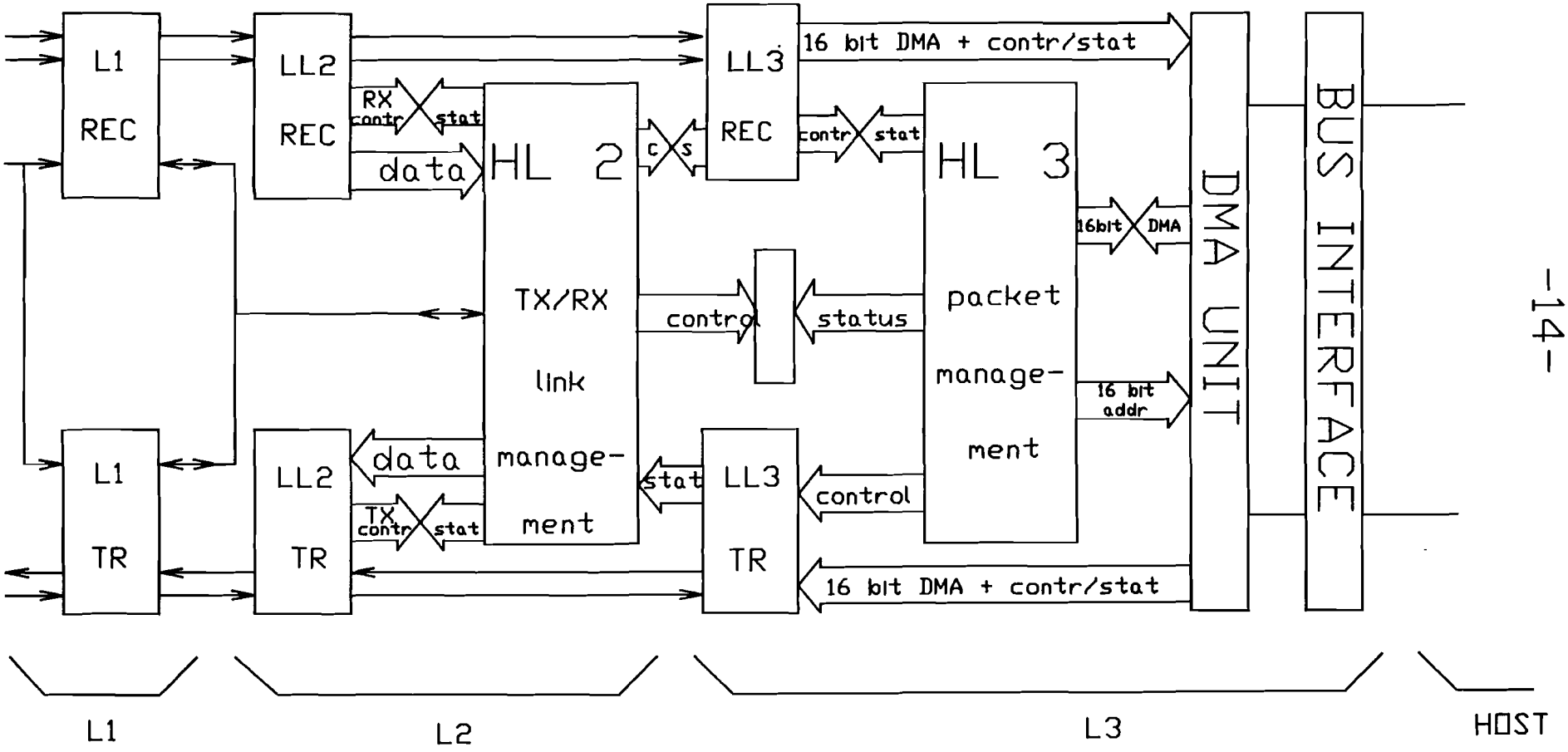

L1
REC
LL2
REC
RX
contr
stat
data
HL 2
C
S
LL3
REC
16 bit DMA + contr/stat
contr
stat
HL 3
16bit
DMA
DMA UNIT
BUS INTERFACE
TX/RX
link
manage-
ment
contrd
status
packet
manage-
ment
16 bit
addr
L1
TR
LL2
TR
data
TX
contr
stat
stat
LL3
TR
control
16 bit DMA + contr/stat
L1
L2
L3
HOST

FIG. 3

2 Level 1

Level 1 is designed according to the CCITT X.21 recommendation. This layer is the electrical interface between the network and level 2, the link layer. Level 1, the physical layer, is intended to send data from level 2 if the network is operational (ready). In the same way received data will be delivered to level 2.

Level 1 consists of four units. A receiver, a transmitter and a test-loop provision. The fourth part is a clock synchronizer. This piece of hardware has to sample the incoming clock (S) with a high sample rate. The outgoing clock signal will only change on an edge of the high frequency system clock, used on the chip. The S signal will be a 48 kHz signal typical, or at most 64 kHz. 48 kbit/sec is the highest bitrate allowed in the X.25 recommendation. The on chip clock will be about 20 MHz with high speed CMOS.

2.1 Level 1 states

We can distinguish four states in the level 1 DCE-DTE relation. See figure 4. The first state after power-on is state 22: DTE not ready and DCE not ready. State 24 is the situation in which the DTE is not ready and the DCE is ready. In state 18 is the DCE not ready and the DTE ready. The last state, state 1 is the state in which both DCE and DTE indicate that they are ready, and the data transfer state can be entered. As far as level 1 is concerned, there is no difference between a chip acting as DCE or as DTE. The only difference is a name for the "other side not ready" situation. This state can be called state 18 or state 24, but it is in fact the same state, only the opponent has another name.

In all states a signal is regarded as stable if it has no changes during 16 bittimes. A state has to be presented to the outside for at least 24 bittimes, if the state is to be considered stable.

2.2 Level 1 Transmitter

The level 1 transmitter is the chain between the level 2 transmitter and the network. The transmitter on level 1 has

to transfer the data from level 2 to the DCE if level 1 is enabled and the DCE signals READY (I is low and R is 1) or the I-line from the DCE is high (see figure 4). If level 1 is enabled it means that the higher levels want to make or maintain an electrical connection to the network. The DCE signalling ready makes clear that the network can start an electrical connection. If the I-line from the network is high then the bits on the incoming line, the R-line, are valid. Then connection is allowed too. The C-line must be set high in case of transmitting data (level 1 enabled), thus making the bits on the outgoing T-line valid to the DCE. If the DCE is not ready, information transfer is useless. This is further explained by the receiver of level 1. In case of not being enabled, level 1 transmitter must set its T-line 0 and its C-line low. This means that the chip is not ready for data acceptance. If the chip is ready for data acceptance but the network is not, the C-line must stay low but the T-line can be set to 1. The situation of a network not being ready is detected by the level 1 receiver and given to the level 1 transmitter by an enable-tx line. This is another enable line as the enable from level 2 and higher.

The four states mentioned can be brought back to three states for the transmitter. First of all the reset state, a combination of states 22 and 24 (DTE) or state 22 and 18 (DCE). In this state the tx signals uncontrolled not ready (C=off, T=0). If level 1 is enabled, state 18 is entered, in which case the tx signals ready. If the tx is enabled then, by the receiver, state 1 is reached. In this state the clock is transferred to the low level 2 transmitter and the data from this transmitter goes to the T-line. The C-line is on then. From all states the reset state is entered if level 1 is not enabled anymore. If the tx is not enabled, from state 1 state 18 is entered again.

(Note: The Dutch PTT only enables level 1 and then starts sending data to the DCE.)

Signals:

input: CLOCK, DATA, enable-tx, enable-level-1
output: CLOCK, C, T

Input data must be delivered to level 1 on the rising edge

of the outgoing clock. On the trailing edge of the clock the data has to be valid. These signals concern level 2 low. On the network side of level 1 the C-line must rise to high level on the leading edge of the clock, and the data on the T-line is allowed to change at the same moment. On the trailing edge the data has to be valid. The clock is given to high level 2, the incoming data is transferred to the T-line and the C-line is made high, if level 1 enable is true and tx enable from the receiver is true.

2.3 Level 1 Receiver

Received data bits are fed into level 2 if level 1 is enabled and the I-line is high (see figure 4). The enabling comes, as in case of the transmitter, from higher levels, and enables the electrical connection of the network signals to the chip. If the I-line is high then data coming from the DCE by the R-line is valid. It also confirms that the network is present and thus the transmitter is allowed to send data. The situation that the I-line is low and the R-line is low means that the DCE is not ready. In this case no bits are let through to the level 2 receiver.

Again from the four states mentioned above, three states can be distilled. The first state is entered after reset. As long as level 1 is not enabled by level 2, the receiver stays in this combination of states 22 and 24 (DTE) or states 22 and 18 (DCE). After the enable level 1 signal, state 18 is entered, if the DCE is not ready, the receiver stays in state 18, if the I-line goes "on", state 1 is entered and the received data is transferred to the low level 2 receiver. Level 1 tx is then also enabled. If the DCE signals ready, also state 1 is entered, but the data on the R-line is not transferred to the low level 2 receiver until the I-line is on.

If level 1 is not enabled, the first state is entered from state 18 as well as from state 1. If the DCE is not ready, or I is not on, state 18 is entered.

Signals:

input: CLOCK,R,I,enable-level-1
output: CLOCK,DATA,enable-tx

Incoming data can change on the leading edge of the clock and is valid at the trailing edge. The outgoing data must have the same characteristics. Enable-tx is only allowed to change on a leading edge. Similarly the enable-level-1 line.

The function of the receiver is also: connect the R-line to the level 2 Rx data entrance if the I-line is high for 16 bittimes and level 1 is enabled. Make enable Tx true if the I-line is high for 16 bittimes, or the I-line is low and the R-line is 1 for 16 bittimes.

2.4 Initialisation of level 1

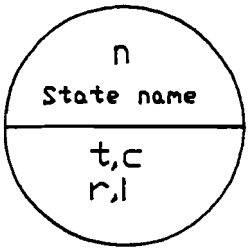
Level 1 can be enabled after the higher levels. First these higher levels have to be ready with their own initialization, or restart program. The enabling comes from high level 2, since this is the controlling part of the level just above level 1. There is a possibility to read the status of level 1 by clocking out the states of the receiver and transmitter. So far, there was no need for level 2 to know the status of level 1, but the host can be curious why level 2 is still in the state of trying to make a connection after a certain time. The state of level 1 can maybe be read with the same mechanism that has to test the chip, and is being developed by another student, H. van Ooijen. A provision that is present and can be used for test purposes is the level 1 testloop. In this testloop the I and C lines and the T and R lines are connected. See figure 6. In this way the transmitted data from the chip is fed back into the receiver of the chip. The C line to the DCE must be off, and the T line 0 in this case. A state of being not ready is so signalled to the network. In this way the whole chip can be tested, without the line drivers and buffers. As can be seen in figure 6 the circuit is quite simple, so this testloop is incorporated in the design.

According ANNEX A to Recommendation X.21

Interface signalling state diagrams

n State number
t Signal on T circuit
c Signal on C circuit
r Signal on R circuit
I Signal on I circuit

OFF Continuous OFF (binary 1)
ON Continuous ON (binary 0)



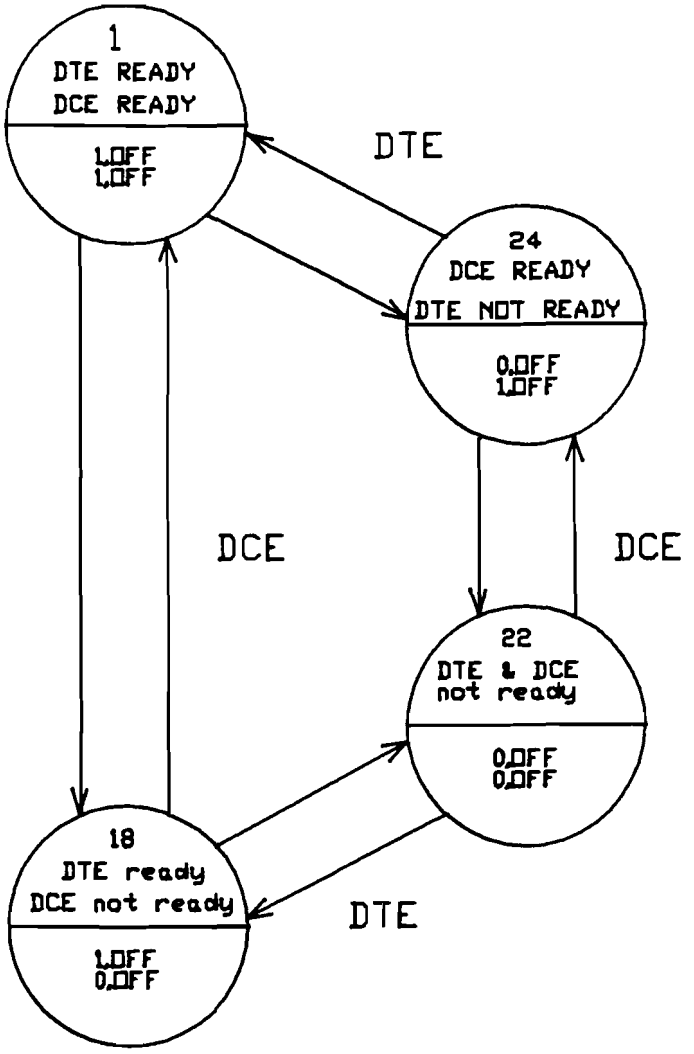


FIG. 4

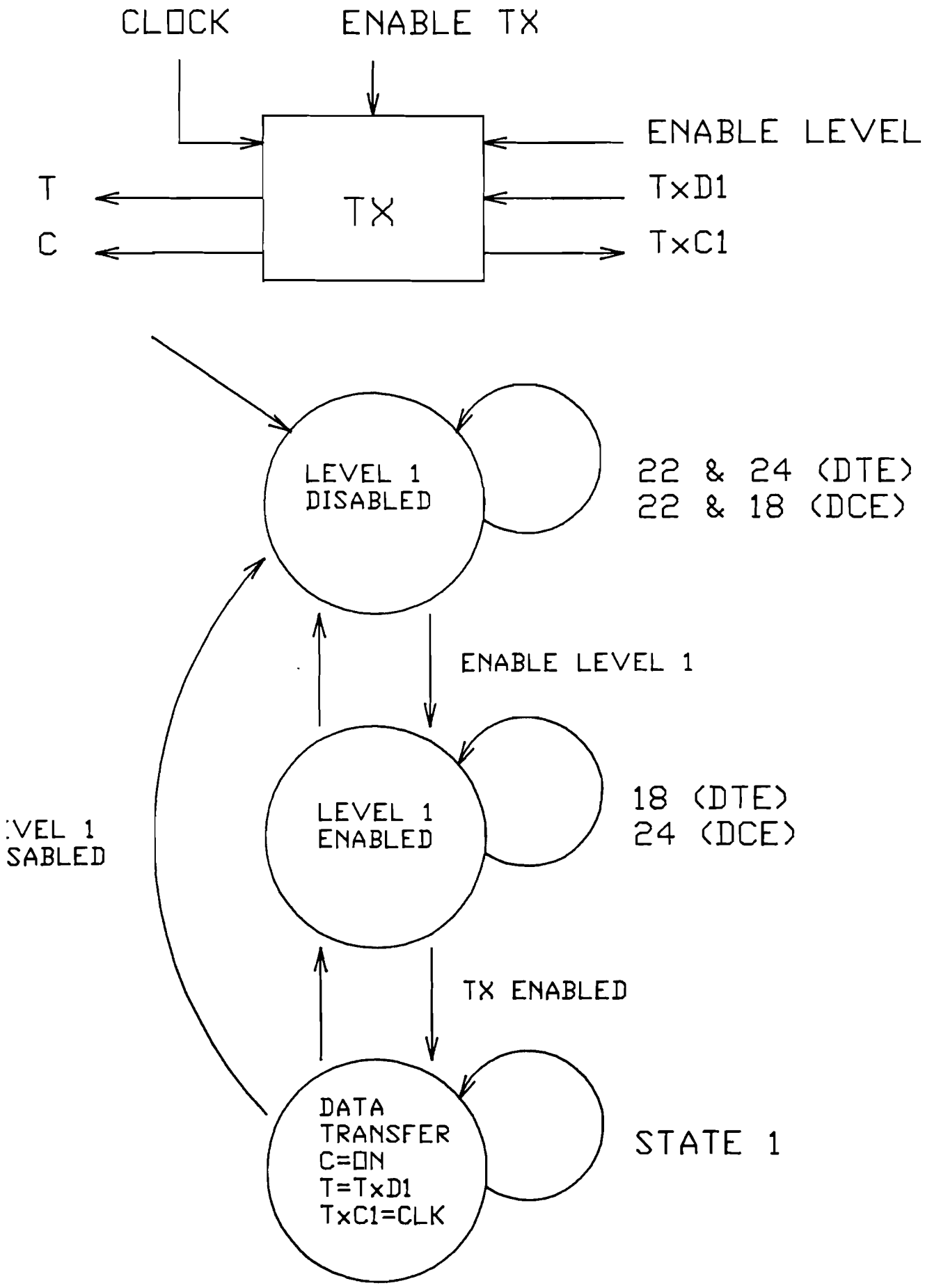

CLOCK
ENABLE TX
ENABLE LEVEL
T
TxD1
TX
C
TxC1
LEVEL 1
DISABLED
22 & 24 (DTE)
22 & 18 (DCE)
ENABLE LEVEL 1
:VEL 1
SABLED
LEVEL 1
ENABLED
18 (DTE)
24 (DCE)
TX ENABLED
DATA
TRANSFER
C=ON
T=TxD1
TxC1=CLK
STATE 1

FIG. 5

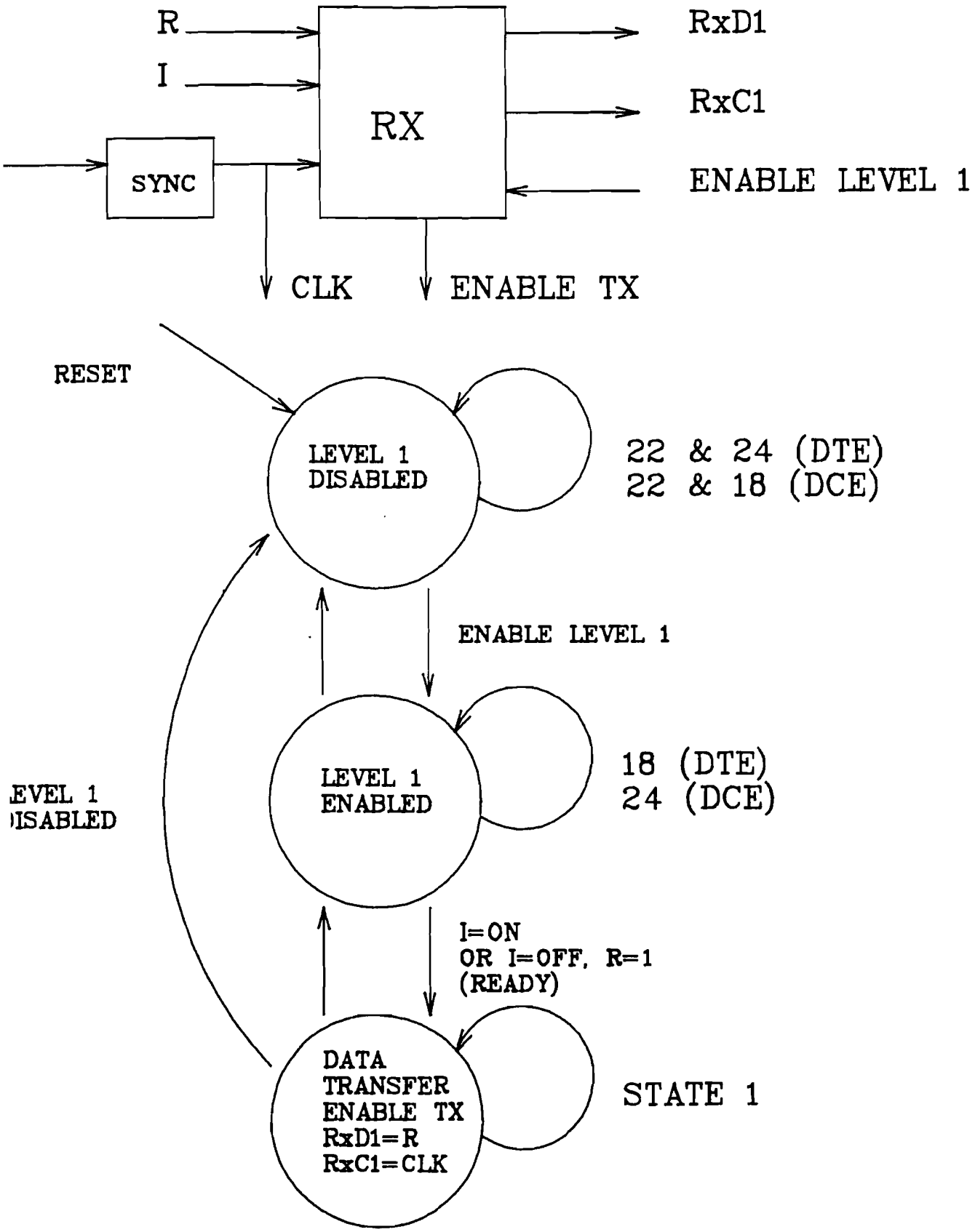
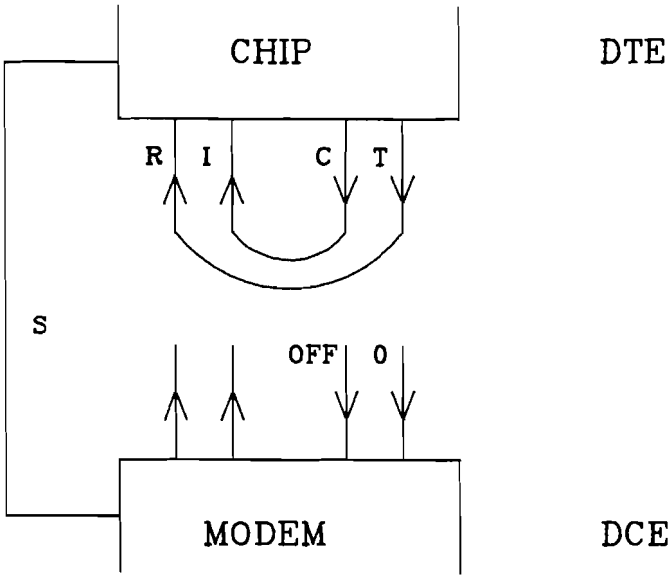
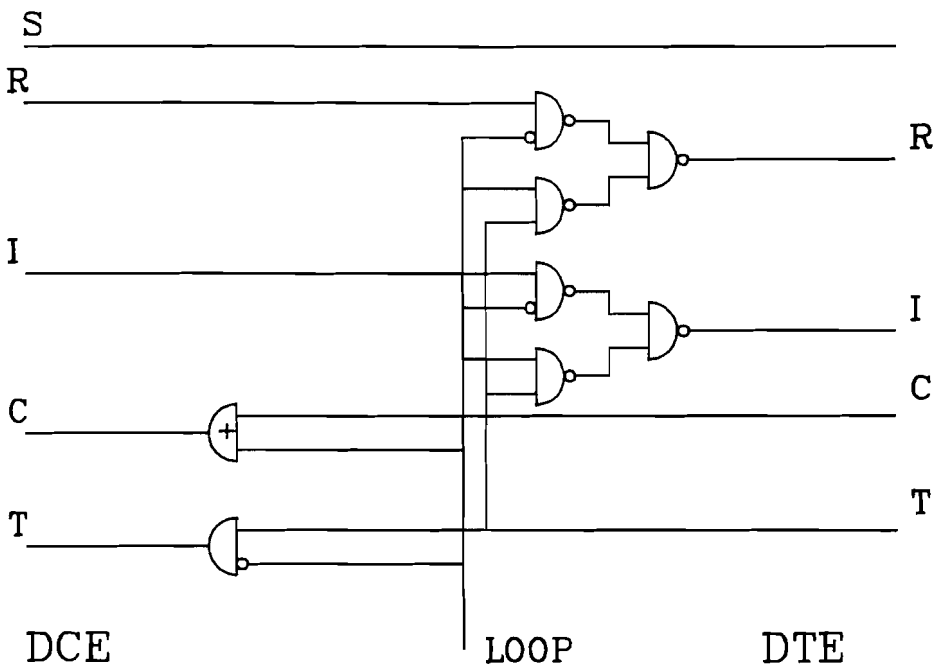

R
I
RX
RxD1
RxC1
ENABLE LEVEL 1
SYNC
CLK
ENABLE TX
RESET
LEVEL 1
DISABLED
22 & 24 (DTE)
22 & 18 (DCE)
ENABLE LEVEL 1
LEVEL 1
ENABLED
18 (DTE)
24 (DCE)
EVEL 1
ISABLED
I=ON
OR I=OFF, R=1
(READY)
DATA
TRANSFER
ENABLE TX
RxD1=R
RxC1=CLK
STATE 1

FIG. 6





Testloop 1

FIG. 7

3 Low level 2 transmitter

Level 2 is intended to set up, maintain and disconnect a logical link between a DCE and a DTE. We can distinguish higher and lower level functions. The more intelligent part is called high level 2, and controls the connection, while the more standard functions are called low level 2. The controlling of the link is the task of a microprogrammable controller. This controller is situated in high level 2. Low level 2 is separated in several dedicated task machines. The data manipulation parts and the controlling parts of the finite state machines in the low level transmitter are separated. See figure 8.

Briefly all low level 2 transmitter functions:

- Transmit flags on request of high level 2.
- Transmit on request an information, unnumbered or supervisory frame.
- Transmit on request an FRMR frame.
- Support the transmission of frames with the automatic insertion of a zero after five ones.
- Generate automatically a Frame Check Sequence for the frames being transmitted.
- Transmit on request an ABORT sequence.
- Transmit on request an IDLE sequence.
- Support the transmission of frames with a leading flag, an address byte and a command byte, the latter delivered from high level 2. The same with the FCS and the trailing flag.

Signals:

input: DATA from level 3,
CLOCK from level 1,
Commands from high level 2:

FLAG/NON_TX_FRAME, ABORT, IDLE, FRMR, PACK_END,
****(RESET)****
ADDRESS, CONTROL, FRMR bytes.

output: DATA to level 1,
CLOCK, ENABLE to level 3,
CLOCK, ENABLE to high level 2,
Status to high level 2: ready

3.1 Pattern generation

Information from level 3, or link control data from high level 2 must be transmitted in a standard frame envelop. At the start a flag: 0111 1110. This flag is generated at command of the low level 2 tx manager, by the flag generator. This finite state machine also generates an abort (111 1111): seven ones, or an idle (1111 1111 1111 1111 111), fifteen ones, if this is requested.

An abort or idle command from high level 2 is followed by an abort or idle sequence consisting of ones, generated by the flag/abort/idle generator. Frames being transmitted are aborted then. If the low level transmitter is ready with the transmission of a single flag, abort or idle, or of a whole frame, the ready state is reported to high level 2 via the ready bit register.

The pattern generator receives the clock from level 1 and gives it to the zero inserter control if the pattern generator is not busy transmitting any sequences. If the generator receives the flag, idle or abort command, it makes the outgoing MUX1 signal high, so the multiplexer selects the pattern generator as source. This same signal disables the other level 2 machines. The DATA line is then fed with the required patterns.If the pattern generator has done its job, it sets its status line high, meaning it is ready for a next command from the Tx manager.

3.2 Zero insertion

During transmission a sequence of five ones is automatically interrupted and a zero is inserted. The data stream then goes on until a next sequence of five ones. The object of this zero insertion is to prevent flag, abort or idle imitation of data transmitted between two flags. This could disturb the correctness of the receiving process. The zero

inserter transfers the incoming data from line TXD2 to the outgoing dataline TXD2A, if the ZERO INSERT line is low. If this line is high the TXD2A line is made zero. The data from line TXD2 is also transferred to the zero inserter control, for analysis.

The zero inserter control counts the number of clock periods the data on line TXD2 is one, and makes after five ones the zero insert line true. The zero insert control starts its work from zero if the MUX1 line goes low. A reset takes place if a zero is inserted, or a zero in the data stream occurs and on th trailing edge of the MUX1 signal. The clock is not transferred to the FCS control if a zero is being inserted, or if the MUX1 line is high.

3.3 FCS generation

The FCS control receives the clock from the zero inserter control and transfers it to the Tx manager. from this Tx manager it receives the signals flag, and frame end. On the trailing edge of the flag signal the FCS control generates a reset or preload pulse for the FCS generator. On the leading edge of a frame end signal, the FCS control makes the Calc/non-FCS line low. The clock is also delivered to the FCS generator.

The Frame check sequence generated by the FCS generator is a pattern that is left in a 15 bit register after the division of the transmitted data by a so-called generator polynom. This modulo two division is done in hardware. The generator is drawn in figure 9. The generator polynom is according the CCITT recommendations x15 + x12 + x5 + 1. The remainder of the division is transmitted with the x15 component first, and lower bits after that. The registers in fig. 9 are preloaded with ones, on the reset pulse from FCS control. The data comes via two ports in the registers, and the contents of the registers can be shifted out to the zero inserter by making the Calc/not FCS line low. The data is valid on the trailing edges of the clock, and is allowed to change on the leading edges. The flipflops of the shiftregister have to be master slave types.

3.4 Transmit manager

After the leading flag, an address is transmitted. The tx manager provides the necessary enabling signals to the high level 2 registers involved. Following the address byte, a

control byte is transmitted. To simplify the design the first control bit is transferred to the last address register bit. The registers concerned are simple shift registers and so an enable signal only has to enable a shift operation, in the rhythm of a clock supplied by level 1. If a FRMR frame has to be transmitted, the enable to the shift register lasts as long as necessary (5 bytes extra). After a leading flag, an address and a control byte, the data from level 3 is enabled to flow in the case of an information frame. The end of the frame is reported by the frame-end bit register. This frame-end command initiates the tx manager to disable level 3 and the registers of level 2 (after finishing the current control or FRMR information transmission. Then the tx manager starts the frame check sequence generator, by making the frame end signal high. This FCS generator is fed with all the bits transmitted after the leading flag and this datastream is modulo 2 divided by a polynom. The remainder of the value is then transmitted before the trailing flag. The trailing flag transmission is again initiated by the tx manager which starts the flag generator. The ending of the flag signal is also a signal for the pattern generator to preload the generator, because a frame can be transmitted after the flag. If the Tx manager is ready with the commanded actions, it reports ready in the STATUS TX register to high level 2.

3.5 Low level 2 tx/ high level 2 tx communication

See figure 10 for an outline of the registers concerned.

COMMAND:	ACTION:
ABORT	Current transmission must be aborted. Send seven ones. If there is no other command following, send flags. Report ready after the abort sequence is sent.
IDLE	Current transmission must be aborted. Send fifteen ones. IDLE can be continued or a next command or flags can follow. Report ready after the idle signal is completed.
FLAGS	Send continuous flags (0111 1110).
Non-TX-FRAME	Send a flag sequence 8 bits

	Send an address 8 bits from high level 2
	Send a control byte 8 bits from high level 2
	Enable the DMA controller to deliver data from level 3 and the host. Wait until the FRAME-END signal.
FRAME-END	Send the FCS 16 bits Send a flag 8 bits If there is another Non-TX-FRAME send the next frame. Otherwise transmit flags, or idle, or abort. Report the finishing of a frame by "ready".
FRMR & Non-TX-FRAME	Send flag 8 bits Send address 8 bits Send control 8 bits Send FRMR information 24 bits Send FCS 8 bits Send flag 8 bits
RESET	Start from scratch. Everything is reset as if a power-up has occurred.

Support:

Automatic zero insertion in all data sequences between an opening and a trailing flag. This is done by the zero inserter unit

Combined flag, abort and idle generation. The pattern generator which makes this sequences is located after the zero inserter because this patterns are not allowed to be disturbed by zeros. These patterns have to be unique.

FCS generator for generating the error detecting code. All bits between the two enclosing flags of a frame is fed through this unit. After the last bit of a frame the contents of the generator is sent before the closing flag.

Address byte, delivered by high level 2 in two 4 bits shift registers.

Control byte, delivered by high level 2 in two 4 bits shift

registers. These registers are connected to the address the registers. In case of shifting the data out, the control data byte goes via the address registers to the low level 2 transmitter.

FRMR registers, containing the frame reject information. These register are connected to the control byte register. The FRMR information is transmitted by shifting via the address and control registers.

Tx-manager for the data flow control and the execution of level 2 high commands. This manager selects the data sources, and starts the different low level 2 machines.

3.6 Initialization

The low level 2 transmitter is reset by power up, and can be software reset by the reset register in the COMMAND TX 2 register. After the power-up reset or software reset simply the FLAG command has to be given to start the logical connection to the DCE by transmitting flags. The rest of the link set-up and so on can be directed by the high level 2 transmitter.

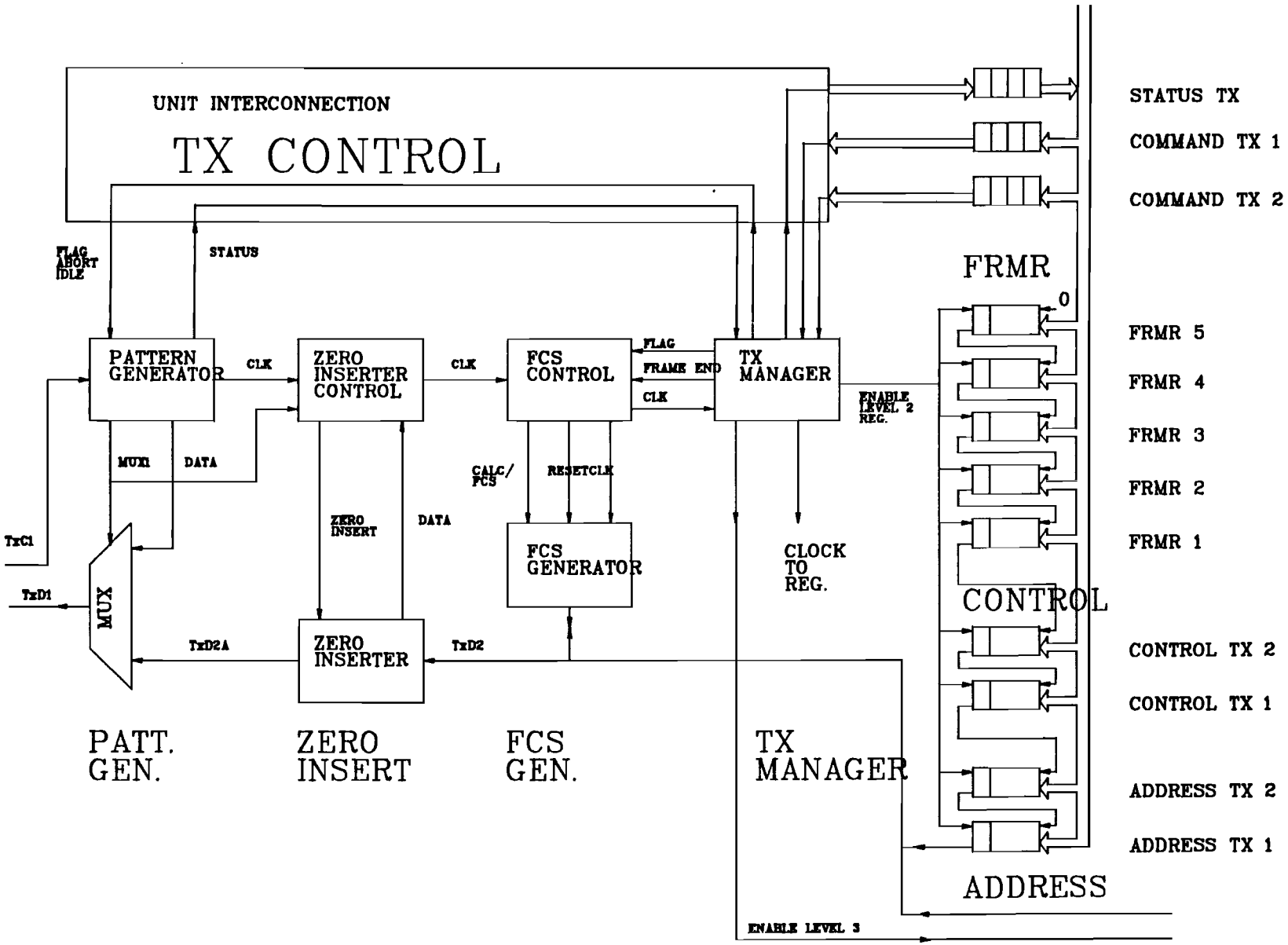
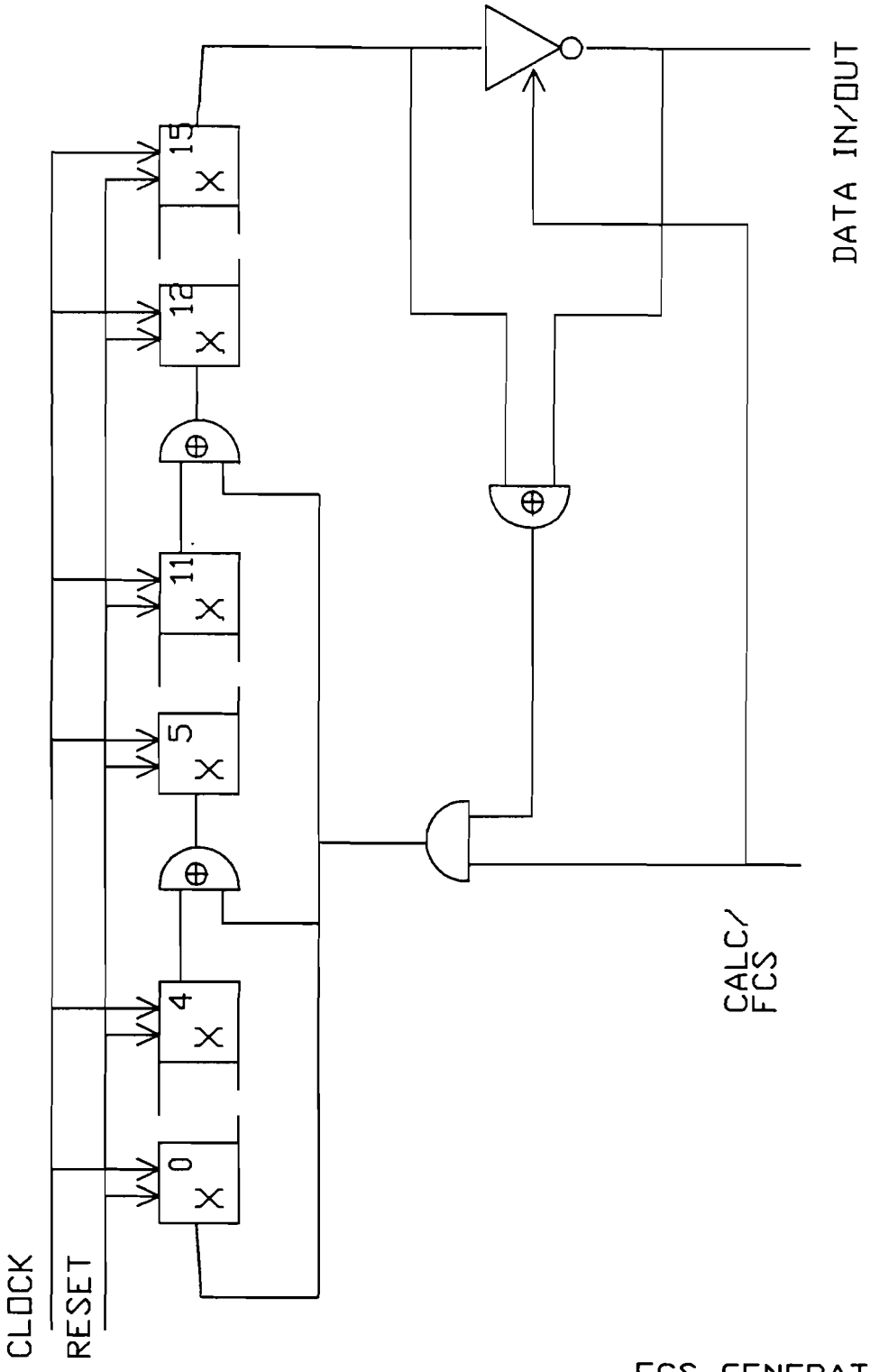

UNIT INTERCONNECTION
TX CONTROL
STATUS TX
COMMAND TX 1
COMMAND TX 2
FLAG
ABORT
IDLE
STATUS
FRMR
0
PATTERN GENERATOR
CLK
ZERO INSERTER CONTROL
CLK
FCS CONTROL
FLAG
FRAME END
CLK
TX MANAGER
ENABLE LEVEL 2 REG.
FRMR 5
FRMR 4
FRMR 3
FRMR 2
FRMR 1
MUXI
DATA
CALC/ FCS
RESETCLK
ZERO INSERT
DATA
TxC1
TxD1
MUX
FCS GENERATOR
CLOCK TO REG.
CONTROL
CONTROL TX 2
CONTROL TX 1
TxD2A
ZERO INSERTER
TxD2
ADDRESS TX 2
ADDRESS TX 1
PATT. GEN.
ZERO INSERT
FCS GEN.
TX MANAGER
ADDRESS
ENABLE LEVEL 3

FIG. 8



FCS GENERATION

FIG. 9

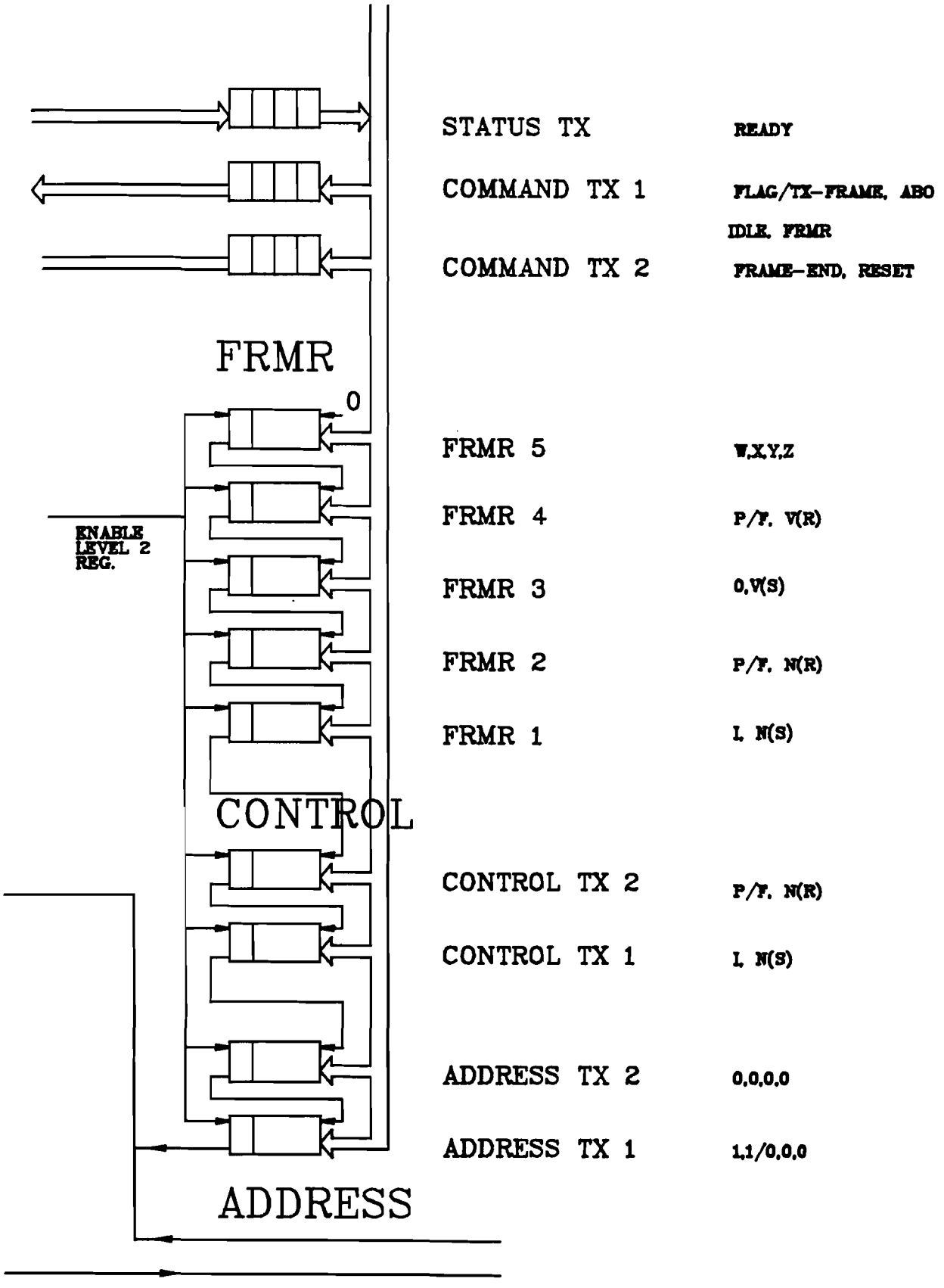

STATUS TX
READY
COMMAND TX 1
FLAG/TX-FRAME, ABO
IDLE, FRMR
COMMAND TX 2
FRAME-END, RESET
FRMR
0
FRMR 5
W,X,Y,Z
FRMR 4
P/F, V(R)
ENABLE
LEVEL 2
REG.
FRMR 3
0,V(S)
FRMR 2
P/F, N(R)
FRMR 1
I, N(S)
CONTROL
CONTROL TX 2
P/F, N(R)
CONTROL TX 1
I, N(S)
ADDRESS TX 2
0,0,0,0
ADDRESS TX 1
1,1/0,0,0
ADDRESS

FIG. 10

4 Low level 2 receiver

The low level receiver handles the standard envelop of packets and supervisory or unnumbered information. Low level 2 supplies high level 2 with all the needed standard information for the link-control. The frames are arriving from the level 1 receiver, and partly go into level 3, the DMA controller, and the high level 2 controller. The data manipulation part and the controlling part of the finite state machines are separated as can be seen in figure 11. The other parts of the low level 2 receiver are an 8 bit shift register on the front end and a 16 bit shift register on the border of level 2 and 3. The 8 bits shift register prevents the trailing flag of a frame from going into the zero deleter and the frame check sequence checker. The 16 bits shift register captures the 16 bits of the frame check sequence before level 3 gets this information. The reason for this "delay registers" is that only when the trailing flag is fully received, the preceding 8 bits can be said to be the flag and the 16 bits before these, the frame check sequence.

A report by J.M.H.M. van Kessel describes the functional tests with TDU of the low level receiver.

High level 2 can tell the low level receiver not to send any data by giving a busy signal via the command rx register.

Briefly all low level 2 receiver functions:

- Detect a flag (0111 1110) and signal the high level 2 controller.
- Detect and decode A/B address and deliver this to high level 2.
- Catch the control byte in a register and give this to the high level 2 microcontroller.
- Supply a valid signal for the address and control information bytes.
- Support the analysis of the frame check sequence by a 8 bit delay line. This prevents the trailing flag going into the FCS checker.

- Prevent that the frame check sequence goes into level 3. This is done by a 16 bit delay line.

- Check the frame check sequence and signal the correctness of the data to high level 2.

- Give a Frame end signal to the microcontroller.

- Detect and signal an IDLE situation.

- Detect and signal an ABORT situation.

- Delete the during transmission from the DCE inserted extra zeros.

- Check the number of bits after the begin flag:
  <32: send a FRMR frame.
  >32: OK, for a I-frame.
  = 32 OK, for all frames, in case of an I-frame: signal level 3 the arrival of a zero information field.
  >N2 bits. More than the maximum number of bits allowed are counted after the leading flag.

Signals:

input: DATA, CLOCK from level 1
status from high level 2: busy

output: DATA, CLOCK*, enable to level 3
control byte to level 2 in 2 registers, four bits wide.
Status to level 2:
A-ADDRESS, B-ADDRESS, ADDRESS-VALID, FLAG, FRAME END, FRAME-OK, CONTROL-VALID, IDLE, ABORT.
FRAME LENGTH:
<32
=32
to long.

4.1 Pattern recognition

A frame delivered from level 1 to level 2 has a general form. The frame begins with a flag sequence of 01111110. The

flag synchronizes the receiver to start a standard procedure of decomposition of the frame. After that an address is following: 11000000 or 10000000 depending on the destination being a DCE or DTE and the frame being a command or response frame. The beginning of a frame must be signalled by the low level 2 receiver and the address must be recognized, accepted and delivered to the high level 2 receiver. This recognition is quite simple and can be done by a finite state machine, called the pattern recognizer. Two patterns more can occur which are important to be signalized. These patterns are 1111111 (seven ones) and 1111111 11111111 (fifteen ones), called abort and idle. These patterns, once recognized, must be signalled to the high level 2 receiver. Abort means: stop the current frame consuming and start again with flag hunting. Idle indicates a state of non presence of any information on the line. The end of a frame is also recognized and signalled to high level 2 by the pattern recognition unit.

The leading flag is received in the 8 bit shiftregister, and the flag pattern is recognized and reported to high level 2. The shiftregister is clocked by the trailing edge of the clock, and shifted by the leading edge of the clock. The other patterns are also received and recognized in the 8 bit shiftregister, but the main reason for its presence in the design is that it allows us to destroy the leading flag before the data over which a frame check sequence is generated is checked by the frame check sequence checker. The pattern detector enables after the leading flag is clocked out, and an A, B, C or D address is recognized, the zerodeleter and following machines.

The pattern detector receives the data and clock from the level 1 receiver and gives the flag, abort, idle, A or B address, attention, address valid and frame end reports to high level 2. A reset line resets the pattern recognizer to a state in which it goes flag hunting again. After a flag and a non flag bit pattern a next flag generates a frame end signal.

4.2 Zero deletion

The zero deletion machine removes the extra zeros inserted by the transmitter. These zeros are inserted to prevent flag or abort imitation of the transmitted data. After every sequence of five ones a zero is inserted to the data stream. The zero deleter controller compares the incoming data with a one and counts until 5 consecutive ones are counted

without a single zero in between. Than a signal goes to the zero deleter to make the next data bit one. The controller itself skips one clock cycle and so deletes the zero following the 5 ones. Because the zero deletion only works with data without the flag enclosure, the flag is not disturbed by this unit. The unit only works if a flag is recognized by the pattern recognizer and is followed by an address. The zerodeleter gives the enable line unchanged to the FCS check control. The data and clock lines are influenced as described above.

4.3 FCS checking

After the data, a 16 bit long error detecting code is added. This so-called frame check sequence is checked on being correct in the frame check sequence checker. The correctness is told to high level 2 via the rx status registers. The FCS check control starts if the enable line goes high with the preload of the FCS checker with ones and supplying the clock to the checker. The Rx manager is supplied with clock and enable, as is the Bit counter. The FCS checker is drawn in figure 12. The reset from the FCS check control preloads the unit with ones and the clock and data input perform the modulo 2 division as described in chapter 3. The remainder of this division is continuously monitorred and results in a high status signal if the remainder is F0B8. The FCS check control samples this status and makes a frame ok report to the STATUS RX 1 register, on the reception of a leading edge of the frame end signal. The frame end signal comes from the pattern handler. An attention signal for the high level 2 receiver is generated and set in a high level 2 register. The detection of an error in the FCS detects all single bit errors, all double bit errors, all even number of errors, all bursts =< 16 bits, 99.997% of the 17 bits bursts and 99.998% of the bursts >= 18 bits. This is only detection! the correction of an error is done by retransmission, or link reset. The FCS checking by the modulo 2 division is according the polynom x15 + x12 + x5 + 1, as used in the FCS generator (see chapter 3).

4.4 Bit counter

The bit counter is started by the enable signal and counts the bits between the leading and trailing flag. The reset line resets the counter. The leading edge of the enable line does also. The bit counter counts the frame bits and gives the following signals to high level 2 via the STATUS RX 3

register:

< 32	The frame is without flags shorter than 32 bit, and not a valid frame.
= 32	The frame is 32 bit long between the flags, and so it is a valid frame. In the case of a I frame this fact of being 32 bit long is to be told to level 3 because the I frame contains no data then.
too long	The frame is longer than N1 bits. This N1 value is read in to the counter during initialisation. The frame is invalid then. The N1 value is continuously compared to the current counter value by a comparator. The most used value is 128 octets, 1024 bits.

The bit counter is only using the clock and data information, and does no operations on them.

4.5 Receive manager

After the flag and address bytes there normally follows a control byte. This control byte must be given to the high level 2 part via 2 registers, each 4 bits wide. The flag and address bytes must be removed before this can happen. This is the task of the low level 2 rx manager. This simple finite state machine delivers enable-signals to the different parts of the chip after the zero deletion machine. The machine is started after the leading edge of the enable signal. The 8 bits of the address are deleted by not enabling any registers after the rx manager for the first 8 clock cycles after the leading edge of the enable signal. The 8 bits of the control word are fed into the STATUS RX 1 and STATUS RX 2 registers by an enable signal after the address is deleted. The rx manager enables after the 8 bits of the control byte the 16 bit shift register and when this register is full, enables level 3. The received bits are then going to the low level 3 receiver for analysis and to the DMA controller for putting them in the host's memory. When the frame end signal comes from the pattern recognition unit, all lines become disabled. The 16 bit shift register deletes in this way the 16 FCS because the delay of 16 bits before level 3 is reached catches the FCS and something not transferring is deleting it. The reception of the control

byte is reported to high level 2 by generating a control valid report via STATUS RX 1 register and an attention. The signal busy from high level 2 (the reset in the COMMAND RX register) resets the RX manager and the controlling is started again if the pattern recognizer has seen a flag and an address. If the reset line stays high, no flag hunting is started as is described at the part discussing the pattern recognizer. The rx manager distributes clock information to the control shift register, the 16 bit shift register between level 2 and 3 and to level 3.

4.6 Low level 2 rx/ high level 2 rx communication

COMMAND:	ACTION:
BUSY	Stop the data stream to level 3 and start again with flag hunting. This is more or less a reset command.

PATTERN RECEIVED:	ACTION:
Flag	Set FLAG bit in the STATUS OUT register and go flag hunting.
Abort	Set the ABORT bit in the STATUS OUT register and go flag hunting.
A address	If this pattern is following after a flag then set the A_ADDRESS bit in the STATUS OUT register. Set the ADDRESS-VALID bit in the STATUS OUT register.
B address	If this pattern is following a flag then set the B-ADDRESS bit in the STATUS OUT register. Set the ADDRESS-VALID bit in the STATUS OUT register.
C or D address	In the case of a multilink control these addresses are also valid.
Control byte	This is the third byte received after the flag and is fed into a shift register that is accessible from high level 2. If all 8 bits are received the CONTROL-VALID bit is

set in the STATUS OUT register.

Data — Data for level 3 will be send to level 3 via the 16 bit shift register.

Closing flag — If the machine is transferring data to level 3 or diagnostic information then a received flag is the closing flag of the frame and the FRAME-END bit is set in the STATUS OUT register. The data transfer is then stopped.

FCS — After stopping the data transfer the FCS code is catched in the 16 bit shift register. The FCS code is checked in the FCS checker. This FCS checker is then read, and the bit "FCS-OK" is then set.

<32 bit — A bit sequence consisting of less than 32 bits is invalid and must be reported to high level 2

=32 bit — A bit sequence of 32 bits is required for several frames, so this must also be reported.

too long — If a frame bit sequence is longer than N2 bits, the frame is invalid and this must be signalled also.

After the closing flag an attention signal is generated for High Level 2 if the status FCS-OK is set (or left zero if an error occurred).

Support:

Automatic zero insertion from all data sequences between an opening and a trailing flag. This is done by the zero deleter unit.

Combined flag, abort, idle A and B address recognition, done by the pattern recognizer. Automatic flag removing. The unit is positioned before the zero deleter because after the zerodeleter, the possibility of flag imitation by the data received arises.

FCS checker for checking the error detecting code. All bits between the two enclosing flags of a frame are fed through

this unit. After the trailing flag, the remaining code in the checker is an acceptance of the frame or not.

Control byte register, for shifting in the control byte received.

Rx manager for enabling the registers and delivering several signals

A 16 bit shift register between level 2 and level 3.

A bit counter for the several frame lengths, and the maximum frame length checking.

4.7 Low level 2 rx initialization

The only action needed for high level 2 is the resetting of the low level 2 receiver. This is done by making the reset bit in the COMMAND RX register high. All the further initialisation is done by the low level 2 receiver itself. The reset bit has to be made low again by high level 2. The only further needed information is delivered by level 3, namely the N1 value for the bit counter. High level 2 can interrupt the receiver by making the reset bit high for as long as the high level 2 receiver is busy. The link set-up and so on is a task of high level 2. Reception of flags is reported via STATUS RX 2 and is a signal for high level 2 that the logical link is possible.

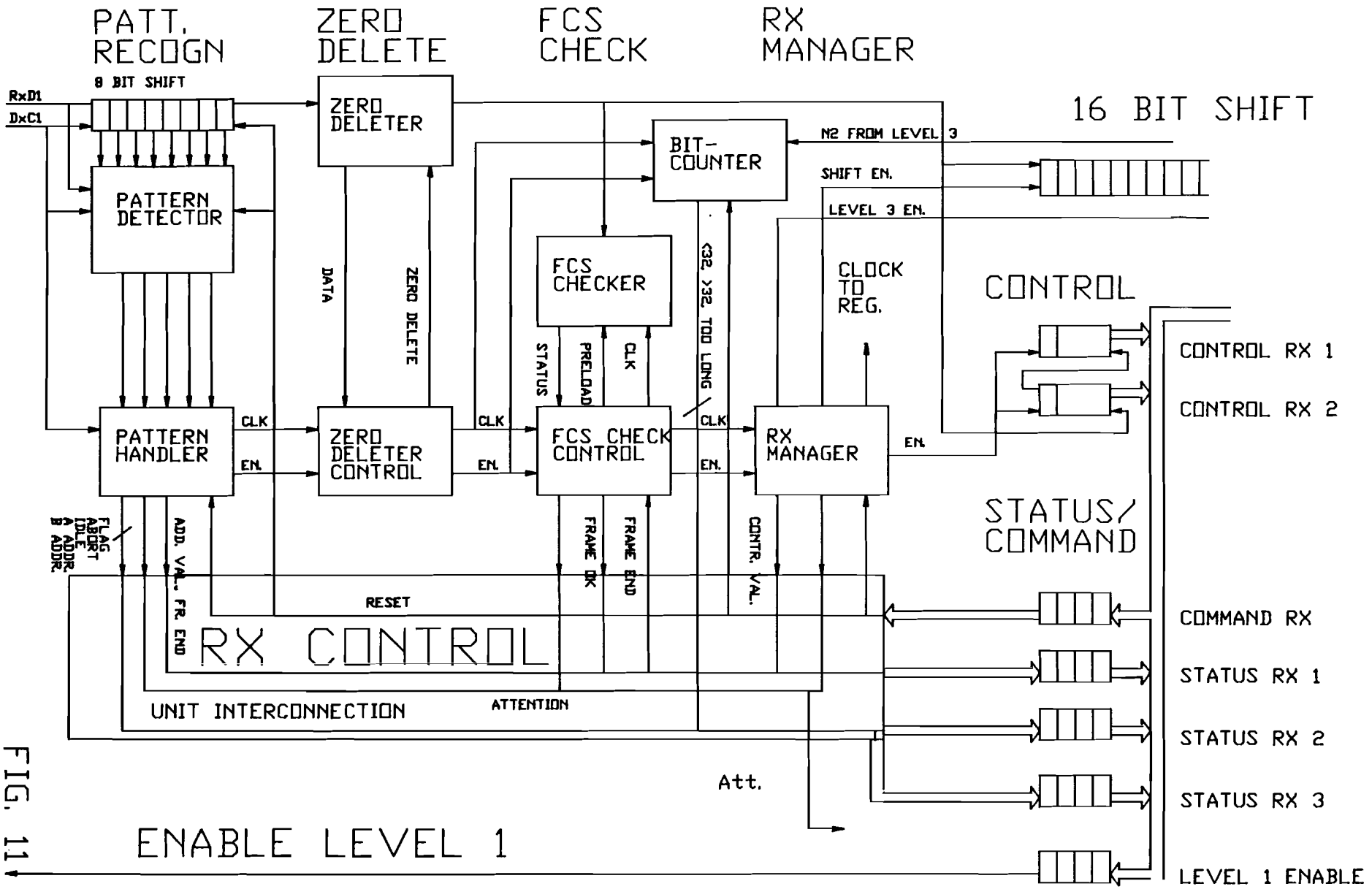

PATT. RECOGN
ZERO DELETE
FCS CHECK
RX MANAGER
8 BIT SHIFT
16 BIT SHIFT
RxD1
DxC1
ZERO DELETER
BIT-COUNTER
N2 FROM LEVEL 3
SHIFT EN.
LEVEL 3 EN.
PATTERN DETECTOR
FCS CHECKER
CLOCK TO REG.
CONTROL
DATA
ZERO DELETE
STATUS
PRELOAD
CLK
<32, >32, TOO LONG
CONTROL RX 1
CONTROL RX 2
PATTERN HANDLER
CLK
EN.
ZERO DELETER CONTROL
FCS CHECK CONTROL
RX MANAGER
STATUS/ COMMAND
FLAG
ABORT
IDLE
A ADDR.
B ADDR.
ADD. VAL.
FR. END
FRAME OK
FRAME END
CONTR. VAL.
RESET
RX CONTROL
COMMAND RX
STATUS RX 1
STATUS RX 2
STATUS RX 3
UNIT INTERCONNECTION
ATTENTION
Att.
ENABLE LEVEL 1
LEVEL 1 ENABLE

FIG. 11

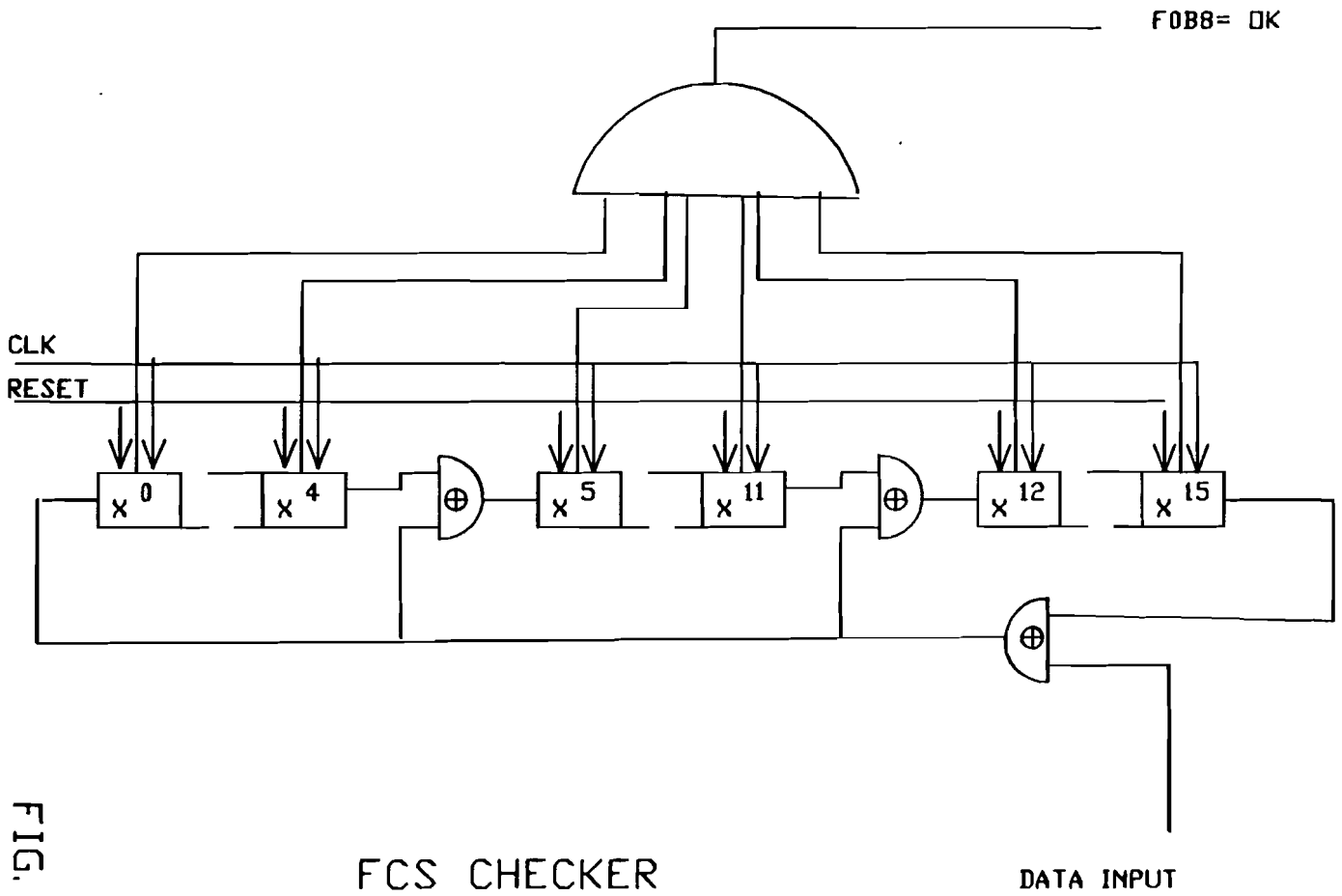


FIG. 12

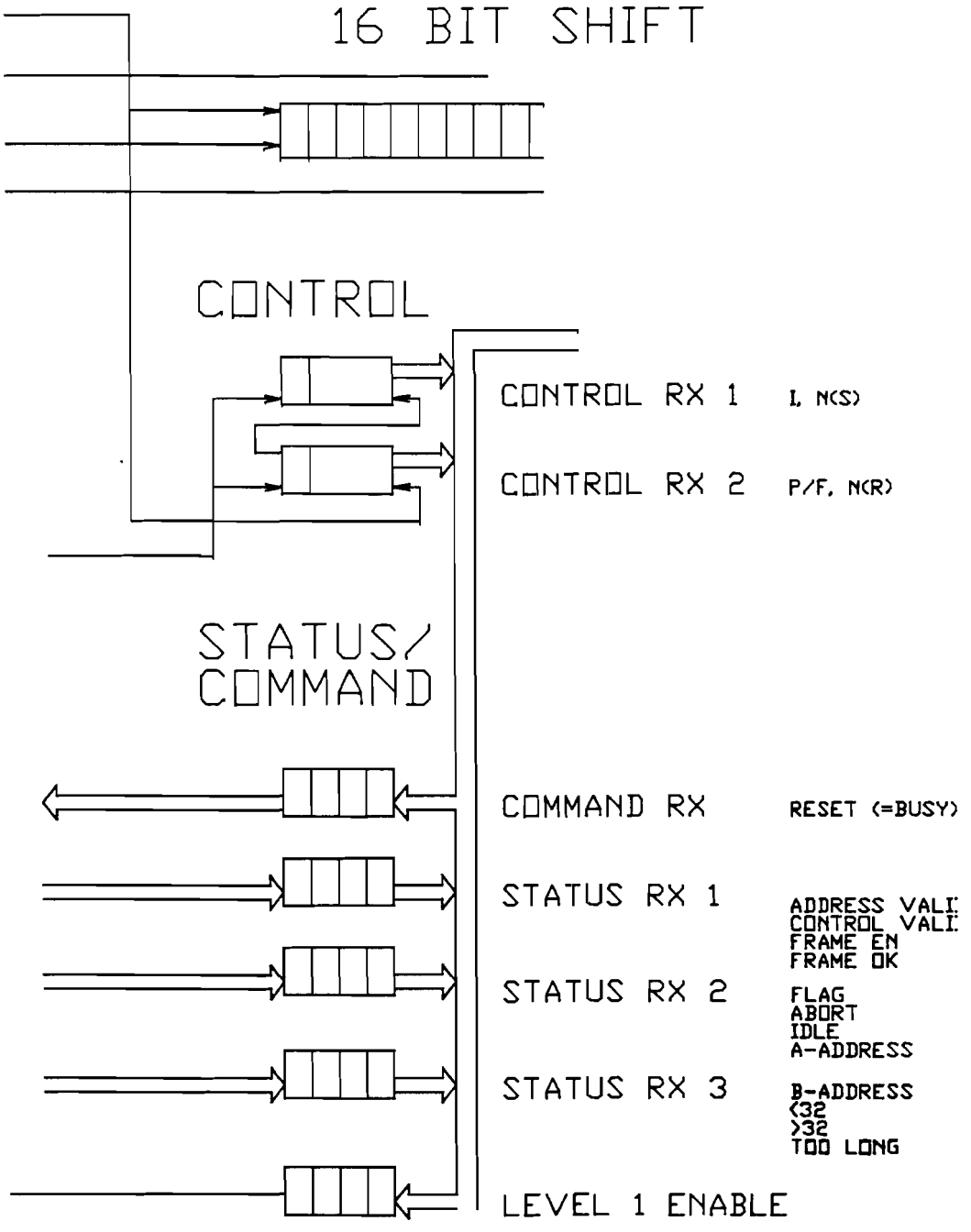

16 BIT SHIFT
CONTROL
CONTROL RX 1
I, N(S)
CONTROL RX 2
P/F, N(R)
STATUS/
COMMAND
COMMAND RX
RESET (=BUSY)
STATUS RX 1
ADDRESS VALI
CONTROL VALI
FRAME EN
FRAME OK
STATUS RX 2
FLAG
ABORT
IDLE
A-ADDRESS
STATUS RX 3
B-ADDRESS
<32
>32
TOO LONG
LEVEL 1 ENABLE

FIG. 13

5 High level 2

Level 2 offers to level 3 the logical connection between DCE and DTE. The link is set up and maintained by level 2. For level 3, level 2 is transparent. Only in special conditions there is an exchange of information (apart from the normal information necessary for transmitting and receiving level 3 packets). The packets of level 3 are packed and unpacked in such a way that an error free connection between level 3 in the DCE and DTE can be reached. For doing this level 2 has several finite state machines at his disposal. Besides information frames, containing level 3 packets, level 2 can handle several level 2 frames for link set up and disconnection, retransmission, etc.

5.1 Level 2 states

The operation of level 2 is supervised by the high level 2 controller. This link manager determines whether the link to the network should be established, disconnected or reset, and co-ordinates the operation of the transmitter and receiver. In chapter 6 a hardware description of high level 2 is given.

High level 2 can be in the link-up situation, in the link connect and the link disconnect phase, and in the link-down state. Besides these four states, there is a Reject condition in which level 2 is waiting for a link reset indication from the DCE for instance (the chip being part of a DTE), or a disconnect command. In the link-up situation we can distinguish 5 more states, explained in paragraph 5.4

The link set-up and disconnection is done by so called Unnumbered frames. These frames contain no sequence numbering. The different frame types are:

SABM :Set asynchronous balanced mode (link reset, link set-up).

UA :Unnumbered acknowledge (various acknowledge functions).

DISC :Disconnect (used in case of a wish to disconnect).

DM :Disconnect mode (indicating a state of no connection).

FRMR :Frame reject (not recoverable error detected).

See figure 14 for an complete control byte survey of the different frames.

5.2 Link set-up

In figure 15 the states important for the link set-up and disconnection are drawn. Starting from a chip reset, the link is in an initial state. In this state idles are transmitted. If the host enables a logical link, the disconnected state is entered. In the disconnected state flags are transmitted, indicating that the chip can accept a logical connection. If a SABM is received, an UA is responsed and the link is up. If the host takes the initiative to connect the station to the network, a SABM is transmitted, and the SABM-sent state is entered. Besides other reactions the solicited frame is an UA, and after the reception of this frame again the link-up state is entered. This is in a short view the normal link set-up. If there are other frames received, then the normal procedure can be interrupted, and several responses are possible then, depending on the received frame. All the responses are in figure 17, and an explanation of this figure is in paragraph 5.5.

5.3 Link disconnection

Link disconnection can be the initiative of the host, and then a DISC command is send out. The DISC-sent state is an intermediate state for waiting for the acceptance of the disconnection, which can be done by several frames. If the other side of the link takes the initiative of disconnection, on the reception of a DISC frame an UA is responsed, and the disconnected phase is entered. The other possibilities of entering the DISC-sent state are in figure 17.

If a FRMR is received, the link is reset by transmission of a SABM. The normal procedure as described in paragraph 5.2 is then followed. In case of a not recoverable error, the chip sends a FRMR and goes to the frame-reject-condition. On the arrival of a SABM the link is reset.

In all states there are several other unnumbered frames both commands and responses possible, but the here mentioned are the most likely. Other frames can result in other state transfers.

5.4 Link-up state

In the link-up state there are four different frames possible. First of all the frame used for data transport:

I-frame :Information frame (frame containing level 3 info).

Besides I-frames, there are frames to maintain the right behaviour of the link. The so-called supervisory frames are:

RR :Receiver ready (indicating a possibility to receive I-frames).

RNR :Receiver not ready (the receiver can not accept I-frames).

REJ :Reject (An out of sequence I-frame is received).

See figure 16 for the control byte contents of these frames.

I-frames, containing level 3 information, have a sequence number. The sequence of this frames is guarded by the level 2 control end maintained with supervisory frames. The S-frames are for indicating a not ready situation, a ready situation and the rejection of out of sequence I-frames. These frames contain a sequence number of the next expected I-frame. The supervisory frames are important in the link-up state. An unnumbered frame is always overruling a supervisory frame.

Inside the link-up state we can distinguish several states. The data transfer state is the state in which both transmission and reception of level 3 frames (I-frames) can occur. We enter in this state after the link set-up or reset. From this state a jump to the other station not ready situation is possible in case of the reception of a RNR frame. No I-frames are transmitted then, but incoming I-frames are accepted. The other station can handle its problems then, until it is able to receive and handle I-frames again. This is signalled by a RR or REJ frame.

If out of sequence frames are received, a REJ is sent, and no incoming I-frames are accepted as long as the right frame is not received. If level 3 signals a busy condition, a RNR is sent out and the RNR sent state is entered. Now incoming I-frames are not handled, and not transferred to level 3. Besides this data transfer state, RNR sent state, other station not ready state, and REJ sent state there are two other states, consisting of a mixture of the preceding states. These states are: the REJ-sent & other station not ready state, and the both stations busy state.

Besides the normal responses to a state change, there are in all states several responses possible, and state changing if required. All these responses to different situations and the initiatives requested are done by the high level 2 microprogrammable processor. A transmitter and receiver process work together, with bus interleaving. Even parts of the controller (for instance the pipeline registers) are shared by the two processes.

5.5 State summary

As mentioned before, in figure 17 all state changes are drawn. Not all high level 2 actions are with the comments near the arrows, because of the complexity and number of these actions. For the same reason no paragraph numbers of the X.25 recommandation are added.

In the horizontal direction the states are put in the form of lines, and the arrows between these lines represent the state changes with a short reason and a short description of the action. A Rx or Tx addition indicates an action primarly of the receiver or the transmitter.

In chapter 6 the receiver and transmitter functions are described and their cooperation. The stars in the drawing are indicating difficulties in the receiver/transmitter actions. A Single star indicates the situation that N2 retransmissions took place. The retranmission counter update is a transmitter task. The link reset in case of an excess of retransmissions is a receiver task. The solution to this problem is to make the receiver check the retransmission counter before preparing a frame in the next-to-send-1 & 2 registers. The receiver tells in the rest of the state transfers the receiver what to transmit. The transmitter simply transmits receiver frames or if these are not present I-frames. This only if it is allowed. If transmission is

allowed the transmitter retransmits I-frames, else, if no transmission is allowed, only the U and S frames from the receiver.The high level 2 working is in the next chapter described.

The two stars indicate a situation that the transmitter has to make a choice. He is allowed to transmit I-frames, but if they are not available, a RR or RNR as commanded by the receiver must be transmitted. This is a deflection of the general rule of the transmitter only carrying out the receiver commands. The solution to this is to let the Tx check the next-to-send control byte first if it is a RNR or a RR, before transmitting I-frames. In case of a RNR or a RR the I-frame is transmitted, else the frame of the receiver is chosen for transmission.

TABLE 3/X.25

Commands and responses

					1	2	3	4	5	6	7	8
Format	Commands		Responses		Encoding							
Information transfer	I	(information)			0	N(S)			P	N(R)		
Supervisory	RR	(receive ready)	RR	(receive ready)	1	0	0	0	P/F	N(R)		
	RNR	(receive not ready)	RNR	(receive not ready)	1	0	1	0	P/F	N(R)		
	REJ	(reject)	REJ	(reject)	1	0	0	1	P/F	N(R)		
Unnumbered	SARM	(set asynchronous response mode)	DM	(disconnected mode)	1	1	1	1	F/F	0	0	0
	SABM	(set asynchronous balanced mode)			1	1	1	1	P	1	0	0
	DISC	(disconnect)			1	1	0	0	P	0	1	0
			UA	(unnumbered acknowledgement)	1	1	0	0	F	1	1	0
			CMDR	(command reject)	1	1	1	0	F	0	0	1
			FRMR	(frame reject)								

Note 1 – The need for, and use of, additional commands and responses are for further study.

Note 2 – DTEs do not have to implement both SARM and SABM; furthermore DM and SABM need not be used if SARM only is used.

HDLC-TABLE

FRAMES	DIRECTION	ADDRESS
COMMANDS	DCE ⟶ DTE	A
	DTE ⟶ DCE	B
RESPONSES	DTE ⟶ DCE	A
	DCE ⟶ DTE	B

ADDRESS-TABLE

FIG. 14

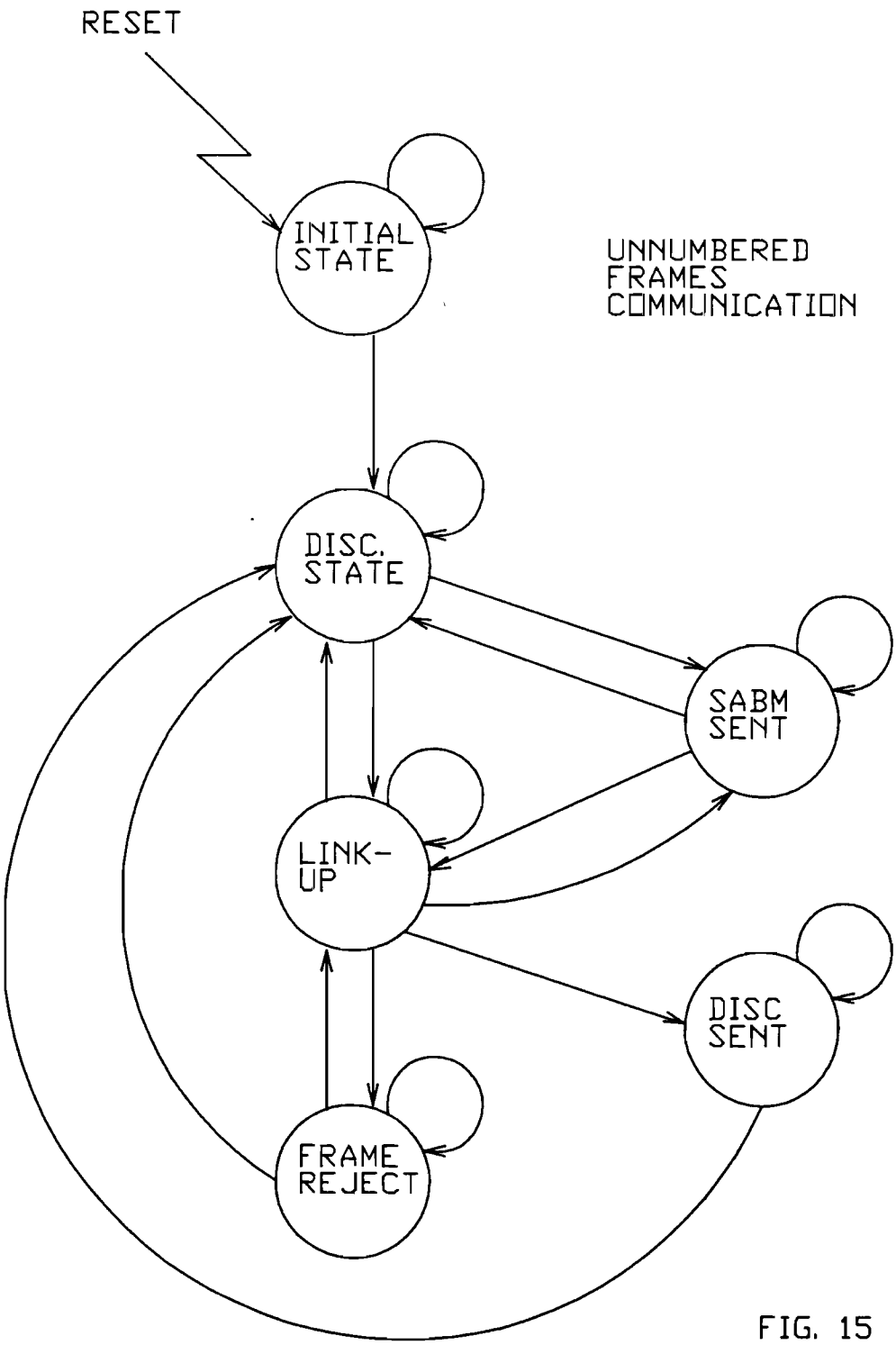

RESET
INITIAL STATE
UNNUMBERED FRAMES COMMUNICATION
DISC. STATE
SABM SENT
LINK-UP
DISC SENT
FRAME REJECT

FIG. 15

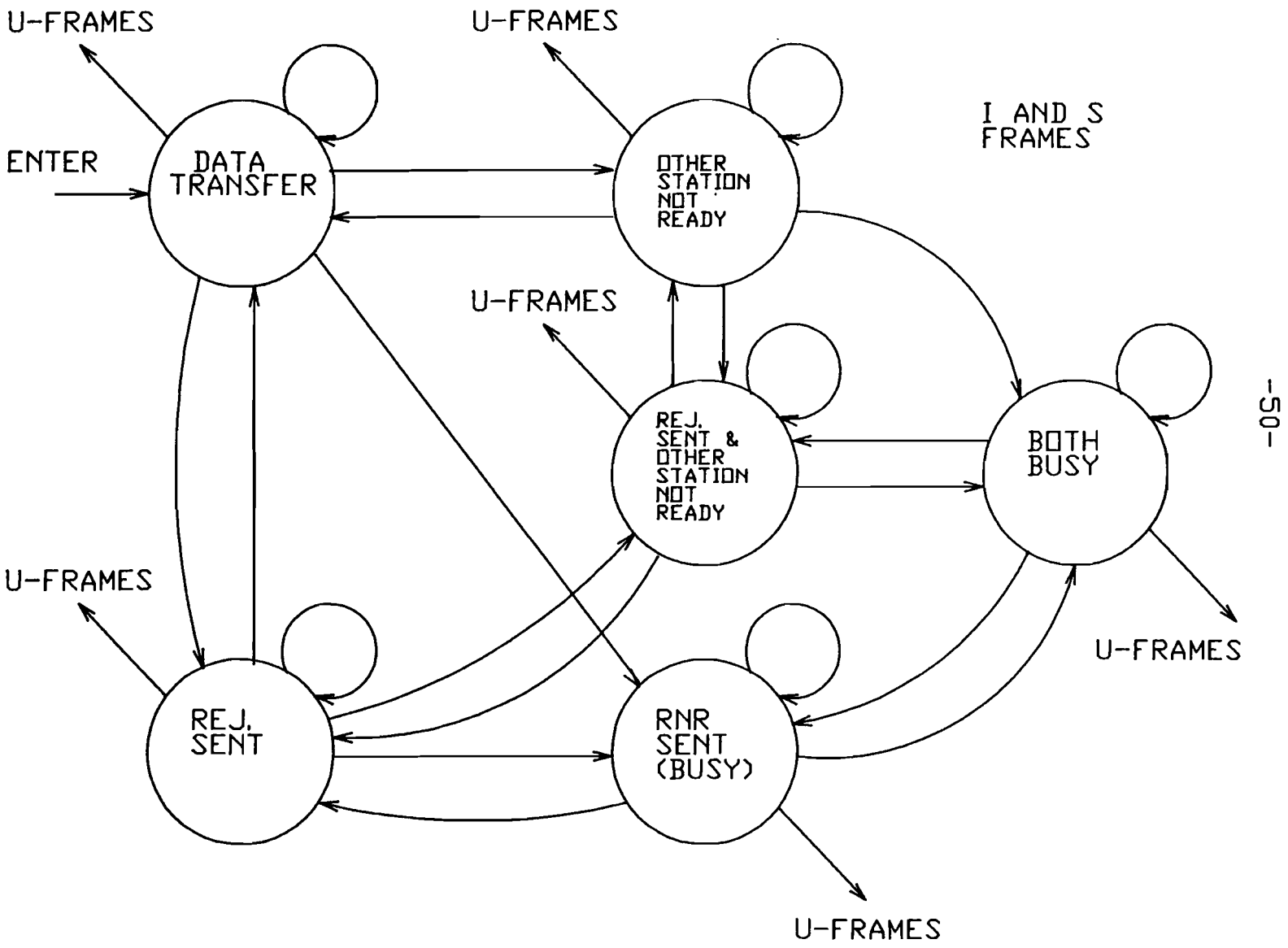

U-FRAMES
U-FRAMES
I AND S
FRAMES
ENTER
DATA
TRANSFER
OTHER
STATION
NOT
READY
U-FRAMES
REJ.
SENT &
OTHER
STATION
NOT
READY
BOTH
BUSY
U-FRAMES
U-FRAMES
REJ.
SENT
RNR
SENT
(BUSY)
U-FRAMES
U-FRAMES

FIG. 16

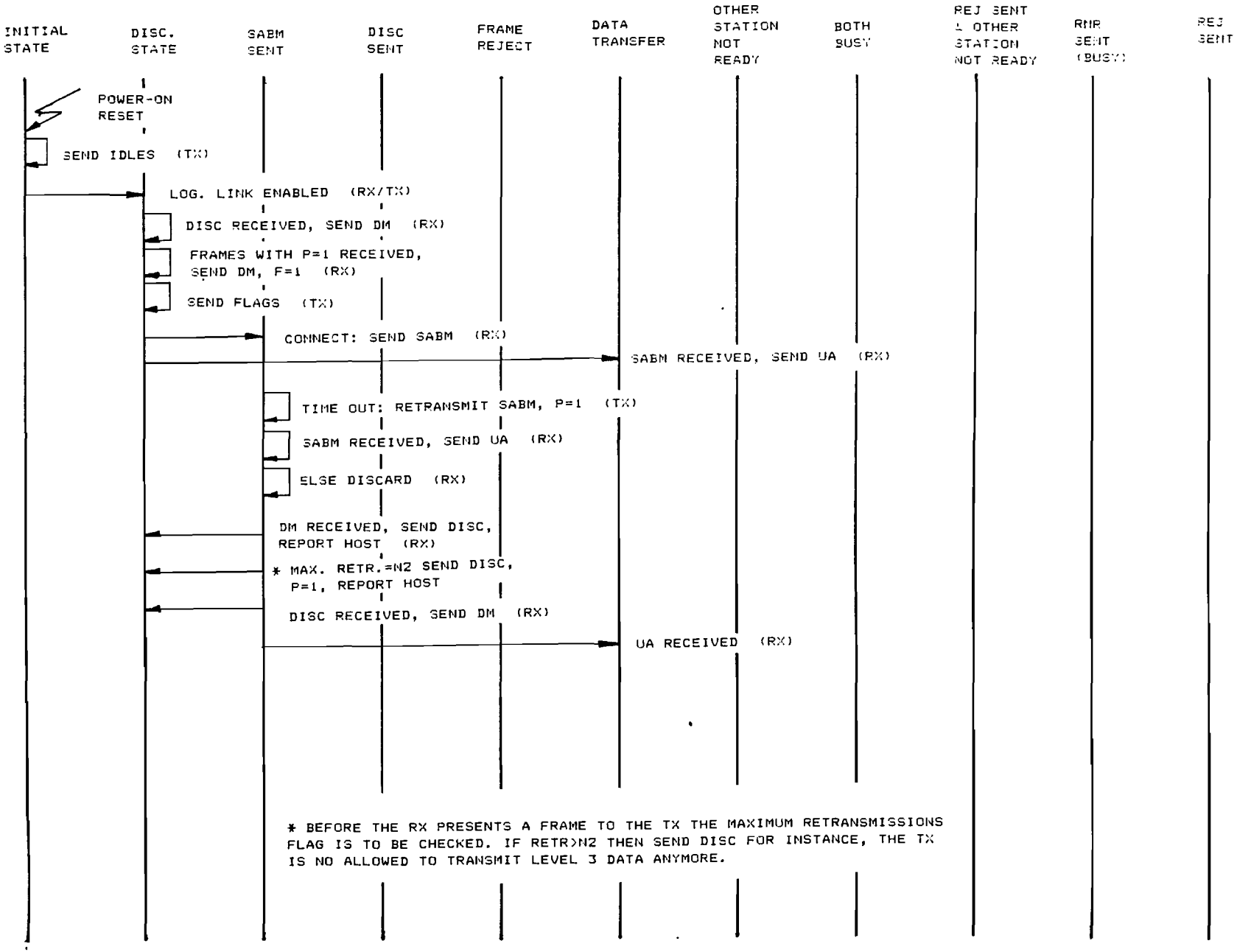


FIG. 17a STATES

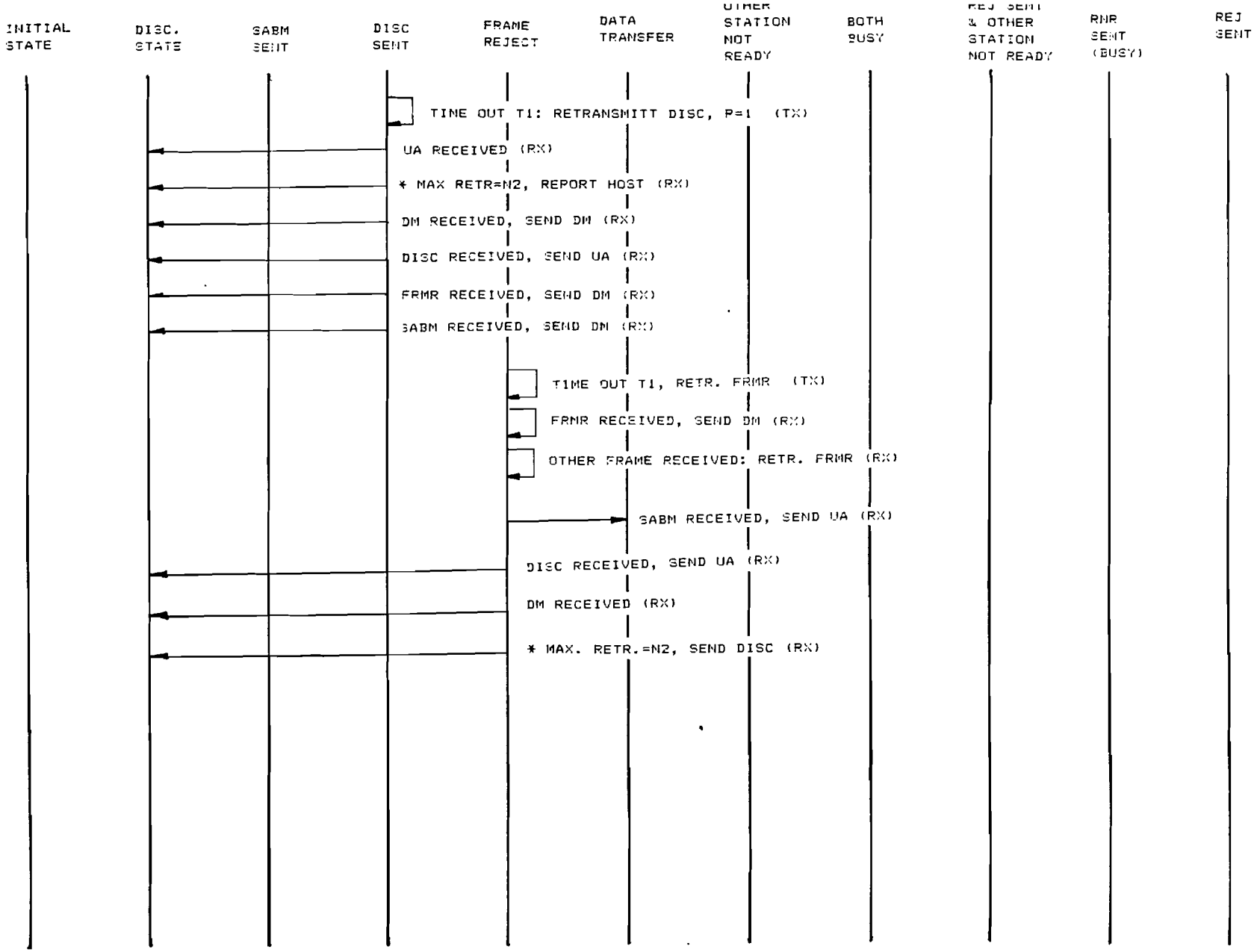


FIG. 17b STATES

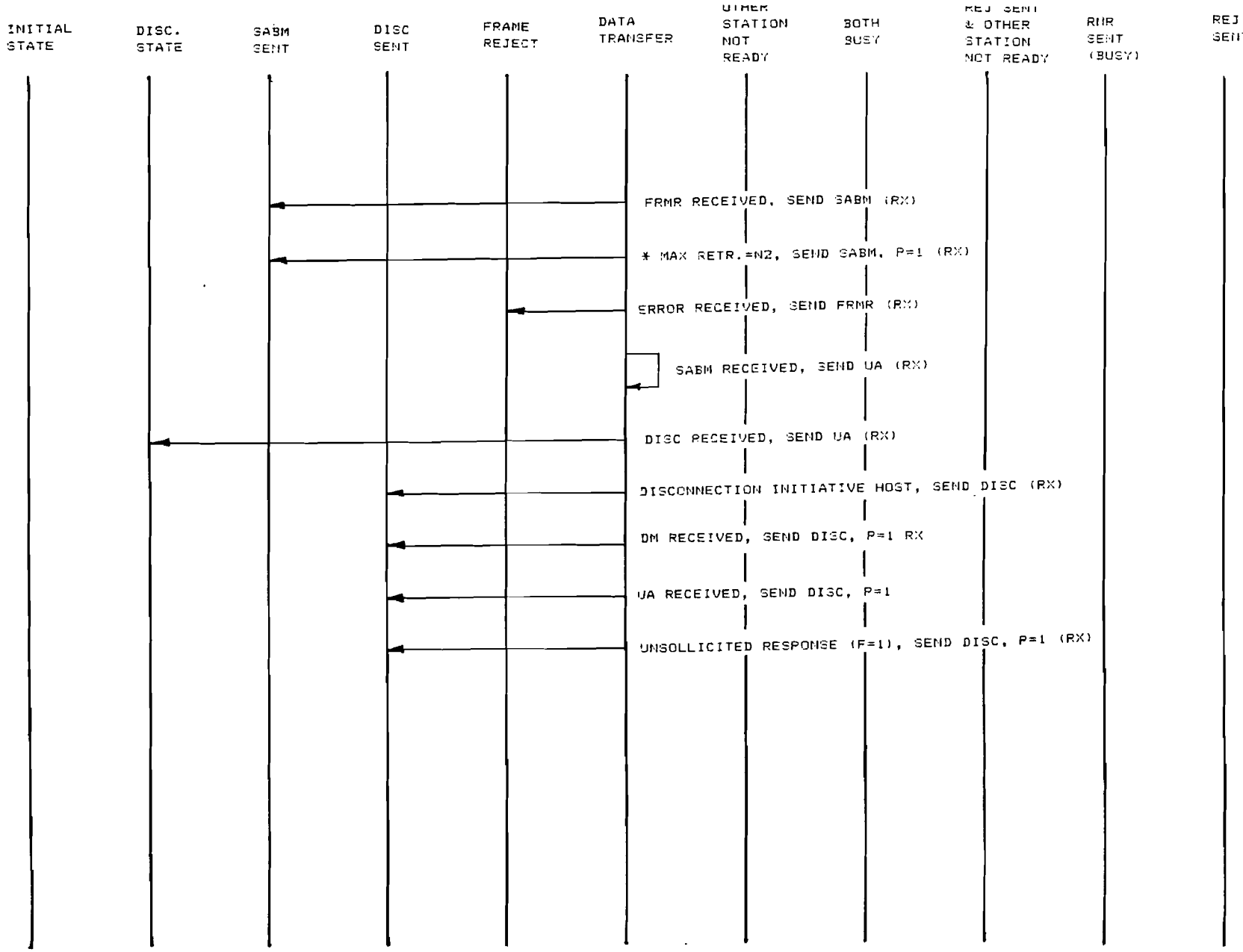


FIG. 17c STATES

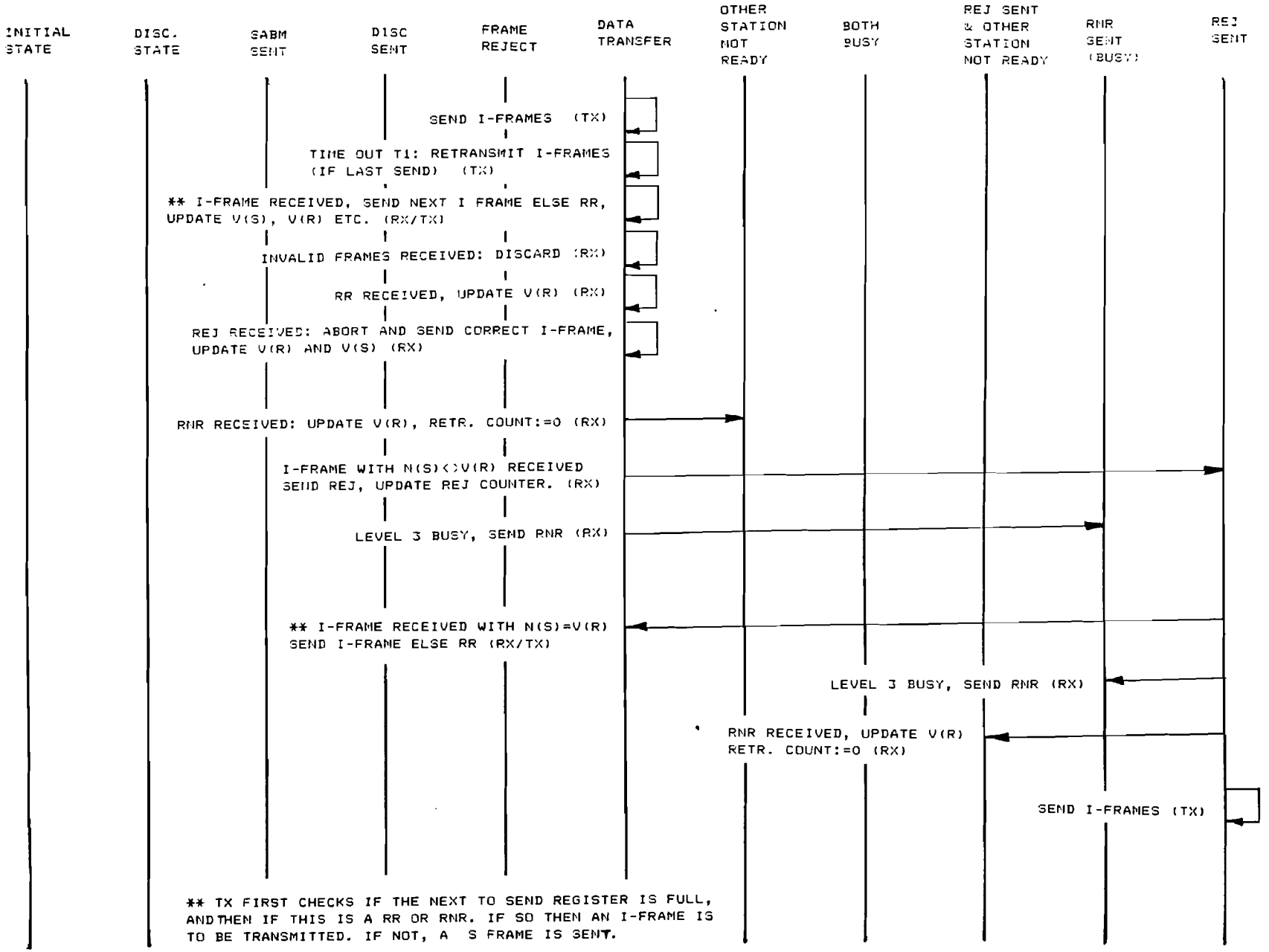


FIG. 17d STATES

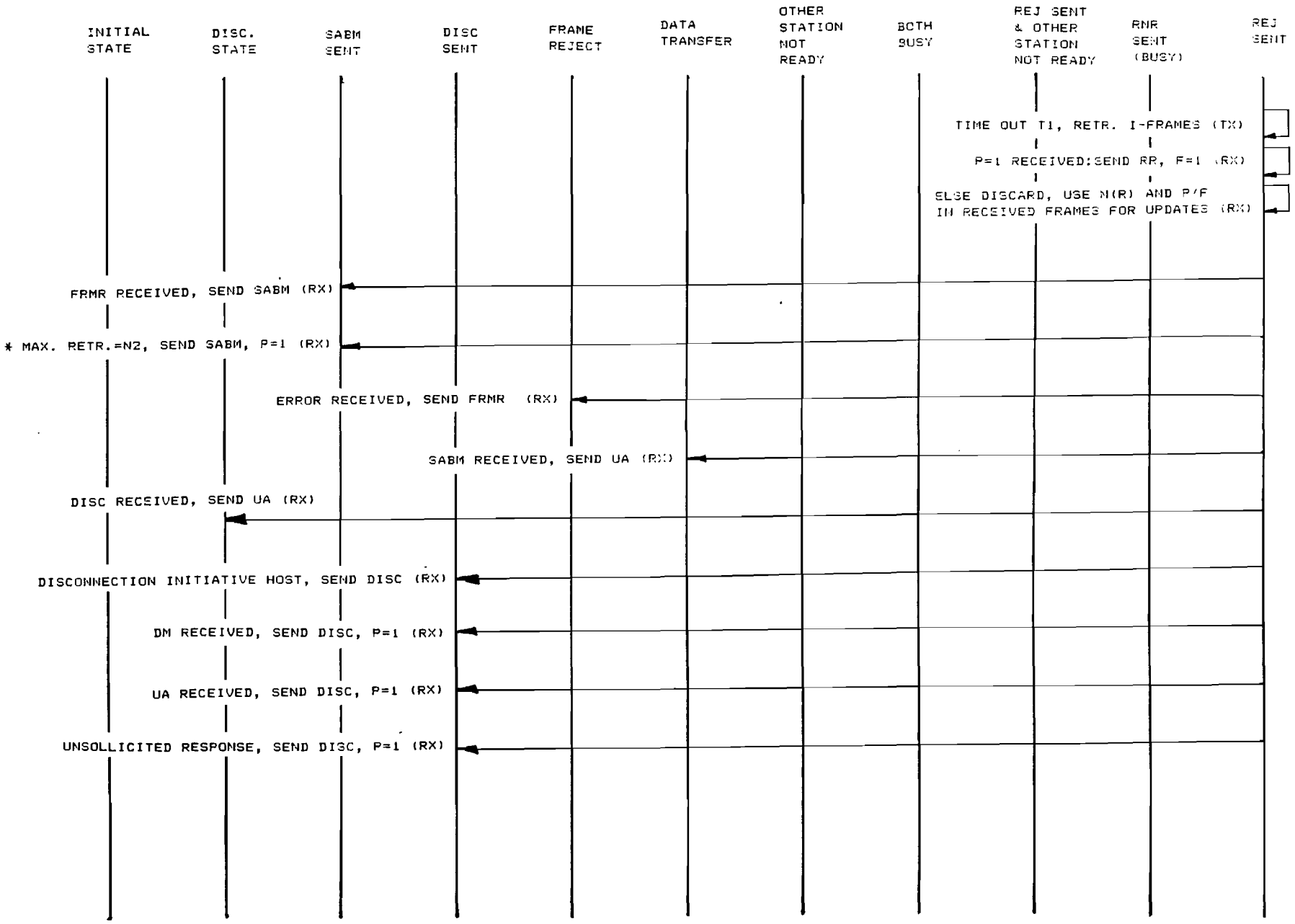


FIG. 17e STATES

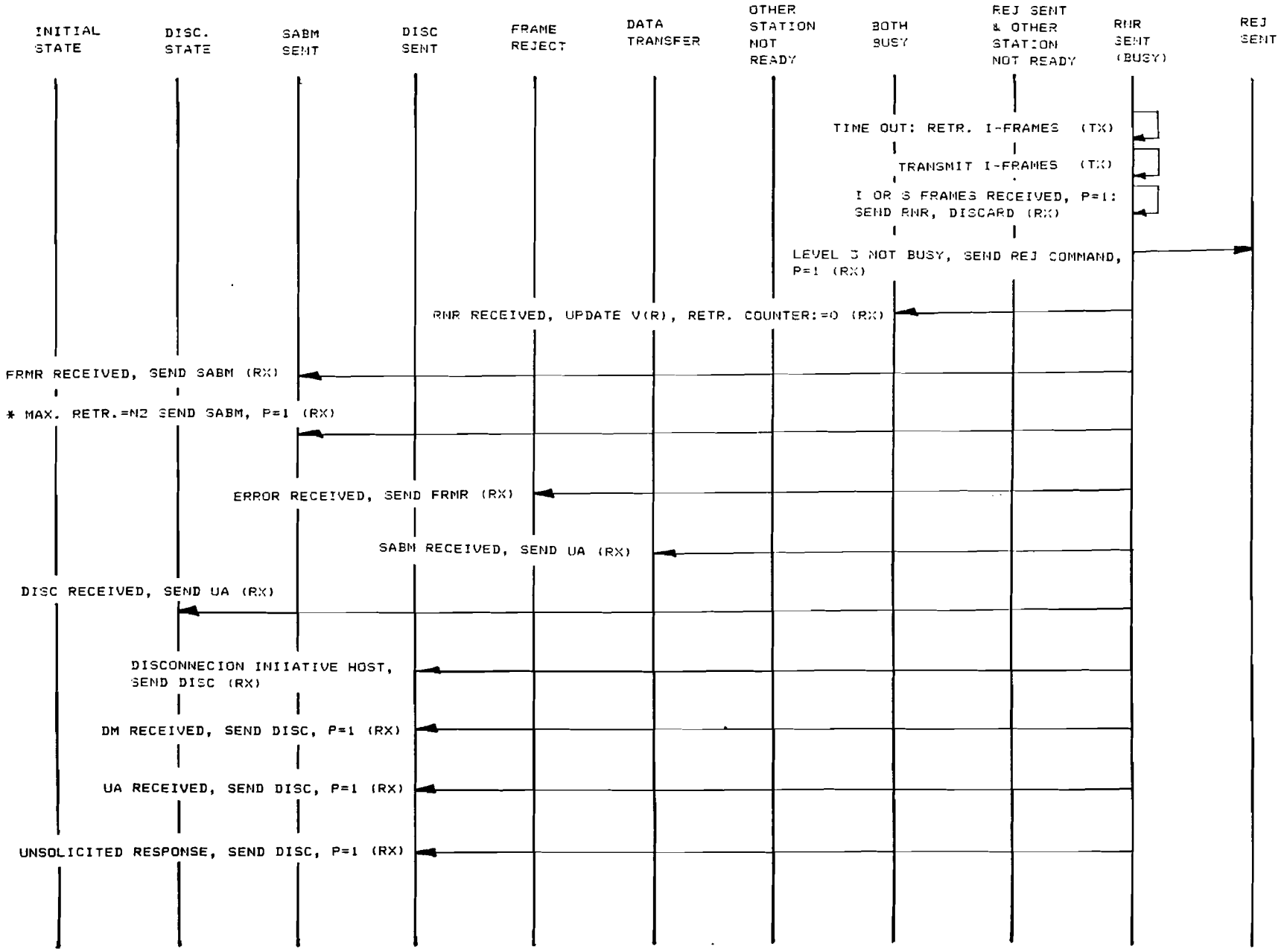


FIG. 17f STATES

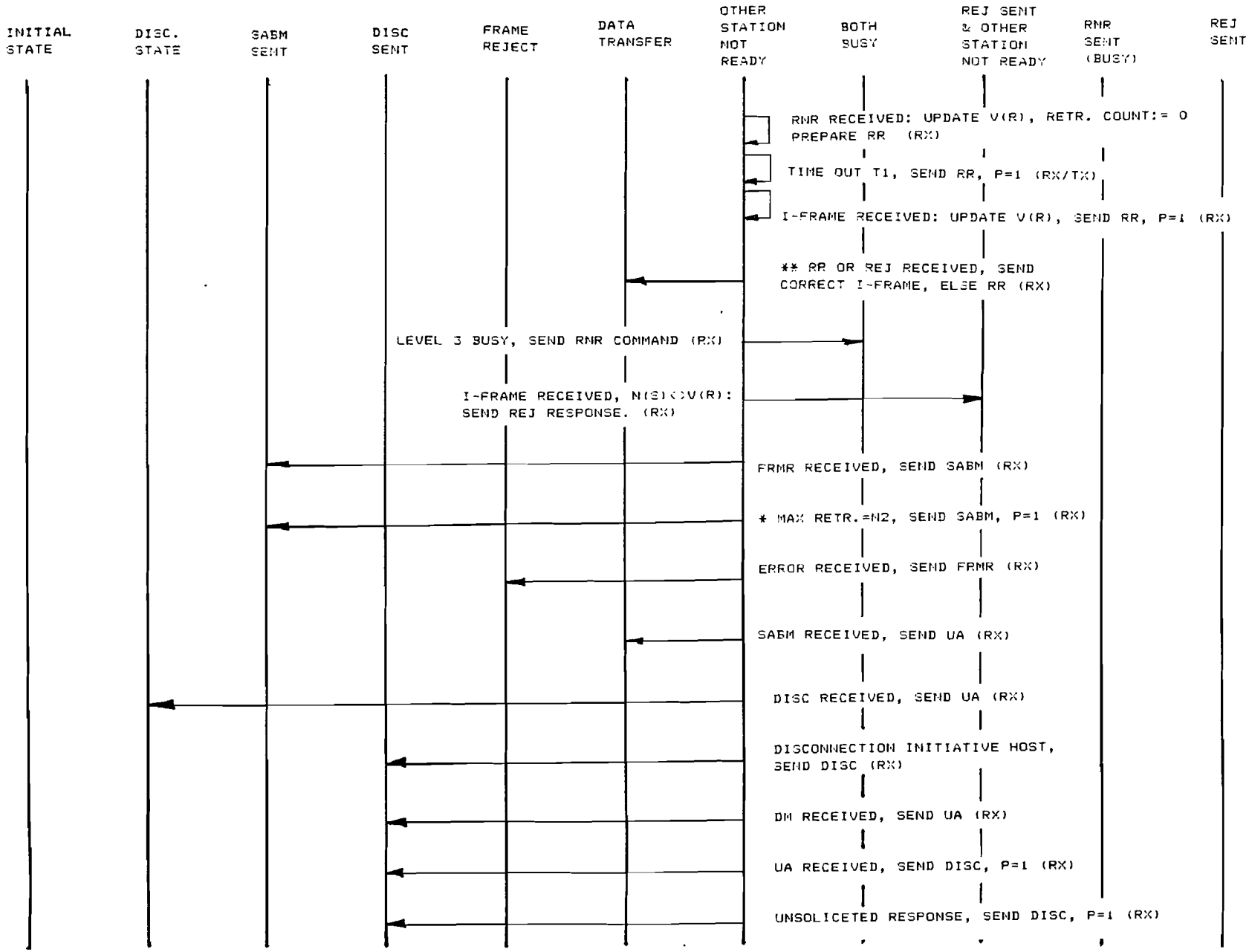


FIG. 17g STATES

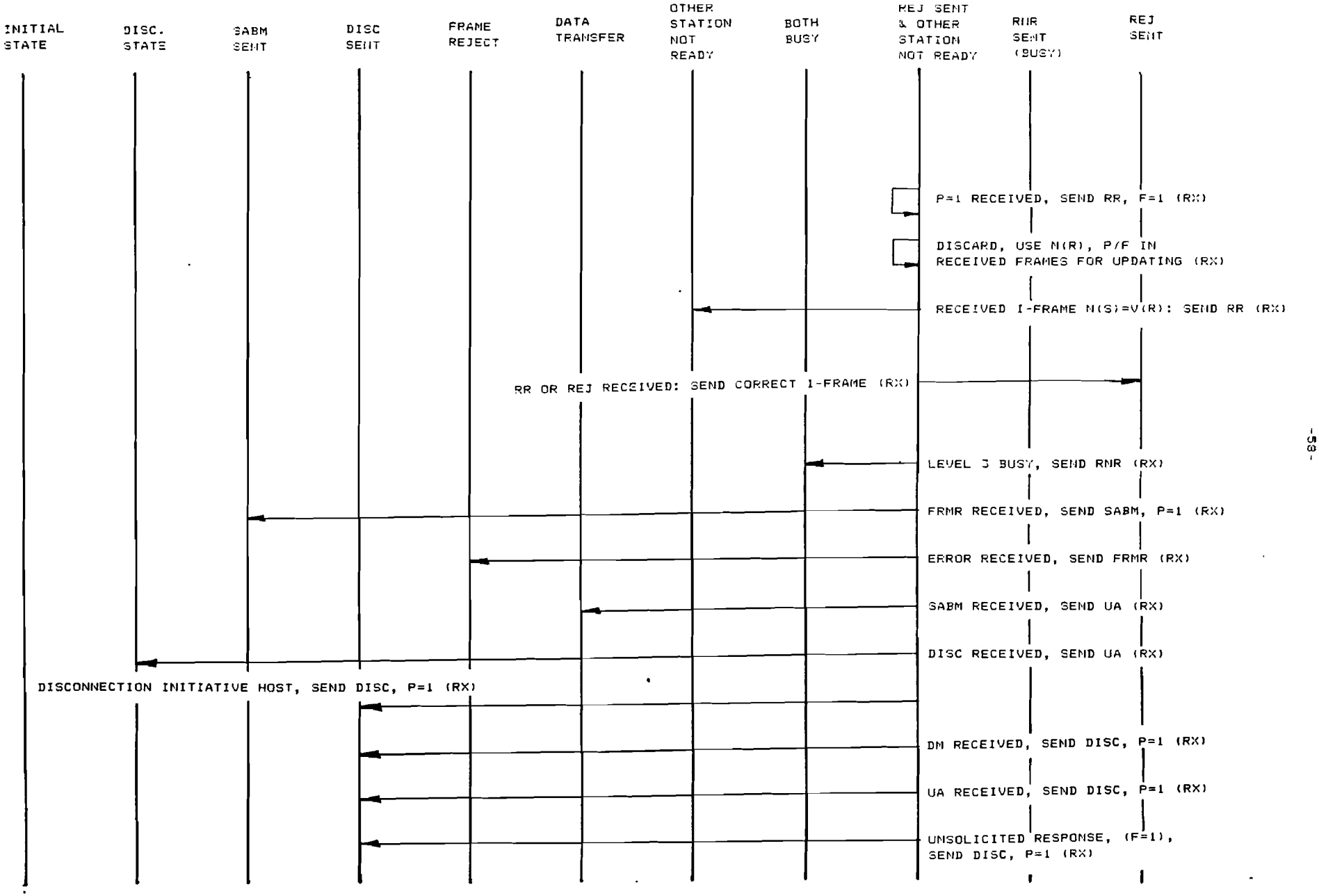


FIG. 17h STATES

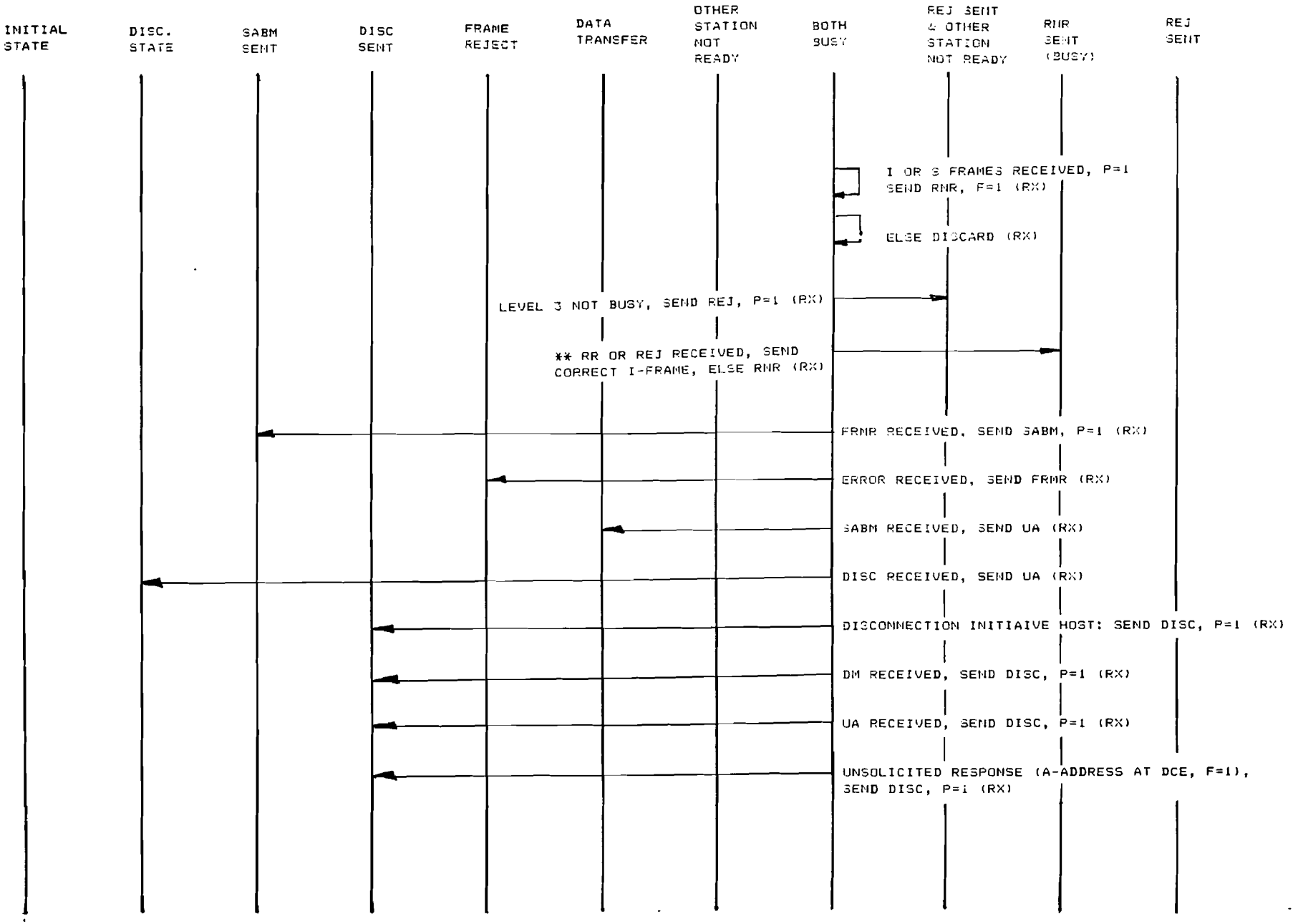


FIG. 17i STAIES

6 High level 2 construction

To support the different functions of the receiver there are a 4 bit wide bus and an ALU shared with the transmitter part of high level 2. The bus is separated in 3 bit data and a single bitline for the microprogrammable controller only. The data is so organized that always a 3 bit number and a single information bit are together in the different frames, so a 4 bit bus is a natural choice. The different registers in the high level 2 receiver are mostly also accessible by the transmitter. Figure 18 shows the high level 2 hardware. On the left side the low level 2 machines and communication registers, and on the right side low and high level 3. The bus interleaving mechanism assures that no collision can occur. After every sequence of microinstructions, the last microinstruction switches the microcontroller to a part of the transmitter program, pointed to by the tx-program count register. This register can be found in the microcontroller. See chapter 7. The other registers shown in figure 18 are registers used for the program status and flow control of the frames.

6.1 High level 2 receiver

The receiver must analyse the received control byte of a frame and the received address information. This includes the different valid signals, delivered by the low level 2 receiver. Depending on the information delivered by this process action must be taken. This actions are described in the X.25 recommendation of the CCITT.

The functions of the receiver are:

- First of all: check the validity of the frame. If this is not OK, then discard the bits that are received.
- Maintain the sequence of level 3 packets.
- Signal to level 3 the receipt of level 2 diagnostic information (FRMR frame). The FRMR frame contains an indication of the detected error in a frame.

- Handle a BUSY signal from level 3. A RNR must be transmitted, and no incoming I-frames can be accepted.

- Handle the frame information as there is PACKET_END and PACKET_VALID, and deliver this to level 3.

- update the information for level 3 concerning the last acknowledged frames, etc.

- Take action on the receipt of an ABORT, IDLE or flag signal from the low level 2 receiver.

- Count the number of REJ frames transmitted. This is an indication of the line quality of the channel to the receiver.

The receiver action in the different states can be read from fig. 17. Every receiver action has an addition "Rx" near the arrow indicating a state change.

6.2 High level 2 transmitter

The transmitter is much simpler than the receiver part of high level 2. The transmitter has only to transmit the frames of the receiver which have to be send to the other station, and the I-frames of level 3. All the different responses to the incoming frames are not handled by the transmitter. Only transmit sequence information (V(s)) and Next-packet-to-be-send pointer are updated by the transmitter. Beside this simple actions the only actions are commanded by level 3 or initiated by a counter or timer overflow.

Briefly all transmitter functions:

- Check the time-out by means of a max-timer (loaded from level 3). This time-out occurs when no response is received after a maximum time. The timer is started when a frame (command) is transmitted, with a bit set, the so-called P(oll) bit. The timer is reset when, as an answer to the poll, a bit set in a received frame, the so-called F(inal) bit.

- Check the number of retransmissions. There is a maximum-retransmission counter that can be preset

from level 3.

- Transmit on request from level 3 the level 3 packets. This includes the preparation of an address, an control byte and the sending of several signals to the low level 2 transmitter.

- Sending of a FRMR frame if required. For this purpose a shiftregister is accessible from the 4 bit bus.

- Keep the send sequence numbers V(s) and next-packet-to-transmit up to date, as well as the retransmission counter.

- Send REJ on fault sequence numbers.

- Control the transmission of packets from level 3 by the lines Packet-Request, Packet-Ready and Transmit-packet-end.

- Communicate with level 3 about the last acknowledged packet number, the next packet to send and the maximum number of outstanding packets.

- If necessary transmit an idle or abort by signalling this commands to low level 2.

- Transmit the different control frames.

The transmitter actions can be read from fig. 17. Mainly transmitter actions have the addition "Tx" near the arrows.

6.3 High level 3 communication

Between high level 2 and the high level 3 controller there is an exchange of information. This communication goes via the registers upper right in the drawing 19. Over these registers an attention is generated, if the first bit of the register is set. The bits are reset by level 3. Setting by level 2 is only allowed if first is tested if the bit is reset. In the same way the registers over which the bits are signalling to be filled with information are only allowed to be accessed by level 2 if the bit in the register is reset. Level 3 of course can only read the registers reporting the reject counter value, the last acknowledged packet number and the next packet to send number. As can be read

hereafter, the other registers are only once set in a certain value at the initialisation. Because of the construction of the level 2 and level 3 controllers there is no problem with the timing. The controllers work exactly on the same clock and on the same microcycle timing, so a collision for instance with a processor writing and at the same time the other processor also writing can not occur.

REJ count — 3 bit (or 4 bit) countervalue, counting the reject packets transmitted (COMMUN. REG 1)

Pack. ackn/ Last ackn. pack. nr. — Flag indicating that a packet is acknowledged./ The number of the last acknowledged packet. (COMMUN. REG 2)

1 sec. timer/ k — A timing flag supplied for level 3 by level 2./ K is the maximum number of outstanding I frames, set at initialisation. Maximum is seven, mostly k is set to 2 (Dutch PTT). (COMMUN. REG 3)

Pack. request/ next pack. to send — A flag indicating that a packet can be transmitted by level 2./ The number of the packet that can be transmitted. (This is the sequence numbering of the packets in the hosts memory). (COMMUN. REG 4)

Max. retransmissions N2 — The maximum number of retransmissions that level 2 is allowed to do before the situation is becoming special. N2 is set at initialisation and is mostly 5 (Dutch PTT). The register is accessible from level 2 and has a parallel register with the actual value of the retransmissions that where necessary up to that moment. There is a maximum overflow flag in this register for level 3 available. This is the reason discussing this register with the level 3 communication and not with the initialisation procedure.

Max. timer T1 — The maximum time a packet is allowed not to be acknowledged. The value of T1 is again set at initialisation and can be 2,125 sec. or 200 msec. (Dutch PTT). The register with the maximum timer value has a counter connected to it and this counter can set a flag that can be read by level 2, and by level 3. This is the reason this register is also put in this part of the report and not with the initialisation.

6.4 Low level 3 communication

Low level 3 handles the standard assembling and disassembling of the level 3 packets. For doing this a certain synchronisation with the transmitter and receiver of level 2 is necessary. The communication goes via the STATUS and COMMAND register. Over these registers no attention is generated.

STATUS:

Diagn./Pack. end/Pack. valid/ reset (soft) — Diagnostics flag indicates that the data being transferred to level 3 is diagnostic information, the contents of a FRMR frame./ Packet end indicates that the frame end flag of the information frame is received./ Packet valid is the signal that no error has occurred in the transmission and that the packet sequence number is allright./ Reset is a possibility to reset level 2. This is also possible by changing the enable level 2 and start connection bits.

COMMAND:

Busy/Transmit pack. end/pack. ready/abort — Busy is a command for level 2 to stop the transfer of level 3 info to the level 3 receiver. A RNR frame has to be send./ Transmit packet end is the command to end the transmission of data from level 3. A

FCS code must be transmitted and a closing flag./ Abort is a command to end the current transmission also, by transmitting an abort sequence./ Packet ready is an indication that the requested level 3 packet is ready for transmission.

6.5 Internal level 2 registers

For the controlling of the link several values are necessary to remember. The sequence control and the retransmission are examples of recalling previous used information. Hereafter the registers for storing the used information. The next frame to be send is stored in the Next-to-send registers 1 & 2, and gives the transmitter the frame control word the receiver wants to be transmitted. The transmitter always tries to transmit the receivers frame first, and after that the I-frames, if allowed.

V(s)	The sequence number of the next I-frame to be transmitted.
V(r)	The sequence number of the next I-frame that is expected to be received.
Last_ackn._V(s)	The number of the last I-frame that is acknowledged from the opposite station.
Last_frame_sent_1	The first nibble of the control byte that is sent with the last transmitted frame.
Last_frame_sent_2	The second nibble of the control byte that is sent with the last transmitted frame.
Frame_to_send_1	The first nibble of the control byte of the frame to be send next.
Frame_to_send_2	The second nibble of the control byte of the frame to be send next.

Beside whole nibbles also singular bits are exchanged between receiver and transmitter. For this purpose a PROGRAM STATUS 1 register and a PROGRAM STATUS 2 register are present.

Poll_received	A flag indicating that a poll is received

and thus a final has to be transmitted.

Poll_send — A flag indicating that a poll is send and a final is expected in response to this.

Tx_enabled — A bit, allowing the transmitter to transmit level 3 info.

Last_address_sent — The last frame was addressed to the command address if this bit is set, else the response address was used. This is used in cooperation with the last sent control byte.

Next_address — The next destination address for the transmitter, when using the next to transmit byte. A one indicates a command, a zero a response

The transmitter is able to test if there is a frame to be transmitted from the receiver by testing the first bit of the next-to-send 1 register. If there is some U or S frame control byte, the first bit is always a one, as can be seen in fig. 14.

6.6 Other level 2 provisions

The transport of data nibbles between the registers in high level 2 is always done via a temporary register or the accu. The accu is a register with a latch for temporary storage in the case of arithmatic operations. For this arithmatic an 4 bit ALU is present and a flag register, both with a port to the 4 bit data bus.

Accumulator — Used as stocking area for the result of ALU operations. The accu is 4 bit wide.

Temporary reg. — This register is used by register-register transfers, and for temporary information memory for ALU operations.

Arithmatic Logic Unit — The ALU is used for the different logic and arithmatic operations needed in the different programs. The ALU can perform AND, OR, ADD and SUB actions. All modulo 16 (4 bit wide ALU).

Flag register — The flag register contains in sequence: a carry bit, indicating an overflow during the previous ALU operation, a zero bit, indicating a zero result in the accu and an attention bit, indicating that the low level 2 receiver asks for service.

6.7 Low Level 2 communication

The registers communicating with the level 2 receiver are already explained about their contents. The receiver has one command register, and 3 status registers. In the Control_rx registers the control byte of the received frame is stored. This register Control_rx_1 can contain for instance in case of an I_packet, a zero and the send sequence number N(s). Register Control_rx_2 then contains a poll bit (0 or 1) and the receive sequence number N(r).

The register for the status of the transmitter is called Status_tx. It contains only the ready message of the transmitter. The command registers are called Command_tx_1 and Command_tx_2. Their contents is also explained before. The Address_tx_1 register is filled with the A or B address first nibble. This is 1100 or 1000. The Address_tx_2 register is always containing zeros. The Control_tx_1 register contains the first control nibble, and the Control_tx_2 register the second nibble. In the case of an I-frame they contain a zero, and N(s) respectively a poll (0 or 1) and N(r). The FRMR registers 1 to 5 contain in the first two registers the control byte of the rejected frame, in the third register a zero and the current V(s), and in the fourth register a zero if the rejected frame was a command and a one if the rejected frame was a response. After this the fourth register contains the current V(r). The fifth register contains the W,X,Y and Z parameters about the reason of the frame rejection. W is set if the received control field was invalid or not implemented. X is set if the U or S frame had an incorrect length or if the frame contained an information field, which is not permitted with an U or S frame. Bit W must be set in conjunction with this bit. Y indicates a too long information field. Bit Z is set if the receiver control field contained an invalid N(R).

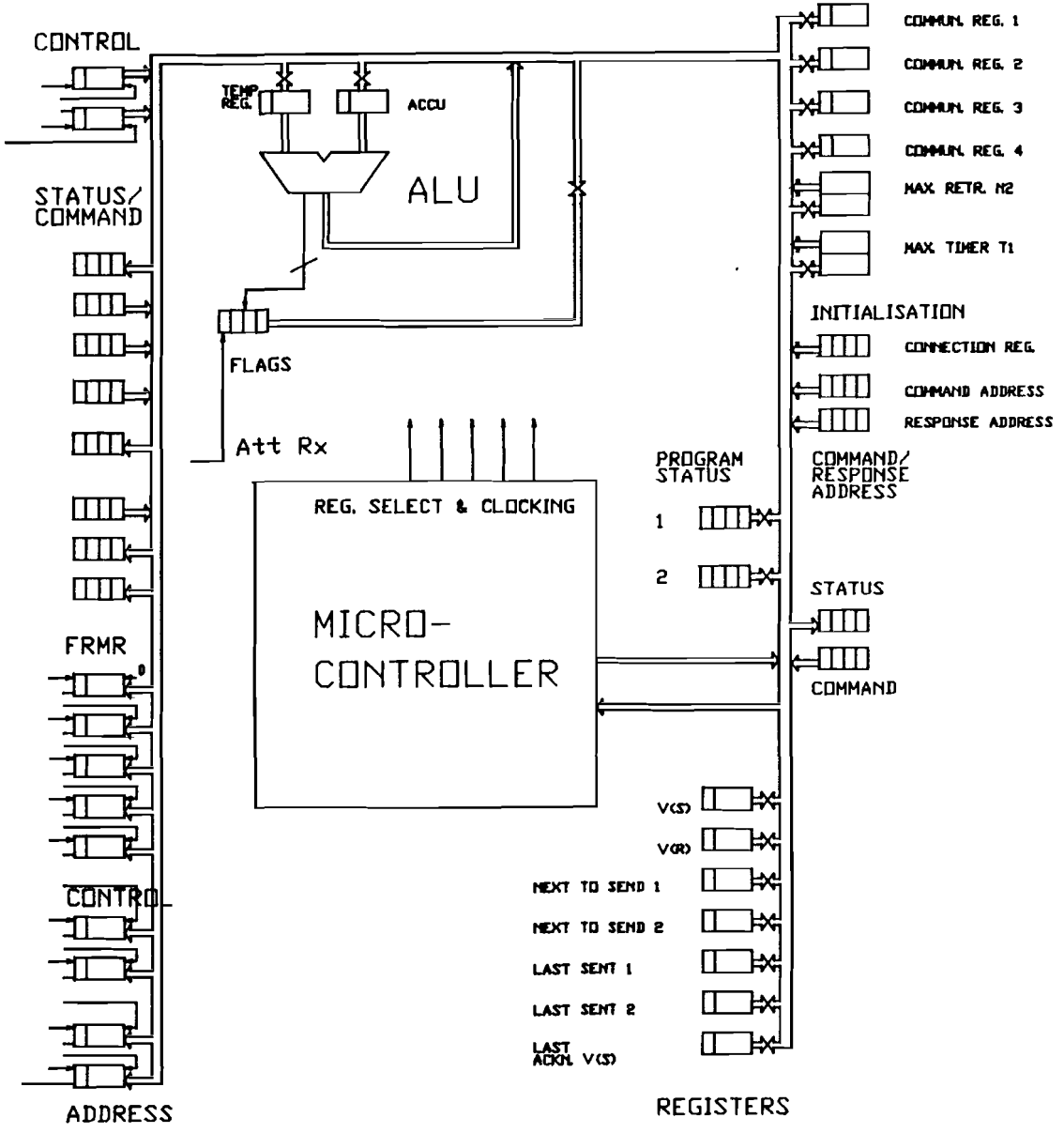
The following five bits in a frame reject are always zero, so the input of this last shift register is connected to zero. All zeros are loaded in this way during the shift operation.

6.8 Initialisation

The initialisation of high level 2 is done by level 3. Level 3 sets the maximum number of outstanding packets (I-frames), k, in comm. reg. 3. The maximum number of retransmissions is also set in a register, as is the maximum timer value T1. K is mostly set to 2, N2 at 5 and T1 at 2,125 or 200 msec depending on the class of operation. N1, the maximum framelenght is also set in low level 2. The command address must be initialized, and the response address. These addresses make the level 2 processor act as a DCE or DTE. See fig 14. The last register is the connection register, and is accessed if level 3 is ready with its own initialization. The first bit is the enable level 2 bit and starts the level 2 controller with the initialization and enables the other side of the link to make a call to the chip. The second bit in the connection register is a connection command and commands the level 2 controller to make a call to the other station. In the state diagram (figure 15) this causes an SABM and a state change to the SABM-sent state. With these actions, the initialization is completed. Low level 2 is than initialized as described. High level 2 only resets in the case of a general reset the V(S) and V(R) to zero and fills the other registers with zeros, and than starts the logical connection if enabled.

enable level 2/
start connection — The enabling flag indicating that level 1 can be enabled to make an electrical connection to the network and that level 2 after its initialisation is allowed to accept a request to make a logical connection./Start connection is a flag indicating that the host via level 3 wants a call set up and that level 2 has to take the initiative to do so.

Command address/
Response address — These registers contain a 4 bit address each. These addresses tell the chip to behave as a DCE or as a DTE, as far as level 2 is concerned. If the command register contains for instance an A address, the response register contains a B address



HIGH LEVEL 3

LEVEL2/3
COMMUNICATION REGISTERS

LOW LEVEL 3

FIG. 18

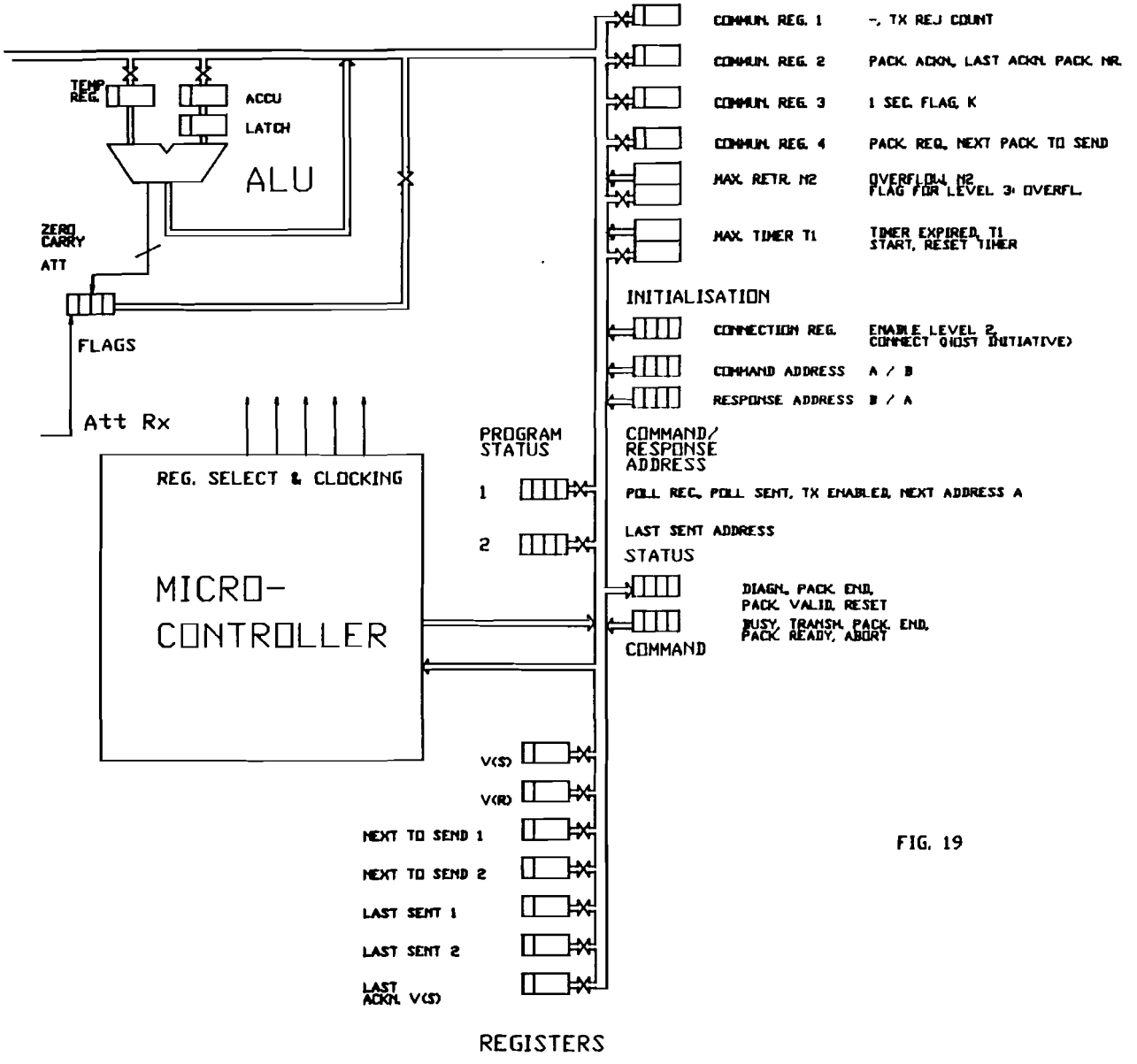

TEMP REG.
ACCU
LATCH
ALU
ZERO
CARRY
ATT
FLAGS
Att Rx
REG. SELECT & CLOCKING
MICRO-
CONTROLLER
PROGRAM STATUS
1
2
V(S)
V(R)
NEXT TO SEND 1
NEXT TO SEND 2
LAST SENT 1
LAST SENT 2
LAST ACKN. V(S)
REGISTERS
COMMUN. REG. 1
-, TX REJ COUNT
COMMUN. REG. 2
PACK. ACKN, LAST ACKN. PACK. NR.
COMMUN. REG. 3
1 SEC. FLAG, K
COMMUN. REG. 4
PACK. REQ., NEXT PACK. TO SEND
MAX. RETR. N2
OVERFLOW N2
FLAG FOR LEVEL 3 OVERFL.
MAX. TIMER T1
TIMER EXPIRED, T1
START, RESET TIMER
INITIALISATION
CONNECTION REG.
ENABLE LEVEL 2,
CONNECT (HOST INITIATIVE)
COMMAND ADDRESS
A / B
RESPONSE ADDRESS
B / A
COMMAND/
RESPONSE
ADDRESS
POLL REC., POLL SENT, TX ENABLED, NEXT ADDRESS A
LAST SENT ADDRESS
STATUS
DIAGN., PACK. END,
PACK. VALID, RESET
BUSY, TRANSM. PACK. END,
PACK. READY, ABORT
COMMAND

FIG. 19

7 Microcontroller

The controlling of the level 2 machines and the reacting on received frames is done by the program stored in the microprogrammable controller. The program itself is stored in a microprogram ROM. The receive and transmit process share most of the controller. As with the registers the two processes need both most of the instructions the controller can execute. In figure 20 a functional sketch of the controller is given. The bus is the high level 2, 4 bit wide databus. The addressbus is 5 bit wide, enough for selecting 32 read/write registers. A separate line from the Rx/Tx control timing gives the read/write information. See paragraph 7.4. Because not all registers on high level 2 are read/write, but for instance write only, combining read and write only registers gives a significant reduction of addresses. In this way 32 register addresses are enough. All conditions that have to be tested are available via the databus. Immediate data is also delivered via the databus to the high level 2 registers.

7.1 Rx/Tx concurrent programming

The change between the receive program and the transmit program is done by a microinstruction after the last program instruction, telling the timing unit to select as next address the other program count register. See figure 21. In this drawing we can see how after every short sequence of microinstructions the controller switches to the other program. The program is divided into pieces that cannot be separated, or need not to be separated for the timing considerations explained below. Critical sections in the program, for instance the testing and writing of a communication register, is not dividable. The programs of receiver and transmitter are thus executed by the controller in a sort of time sharing system.

Program switching costs one instruction-time: the instruction just being build up in the micro program ROM. The maximum time between an instruction-sequence of the receiver and transmitter is dedicated by the transmitter. During transmission of an I-frame the flag "Transmit packet end" can be set and then the instructions to level 2 low have to be given in one bit time. This bit time is at a

speed of 48 kbit/sec, about 21 microseconds. If a microcycle time of about 0.5 microseconds is reachable, this allows the execution of 20 instructions, before the situation becomes critical. In the 21 usec. 40 instructions are allowed, but receiver and transmitter have to share the processor, and so only a sequence of 20 instructions is allowed. For a optional 64 kbit/sec and a high frequency clock of 25 Mhz, divided through 5 for a 200 nsec microinstruction cycle consisting of five phases, this results in 75 microinstructions in a bittime. For the receiver and transmitter each lets say 35 instructions.

If the receiver in this most critical operation allows the transmitter to check the flag bit, it is only for a short time stopped. A sampling rate of twice a bit time is enough to assure the signalling of a bit change in time. The transmitter is so allowed to work after at most 35 receiver instructions.

The transmitting process takes much less time compared to the receiving process. If this is a 25% to 75 % partition, the now most critical operation is the reaction of the receiver to incoming packets. The time for this available is the flag, address, control and FCS code receiving time. This time is also 40 bittimes. This is at a speed of 48 kbit a second, about 830 microseconds. Again at a microinstruction execution time of about 0.5 microseconds this allows the execution of 1660 instructions. If the receiver is 75% of the time allowed to be busy, this results in the possible execution of about 1200 instructions. This seems to be enough. On a much higher bitrate of 64 kbit/sec and a 25 Mhz high frequency clock, about 3000 instructions can be executed. If we take again the 25% - 75% partition the receiver can execute 2250 instructions.

As described before, all communication between the receiver and transmitter process take place with the help of bit-registers, grouped to a nibble, and accessed via the bus. For the working of the receiver and transmitter processes see chapter 6.

7.2 Hardware provisions

After the microcode ROM containing all receiver and transmitter program elements, a pipeline register is situated. The pipeline is a provision to increase the speed of the program execution. It allows the next program address to be supplied to the ROM during the execution of the last

from the ROM read instruction. If this next instruction has had its set-up time, it can be clocked into the pipeline, and clocked out if the next instruction is allowed to take the next program step. This clocking is done by the Rx/Tx/ control timing.

To allow jumps, calls and rx/tx program switching, a next address multiplexer is provided. Its input is from the expansion ROM, from the stack or from the program count registers, rx and tx. It is controlled by the rx/tx control timing. The output of this address multiplexer is supplied to the microprogram ROM, and to an incrementer. This incrementer delivers the address+1 to the program count registers. The value can be clocked into either of these registers by the rx/tx control timing. This timing unit delivers the timing to the several elements of the microcontroller, and to the high level 2 registers. All clock pulses for register operations, and valid signals are coming from this unit. A choice of a limited number of signals can be made, for instance a register to temporary register transfer, a reversing operation, an accumulator to register transfer, and so on. The instruction what to do is from the condition selector and the pipeline.

The condition selector is selecting a true/false information bit from the ALU flags (zero, negative), from the attentions from the different levels, and from the four bits on the databus. The condition selection is done by the output of the expansion ROM. Four bits can be selected, so the microcode only has to contain a two bit code for the selection.

The last unit in the high level 2 microcontroller is the stack pointer, containing the place to where the stack is filled. It is increased or decreased by the rx/tx control timing. Its output directs a value from the stack to the next address multiplexer. The stack is projected to be 4 deep, enabling the receiver to call a nesting of 4 subroutines. The transmitter program is so simple that this needs no subroutines and therefore no stack space.

7.3 Microinstructions

All data handling can be done with 3 groups of instructions. The 3 groups are: a branch group, a data move group and an arithmatic instruction group.

In the first group we have the conditional jump, the

conditional call and the conditional return. By the way, unconditional branches can be achieved by selecting a condition being always one, for instance a bit in the flag register. In the second group we have the move from accu or temporary register to the registers, from the registers to the accu or temporary register and the immediate data move to the accu or temporary register. The arithmatic operations are AND, OR, ADD and SUB. These instruction only consider the ALU, accu and temp. reg. As mentioned before, a register-register transfer always uses the temp. reg. as an intermediate stocking area. A register register transfer takes always 2 microinstructions. As can be seen below where the instructions are fully decomposed, the branch group has as far as the jump and call concerned, the address in the micro word just after the instruction, allowing a branch through the whole program of 1 k words. Because of the construction of the branches, they always take 2 microcycle sequences and one if the condition is false in the case of a return. If the condition is false with a CRET, the stack needs no decrement, so we win a cycle sequence. With a CJUMP and a CCALL the jump and subroutine addresses are present after the instruction and we skip these address by simply incrementing the program counter.

In the same way a process change, the last instruction used after a program sequence, always takes 2 micro instruction times, because the instruction just read from the ROM is not executed and the other process instruction takes a certain time to be presented from the pipeline.

All instructions are so constructed that no more than 10 bit wide words are needed. The word "source" is an indication of the source of a data nibble. "mask" is the selection code of one of 4 bits. Condition true or false is selected with a true/false bit. Dest. is the indication of the destination of the data produced, and can be the accu or temp. reg., or another register. This is directed by a direction bit in the case of a move instruction, when the data flow direction is not clear as it is in the case of for instance a arithmatic instruction, or a CJUMP.

Branch group:

CJUMP	11	xxxxx	yy	z
	instr. code	source reg.	mask	true/false condition
ADDRESS	qq qqqq qqqq			
	10 bit branch address			

CCALL	10	xxxxx	yy	z
	instr. code	source	mask	true/false
ADDRESS	qq qqqq qqqq			
	10 bit subroutine address			
CRET	01	xxxxx	yy	z
	instr. code	source	mask	true/false

Move group:

MOV	001	v	w	xxxxx
	instr. code	accu/ temp.	dir.	reg. source/dest.

The register being source or destination is directed by the w bit. A one is the accu/temp as destination. The v bit selects with a one the accu.

MVI	0001	v	tttt
	instr. code	accu/ temp.	immediate data nibble

Arithmatic group:

AND	00001	00	---
OR	00001	01	---
ADD	00001	10	---
SUB	00001	11	---
	instr. code	operation	

Process change:

CHANGE	000001	----
	instr. code	

As can be seen from the coding, the decoding for the Rx/Tx control timing is very simple, and can be done with an or array or if the code is inverted with an and array. This makes the program execution not depending on the instruction decoding as far as time is of concern. The addresses of the registers involved in a data operation are present in the microinstruction and can be presented to the address bus without additional decoding. The branch addresses need decoding neither.

The arrangement of the bitgroup in a microcode can be hustled if this simplifies the hardware, but the microcode must contain the information mentioned.

7.4 Microcode execution and timing

The Rx/Tx timing control has the following outputs:

- Pipeline, clocks the data from the ROM in.
- Condition selection, clocks condition true/false in.
- Stack pointer increment.
- Stack pointer decrement.
- Next address multiplexer choice, 2 bit.
- Clock next address+1 in the program count register Tx.
- Clock next address+1 in the program count register Rx.
- Selection bit for the accu/temporary register selection.
- Read/write bit for the accu/temporary register.
- Clock data in accu/temp. reg.
- Enabling for the 5 bit register address output.
- Read/write bit for the registers.
- Clock data in registers.
- Output enable for selected registers or accu/temp. reg.

Inputs for the timing control are:

- Data flow direction from the pipeline, see paragraph 7.3
- Instruction cycle selection, 7 possible instructions from the pipeline.
- Condition from the condition selector.

Besides there are a signal for the latch after the accu, which is the temporary storage area in case of an ALU

operation. This signal is referred to as Latch accu. The signals mentioned are all appearing in the execution of the instruction from the previous paragraph. In drawing 22 all possible timing is shown. As can be seen, the 5 cycle timing sequence is mostly the same for all instructions, the only difference between instructions can be the direction of the data flow, and the therefore needed other activated selection-lines. In certain circumstances some lines are surpressed during one timing sequence. The generating of the timing can be done by a simple counter and an AND array generating the different timing for the control lines, and on every line an enabling network, surpressing the pulses if the microcode says to do so.

7.5 Controller initialisation

The only initialization is done with power-on reset, when the program count registers are reset to zero and the resetting of the timing control is done. The exact reset procedure will be hardware-dependent and is to be designed when the technology of the chip is chosen. A code of all zeros would be convenient because this could represent the NOP (no operation) and the first instruction from the pipeline then simply is a NOP if the pipeline is also reset to all zeros during power-on. The controller has in its program ROM the reset procedure for high and low level 2 and for the initialization of level 1. This starts after a reset.

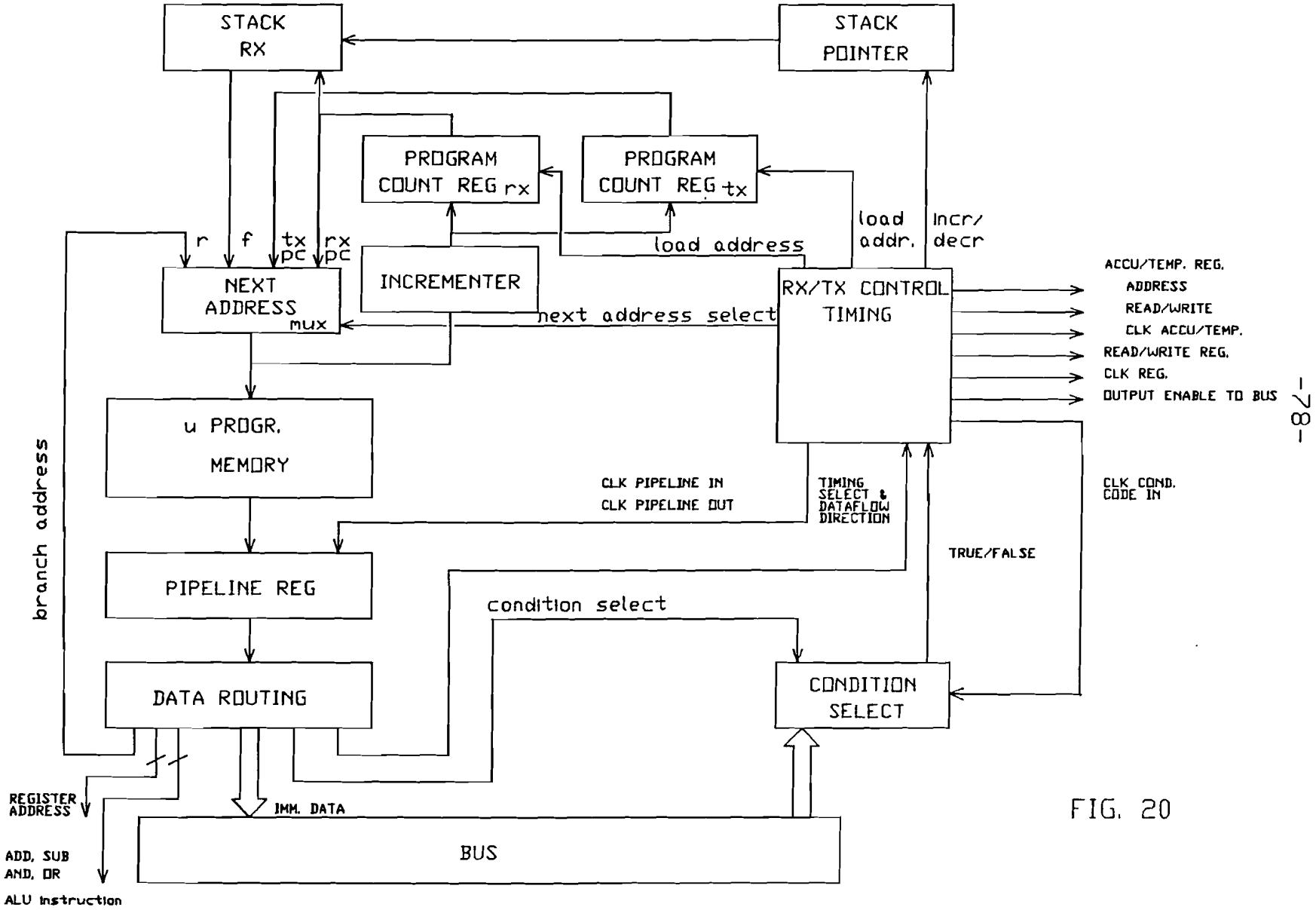

STACK RX
STACK POINTER
PROGRAM COUNT REG rx
PROGRAM COUNT REG tx
load addr.
incr/ decr
r
f
tx pc
rx pc
load address
INCREMENTER
NEXT ADDRESS mux
next address select
RX/TX CONTROL TIMING
ACCU/TEMP. REG. ADDRESS
READ/WRITE
CLK ACCU/TEMP.
READ/WRITE REG.
CLK REG.
OUTPUT ENABLE TO BUS
u PROGR. MEMORY
branch address
CLK PIPELINE IN
CLK PIPELINE OUT
TIMING SELECT & DATAFLOW DIRECTION
CLK COND. CODE IN
TRUE/FALSE
PIPELINE REG
condition select
DATA ROUTING
CONDITION SELECT
REGISTER ADDRESS
IMM. DATA
BUS
ADD, SUB AND, OR
ALU instruction

FIG. 20

-78-

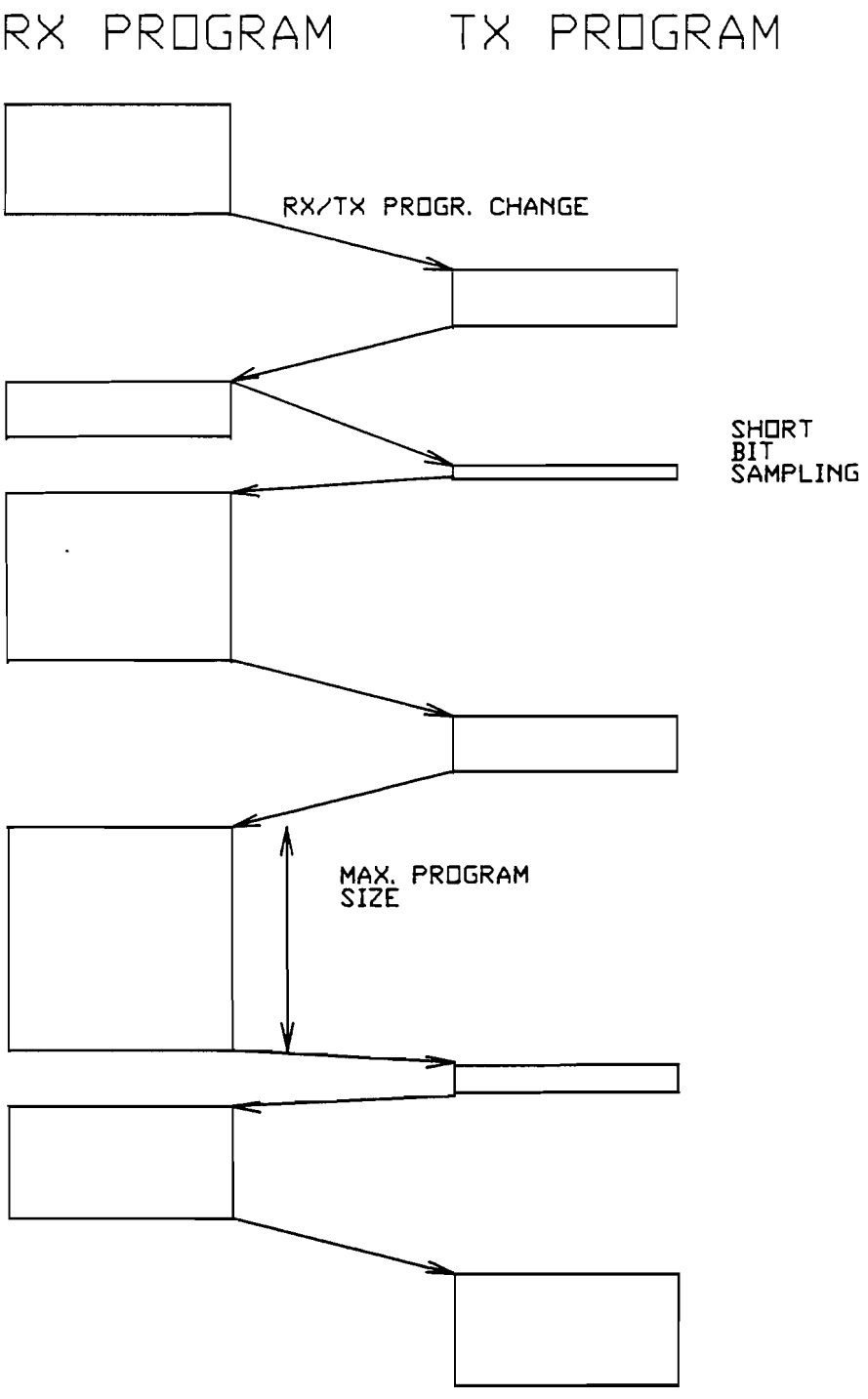

RX PROGRAM
TX PROGRAM
RX/TX PROGR. CHANGE
SHORT
BIT
SAMPLING
MAX. PROGRAM
SIZE

FIG. 21

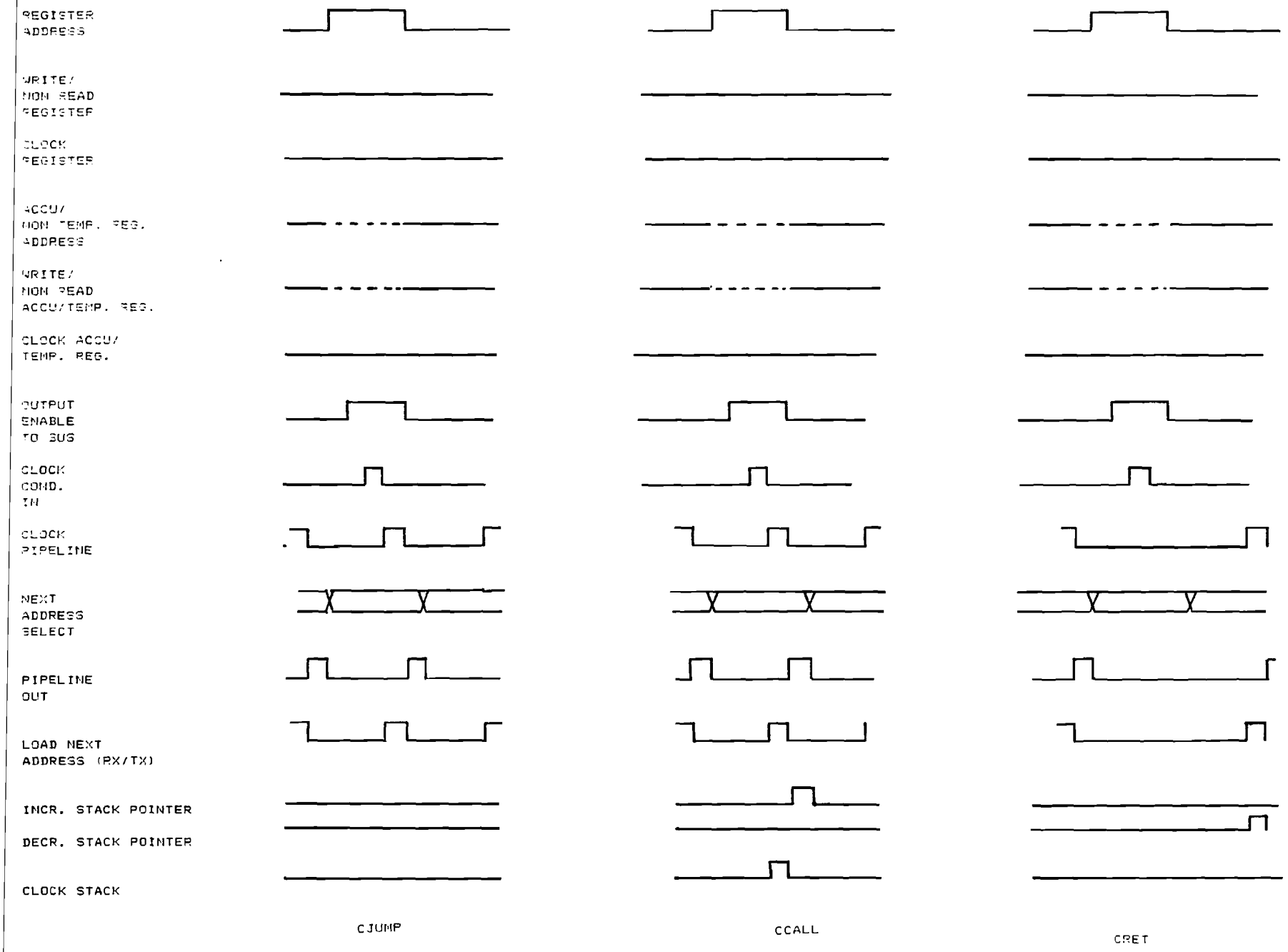

REGISTER ADDRESS
WRITE/ NON READ REGISTER
CLOCK REGISTER
ACCU/ NON TEMP. REG. ADDRESS
WRITE/ NON READ ACCU/TEMP. REG.
CLOCK ACCU/ TEMP. REG.
OUTPUT ENABLE TO BUS
CLOCK COND. IN
CLOCK PIPELINE
NEXT ADDRESS SELECT
PIPELINE OUT
LOAD NEXT ADDRESS (RX/TX)
INCR. STACK POINTER
DECR. STACK POINTER
CLOCK STACK
CJUMP
CCALL
CRET

FIG. 22a

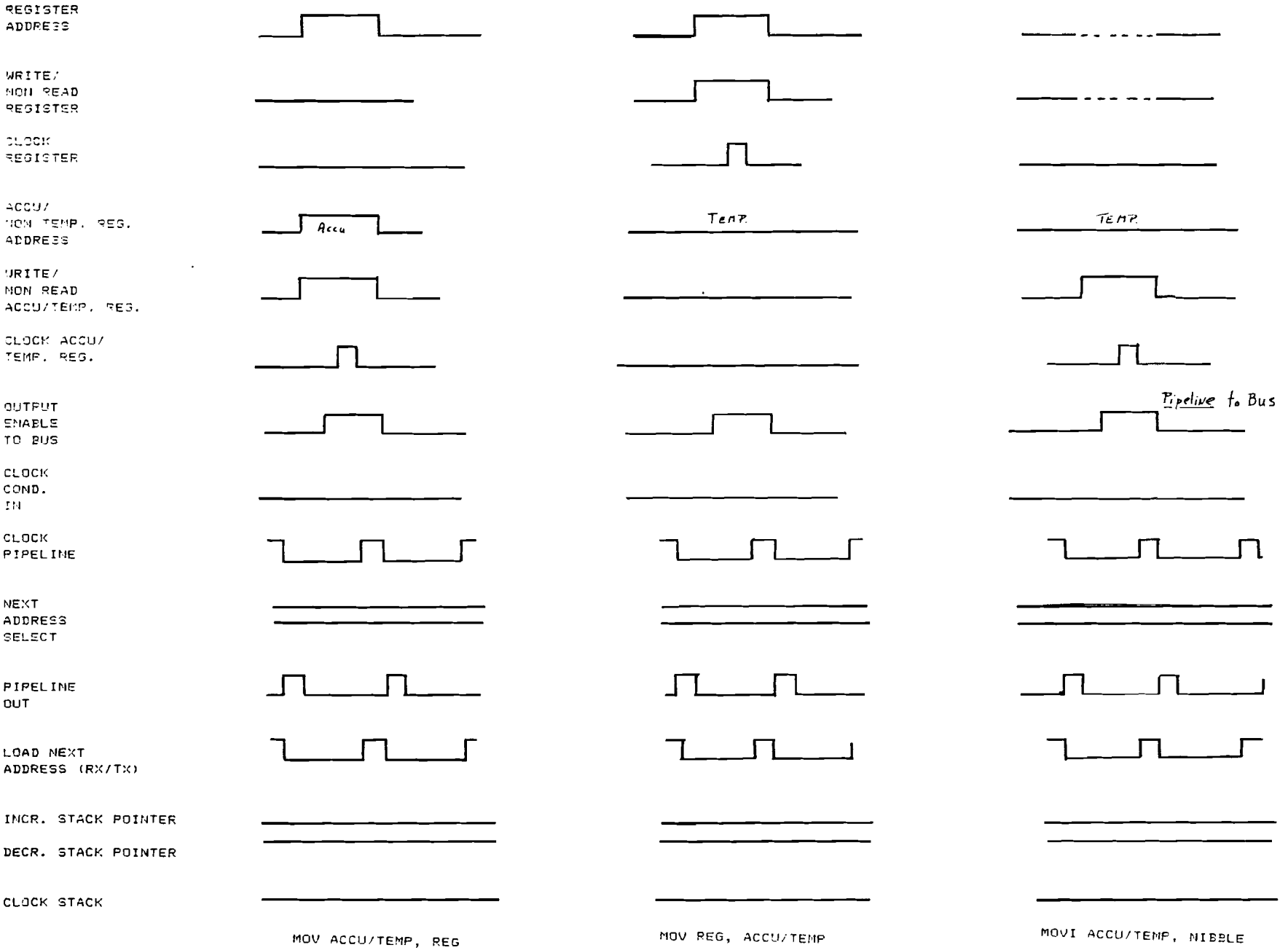

REGISTER ADDRESS
WRITE/ NON READ REGISTER
CLOCK REGISTER
ACCU/ NON TEMP. REG. ADDRESS
WRITE/ NON READ ACCU/TEMP. REG.
CLOCK ACCU/ TEMP. REG.
OUTPUT ENABLE TO BUS
CLOCK COND. IN
CLOCK PIPELINE
NEXT ADDRESS SELECT
PIPELINE OUT
LOAD NEXT ADDRESS (RX/TX)
INCR. STACK POINTER
DECR. STACK POINTER
CLOCK STACK
Accu
Temp
TEMP
Pipeline to Bus
MOV ACCU/TEMP, REG
MOV REG, ACCU/TEMP
MOVI ACCU/TEMP, NIBBLE

FIG. 22b

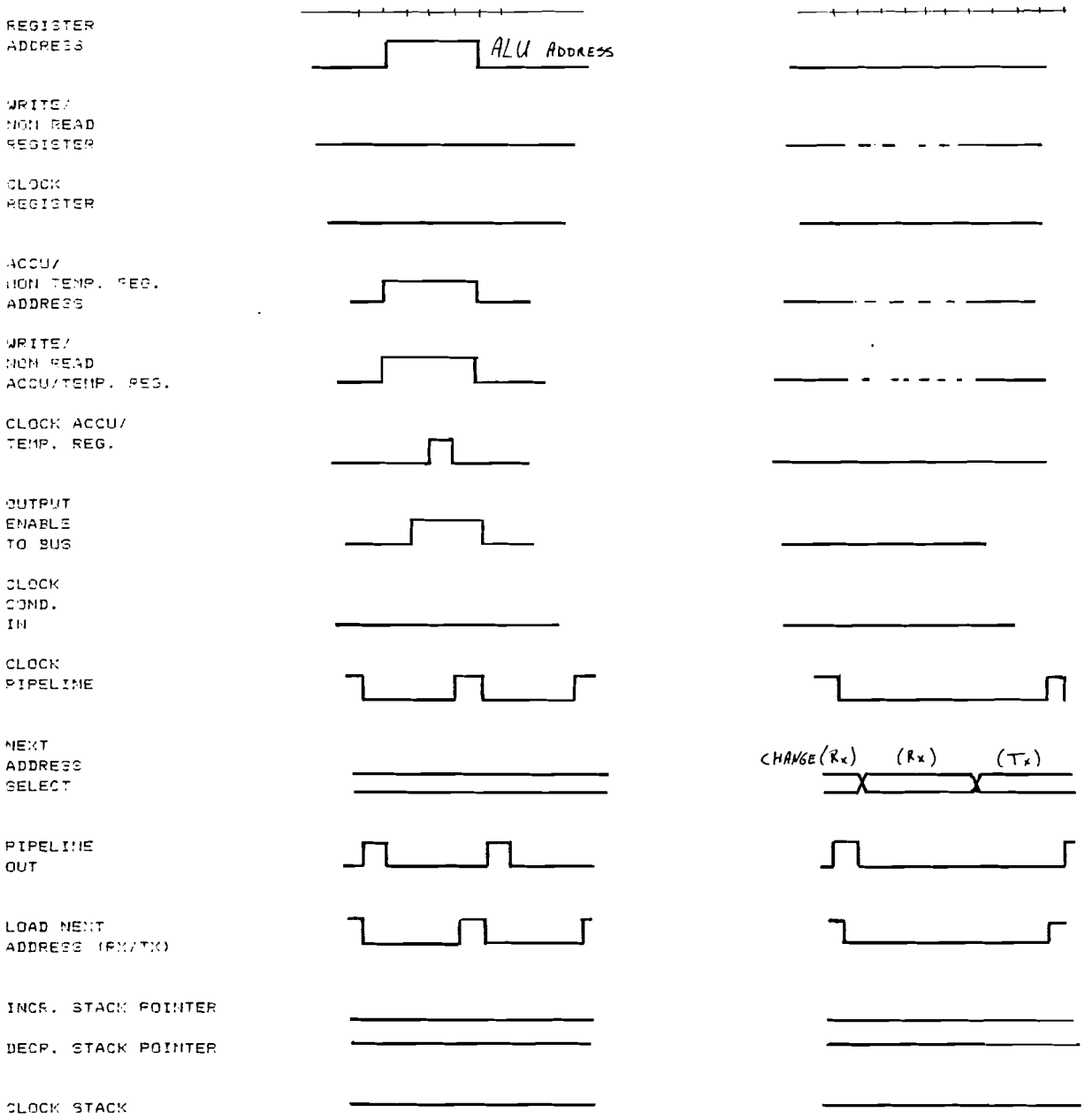

REGISTER ADDRESS
ALU ADDRESS
WRITE/ NON READ REGISTER
CLOCK REGISTER
ACCU/ NON TEMP. REG. ADDRESS
WRITE/ NON READ ACCU/TEMP. REG.
CLOCK ACCU/ TEMP. REG.
OUTPUT ENABLE TO BUS
CLOCK COND. IN
CLOCK PIPELINE
NEXT ADDRESS SELECT
CHANGE (Rx) (Rx) (Tx)
PIPELINE OUT
LOAD NEXT ADDRESS (RX/TX)
INCR. STACK POINTER
DECR. STACK POINTER
CLOCK STACK
AND/OR/ADD/SUB
PROCESS CHANGE

FIG. 22c

8 Microcode program

In figure 17, chapter 5, the states and actions of the high level 2 Rx/Tx controller are shown. The microinstructions which must perform these actions are discussed in chapter 7. The hardware is also explained, the only thing that remains is the translation of the state diagrams into machine code. For this we can use Nassi Shneiderman (N.S.) diagrams and assembly language.

8.1 Nassi Shneiderman

N.S. diagrams are another form of the state diagrams of chapter 5. The N.S. diagrams can easily be translated into Pascal or into machine code. In general a test is made (If-then-else statement) and the program is separated into 2 columns. In the first column are the statements to be executed in case the answer to the if statement was yes. The other column contains the statements in case the answer was no. The case-of statement is essentially a repeated If-then-else statement and therefore so inplemented. A few examples of N.S. diagrams are in figure 23.

The advantage of an intermediate N.S. diagram between the state table of figure 17 and the machine code of figure 24 is the possibility to get an overview of the statements with comments that is more explaining than the the state diagram and more readable than microcode (assembler). Not only programs, but also finite state machines can be put in N.S. diagrams.

8.2 Tx assembler program

The transmitter actions are already mentioned. The actions of the transmitter are indicated in figure 17 by an addition "Tx". The state changes and other actions are always proceded by testing a bit in a register. Every code execution is interrupted by a instruction CHANGE. This switches the processor to the Rx program. The CHANGE code is inserted in the Tx program after every 20 instructions or so as a rule, after less instructions if it is convenient. Some critical actions cannot be divided and therefore the CHANGE comes after the critical section, causing a non-regular CHANGE insertion in the microcode program. To make it easy

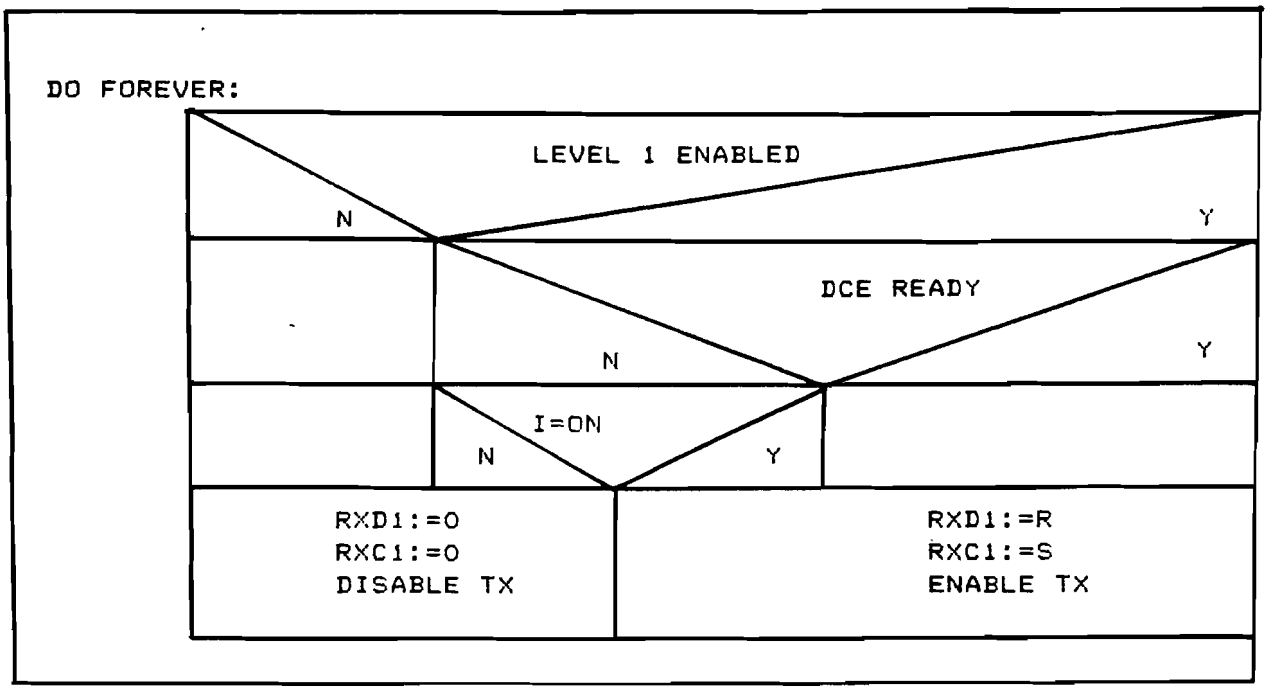
programming, an assembler for the microcode should be made. This enables fast program changing during hard- and software testing. The program in the next pages is not tested and has no pretention to be complete. It is just an example how the microcode can be used.

N.B.: the Tx program is started after the Rx initialisation. The Tx initialisation is the second program sequence after reset. Only one CHANGE has been executed at the beginning of the Tx program. The transmitter is a slave of the receiver and is only allowed to transmit I-frames if the Rx allows it. U and S frames are prepared by the receiver and placed in the next-to-send registers. The MOVI REG,DATA is a macro of the MOVI and MOV instruction.

TX_PROGRAM

```
RESET:  MOVI    V(S),#0                 ;CLEAR V(S) AND V(R)
        MOVI    V(R),#0                 ;
        MOVI    LASTSENT1,#0            ;CLEAR LASTSENT
        MOVI    LASTSENT2,#0            ;
        MOVI    LASTACKNV(S),#7         ;SHOULD BE V(S)-1
        MOVI    STATUS,#0               ;STATUS BITS ZERO
        MOVI    PROGRAMSTATUS1,#0       ;
        MOVI    PROGRAMSTATUS2,#0       ;
        MOVI    TIMERT1,#1              ;STOP TIMER
        MOVI    RETRN2,#0               ;RESET COUNTER
        MOVI    COMMREG4,#0             ;COMM REG ZERO
        MOVI    COMMREG3,#0             ;COMM REG ZERO
        MOVI    COMMREG2,#0             ;COMM REG ZERO
        MOVI    COMMREG1,#0             ;COMM REG ZERO
        MOVI    COMMANDTX1,#1010B       ;SEND IDLE
        MOVI    COMMANDTX2,#0           ;NO OTHER COMMANDS
        MOVI    LEVEL1ENABLE,#1000B     ;ENABLE LEVEL 1
;
;THE TRANSMITTER IS NOW INITIALIZED AND LEVEL 1 IS ENABLED
;
INSTAT: CHANGE                          ;SWITCH TO RX
        JUMPF   CONNREG,#1000B          ;TEST IF A LINK IS
                INSTAT                  ;ENABLED
        MOVI    COMMANDTX1,#1000B       ;SEND FLAGS
;
;THE LOGICAL LINK IS NOW ENABLED AND FLAGS ARE BEING
;TRANSMITTED
;
DISCON: CHANGE                          ;SWITCH TO RX
        JUMPT   PROGRSTAT1,#0010B       ;TX ENABLED?
                CONN                    ;
```

```
        JUMPF   NEXTTOSEND1,#1000B      ;U FRAME TO
                DISCON                  ;BE SEND?
        JUMPF   STATUSTX,1000B          ;TX READY?
                DISCON                  ;
        MOVI    COMMANDTX,#0000B        ;START FRAME
;
;TRANSMIT U-FRAME, TEST ON POLL, START MAX TIMER ON
;POLL, MOVE NEXT TO BE SEND BYTE TO LAST SENT BYTE
;REGISTER, TEST IF A FRMR FRAME HAS TO BE TRANSMITTED,
;ETC.
        MOVI    COMMANDTX,#1000B        ;PREP. FRAME END
        JUMPT   FLAGS,#0001B            ;JUMP ALWAYS
                DISCON                  ;BACK
CONN:   CHANGE                          ;SWITCH TO RX
        JUMPF   COMMAND,#0001B          ;TEST ABORT
                NOABORT                 ;
        MOVI    COMMANDTX1,#1100B       ;ABORT TX
        JUMPT   FLAGS,#0001B            ;JUMP ALWAYS
        CONN                    ;BACK
NOABORT:JUMPF   PROGRSTAT1,#0010B       ;TX ENABLED?
                DISCON                  ;BACK
        JUMPF   STATUSTX,#1000B         ;TX READY?
                CONN                    ;BACK
        JUMPF   NEXTTOSEND1,#1000B      ;S-FRAME TO
                NOSFRAM                 ;TRANSMIT?
        MOVI    COMMANDTX,#0000B        ;START FRAME
;
;TRANSMIT S-FRAME, TEST ON POLL, START MAX. TIMER
;ON POLL, MOVE NEXT TO BE SEND BYTE TO LAST SENT
;REGISTERS, ADD N(R) TO S FRAME, ETC
        MOVI    COMMANDTX,#1000B        ;END FRAME
        JUMPT   FLAGS,#0001B            ;JUMP ALWAYS
                CONN                    ;BACK
NOSFRAM:JUMPF   COMMAND,#0010B          ;TEST PACKET
                NOIFRAM                 ;READY
        MOVI    COMMANDTX,#0000B        ;START TX
;
;TRANSMIT I-FRAME, UPDATE V(S), START MAX TIMER ON
;POLL, CHECK MAX. OUTST. PACKETS, ETC.
        MOVI    COMMANDTX,#1000B        ;END FRAME
        JUMPT   FLAGS,#0001B            ;JUMP ALWAYS
                CONN                    ;BACK
NOIFRAM:MOV     ACCU,COMMREG4           ;GET REQ. FLAG
        MOVI    TEMP,#1000B             ;FOR PACK.
        ADD                             ;SET FLAG
        MOV     COMMREG4,ACCU           ;FLAG TO LEV.3
        JUMPT   FLAGS,#0001B            ;JUMP ALWAYS
                CONN                    ;
```



LEVEL 1 RECEIVER

FIG. 23a

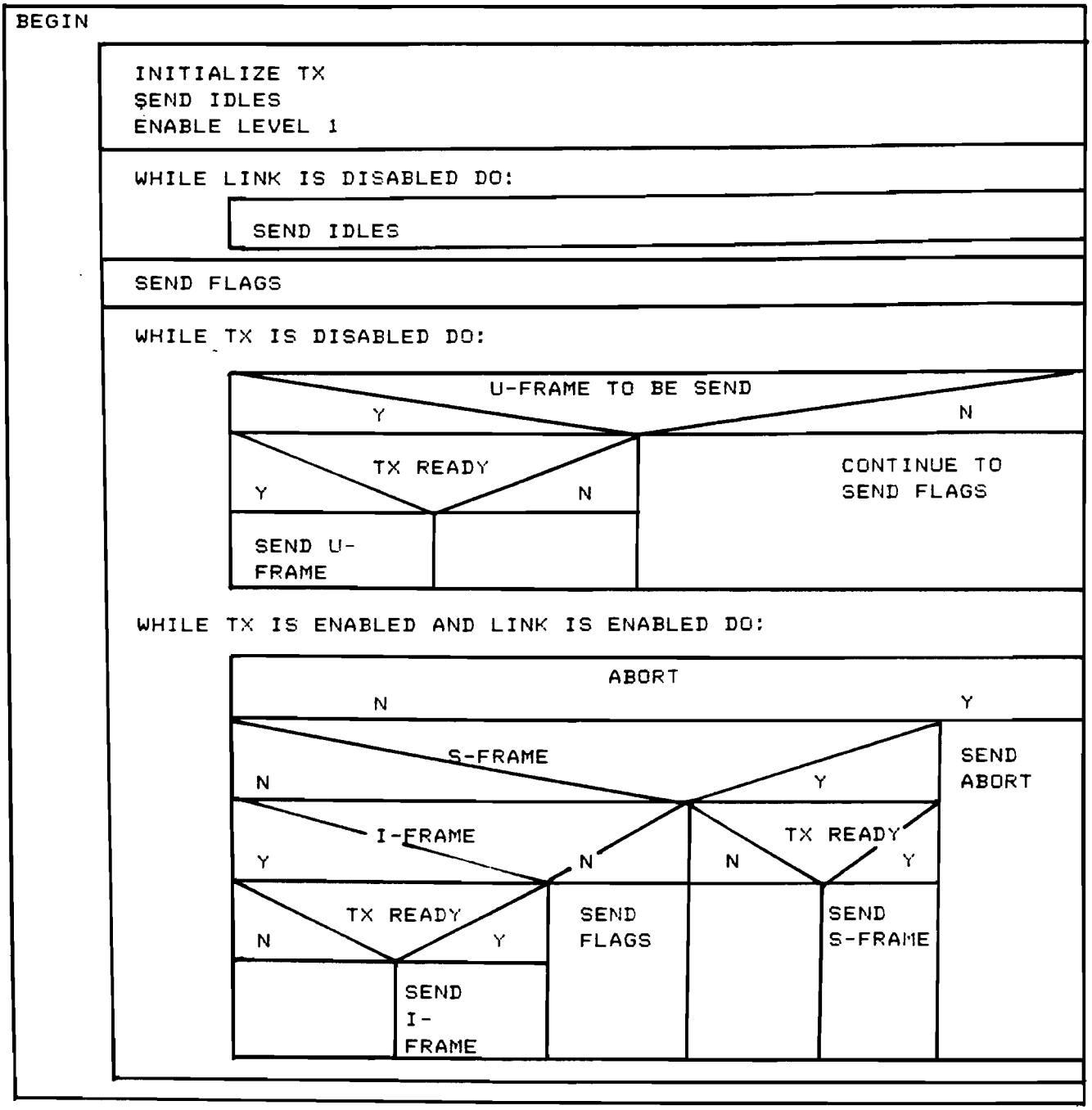


FIG. 23b

LEVEL 2 TRANSMITTER PROGRAM

9 Testability

After manufacturing the chip, the co-processor has to be tested. Because of the construction with a lot of registers, several processors and several finite state machines, this is quite complex. In general, a design can be considered being constructed as shown in figure 24. Memory elements connected by random logic, one or more clocks and a few pins to the outer world.

To test the co-processor in a certain state is difficult because of this construction. The separation of the chip in 3 or more sub-designs, for instance level 2, level 3 and the DMA unit, is a great advantage. The units to be tested are much smaller then. To bring the processors in a requested state, with the requested variables in the registers is the second problem after partitioning the logic to be tested. First to bring the processor in a certain state, and after that the reading of the next state can only be done in a normal configuration with the help of a lot of extra wires. Adding this wires is not only fault chance increasing (more line patterns on the chip), but also impossible because of the number of extra pins this would require. For the factory test (a very expensive element in the production) a very simple solution is the scan test.

9.1 Factory test.

Figure 25 shows a re-arrangement of figure 24. The design is considered as a block of memory cells (registers) and an array of logic. Every state can be set and read out as in the previous case by loading and monitoring the memory cells. The scan path is the solution to our problem of to much wires. All memory cells can shift their contents into their neighbour. Because a lot of registers in level 2 and 3 already are shift registers, only little extra hardware is required for the rest of the registers and the interconnection. The register contents can be serially clocked out in a bit sequence via an extra pin, or a pin with two functions. The scan mode can for instance be set by supplying a high voltage to a pin or software by setting a bit in a test register inside the chip. This enables us to read the processors state. The reverse operation is also possible and is done by clocking in the requested processor state.

Of course, selecting the scan path as a test is not the

solution to the problem of testing the chip. The choice of patterns is important too. As mentioned before, H. van Ooijen will study the testing of this design.

9.2 Go/no go test

Because of the complexity of the design, and the program length, not all states can be tested properly by the factory testing device. There are simply to much patterns possible. If the patterns are analized it can be seen by most designs, that with a few patterns a lets say 65% fault coverage is possible. The coverage in relation to the applied patterns is mostly exponential. Detection of an additional fault is only reachable at the cost of far more patterns than the detection of the first fault required. When the testing device is expensive, the possibility of self test is very nice. Lets assume that the factory testing device tests the correct working of the processor nucleus and the processor, after this, tests the rest of the registers and logic by read/write test sequences and signature analysis. This would only require a power supply for the chip and a very simple indication for the ok or not-ok report from the chip. All the tests mentioned are ment to test the chip in such a way that the customer is supplied with a 99.99% good co-processor. If the chip is suspected to be partly defect, after installation in a system, the customer has to apply his own tests. The functional tests, testing if the chip is doing what we want, are tests to be done before mass production and are more or less software testprograms and not hardware fault detection problems.

9.3 Testing by the host.

Only a few things can be tested when the chip is already installed in a system. The tests that can be done by a user are the loopback tests, internal as well as external as described in chapter 2. If the final design allows it, an additional test initiated by the host, can be done by the co-processor itself by calling a part of the self test program. Figure 26 shows a few possibilities on level 1 and 2 testing. If the host is testing, the scan path test can not be used because of the complex additional test equipment required.

Example

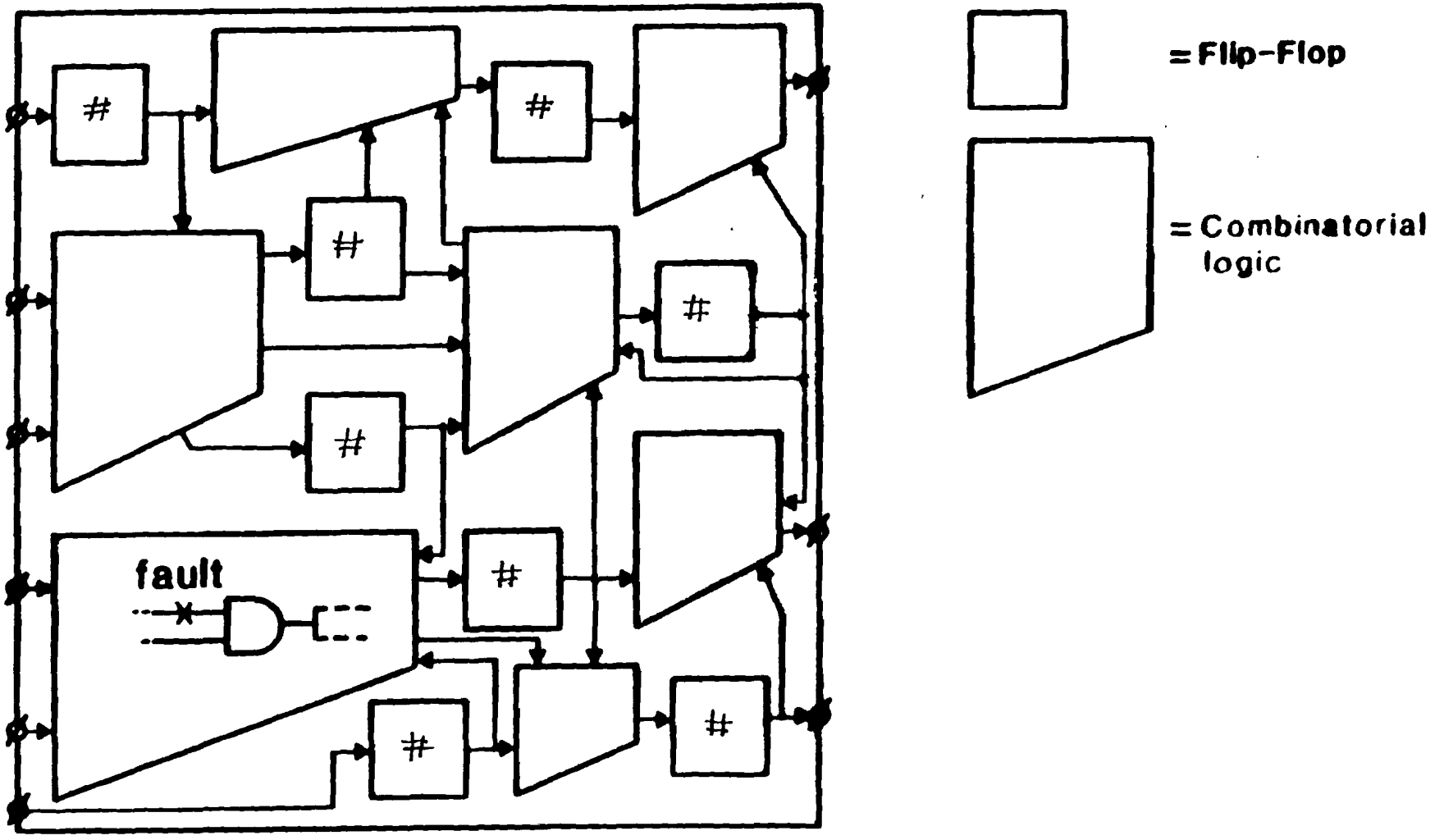
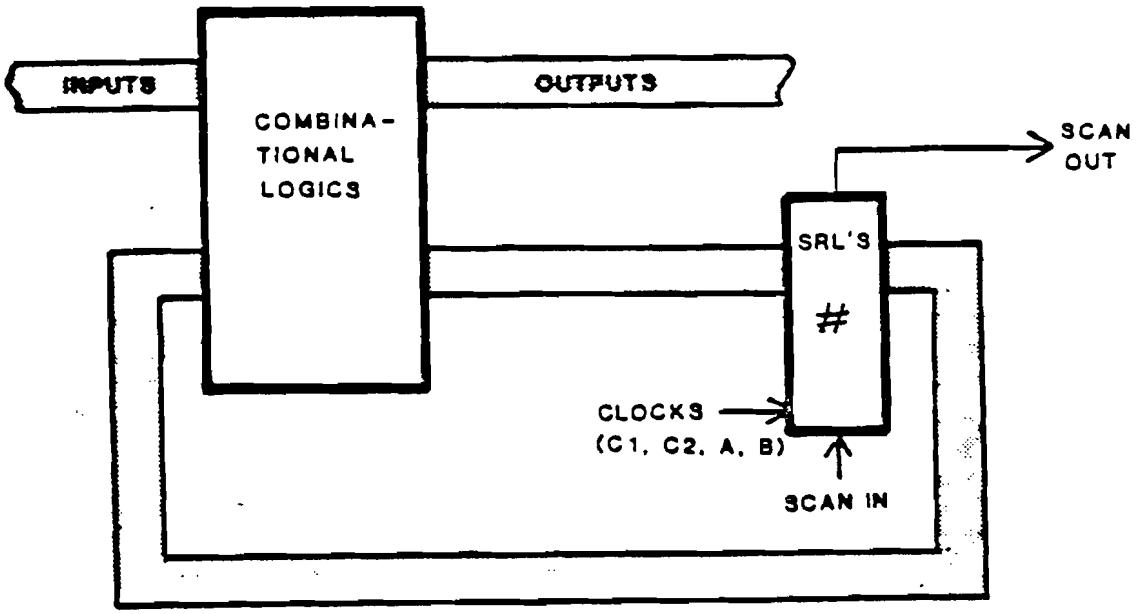


FIG. 24


INPUTS
COMBINA-
TIONAL
LOGICS
OUTPUTS
SCAN
OUT
SRL'S
#
CLOCKS
(C1, C2, A, B)
SCAN IN

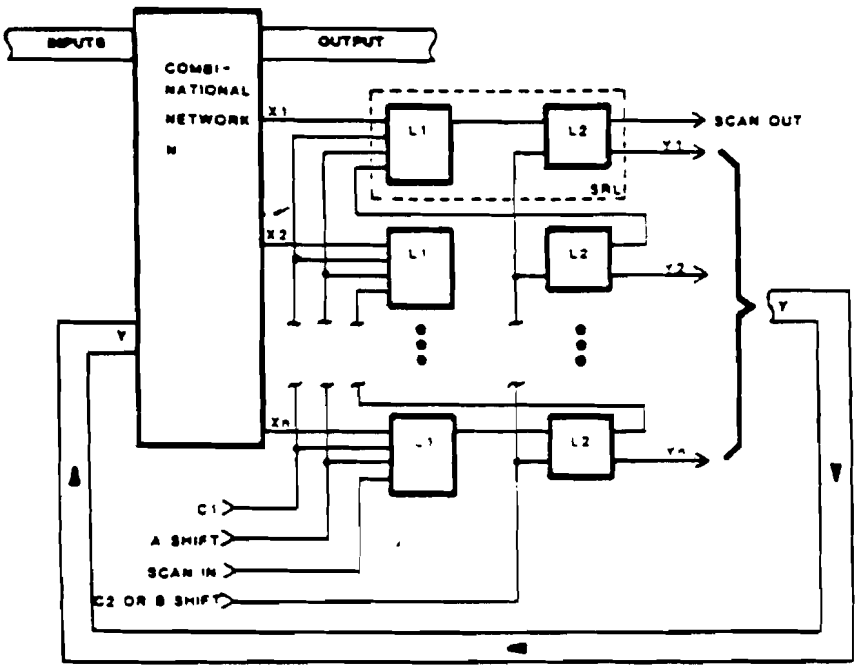
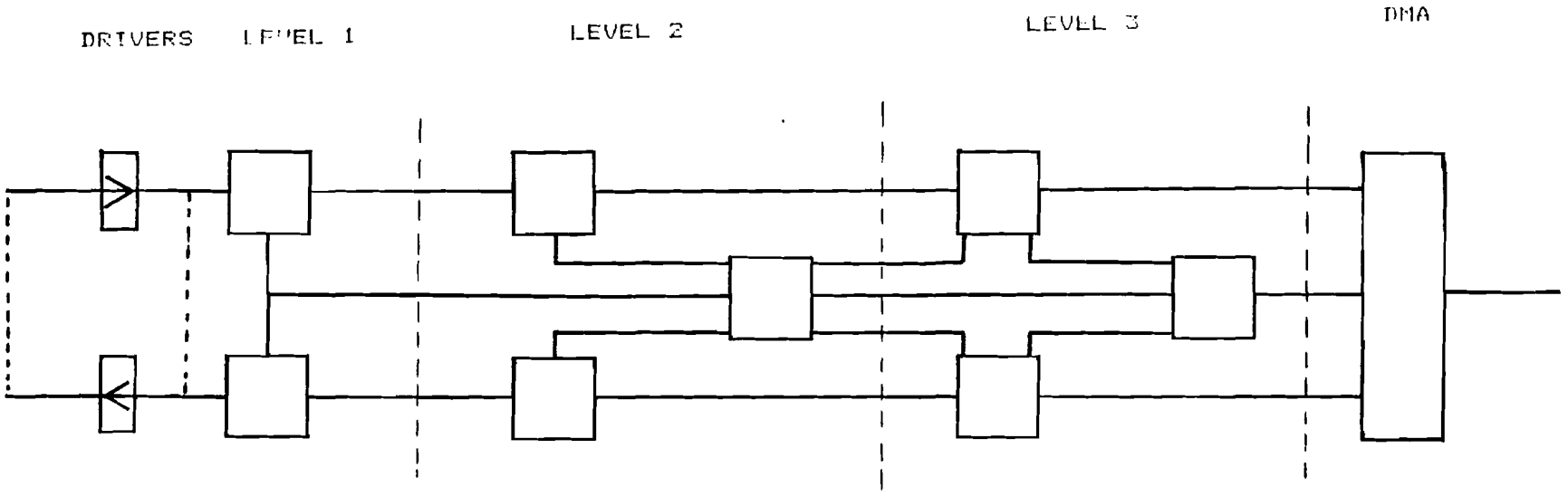

INPUTS
COMBI-
NATIONAL
NETWORK
N
OUTPUT
X1
X2
Xn
L1
L2
SCAN OUT
Y1
Y2
Yn
Y
SRL
C1
A SHIFT
SCAN IN
C2 OR B SHIFT

FIG. 25



EXTERNAL LOOPBACK

INTERNAL LOOPBACK

SELFTEST (MICROCONTROLLER)
+
SCANPATH
+
ROM CHECKSUM
+
SIGNATURE ANALYSIS

SELFTEST (MICROCONTROLLER)
+
SCANPATH
+
ROM CHECKSUM
+
SIGNATURE ANALYSIS

FIG. 26

10 Conclusions

As far as levels 1 and 2 are concerned, the functional design is completed. Parts of level 2 have been simulated in HDL on a VAX computer and can be translated in hardware. The software only needs the last stage of micro programming, the states and state changes are also complete. A realization will only depend on the number of gates involved. For level 1 we can estimate:

Level 1 Rx:	10 gates
Level 1 Tx:	10 gates

For level 2:

Low level 2 Rx:	40 gates 50 bit registers
Low level 2 Tx:	40 gates 50 bit registers
High level 2	90 gates 180 transfer gates 164 bit registers
Microcontroller	96 bit registers 12 kbit ROM

The level 3 design is not discussed here. Until now, no estimation of the total number of gates and registers can be given because level 3 is not fully drawn on gate and register level.

For level 2 it looks as if it is possible with the current techniques to realize the chip in a single 40 pin chip package.

If the chip can work on a 25 MHz clock (internal) and the bit transfer rate is 64 kbit/sec, the performance of the chip would be quite good. The throughput can reach this 64 kbit/sec because of the layering of the design. The only concession to X.25 is not the deliverance of always 100% ok packets to level 3 by level 2, but the deliverance of the data with additional information about the packet being

alright or not.

The final researches should be made into the testability of the design, the hard and software simulation, and the host-processor software. All these are being researched (august 1985) and are thought to be completed in early 1986.

Literature

(1) CCITT Yellow book
Volume VIII Fascicle VIII.2
Recommendations X.25 and X.21
Geneva 1980

(2) Mick J. and Brick J.
Bit slice microprocessor design
New York: McGraw Hill 1980

(3) Bipolar microprocessor logic
interface databook
Advanced Micro Devices Inc. 1982

(4) X.25 DTE/DCE interfaces
Philips Data Systems

(5) Hill F.J. and Peterson G.R.
Introduction to switching theory and logical design
Wiley, 3th edition

(6) MCS 80 Users manual
Intel Corporation

(7) CCITT Yellow book
Volume VIII Fascicle VIII.1
Recommendation of the V series.
Geneva 1980

(8) CCITT Yellow book
Volume VIII Fascicle VIII.3
Recommendation X.40-X.180
Geneva 1980

(9) Specification X.25 interface
Td. 334 Edition 6
Dutch PTT

(10) Design of a X.25 co-processor
level 3 implementation
Schenkelaars H.P.M.J.
TH Eindhoven 1985

Figure list