PLEASE RETURN TO
J. H. COGHLAN
AARSYD

OVERVIEW OF THE MULTIPLEXING SUBSYSTEM FOR THE CYBER 70 COMMUNICATIONS SYSTEM

E. P. Lamoureux
11/19/73

TABLE OF CONTENTS

1.0 INTRODUCTION

This document presents a basic description of the Multiplex Subsystem that is being developed for the CYBER 70 Communications System. The Multiplex Subsystem contains the hardware/firmware/and software elements necessary to provide the data and control paths for information flow between communication lines and user program software. Many of the line/protocol dependent functions that were performed by fixed hardware in previous systems are implemented in common alterable firmware/software allowing reduced development and reoccuring hardware costs for each line and protocol that is added to the communications system.

The primary task performed by the Multiplex Subsystem is to receive data from many communications lines and distribute the characters to line related input buffers and to obtain characters from line related output buffers and distribute to communication lines. The subsystem provides for special table driven and dynamically controlled processing on each character as it is being distributed from a communication line. Circuit, modem and subsystem status is detected and transferred in the form of "work demands" to "user programs" for processing. Control information received from "user programs" in the form of commands to the subsystem are decoded and executed within one or more subsystem element.

The multiplexing scheme used in the Multiplex Subsystem design is based on the "demand driven" loop concept recommended by the Advanced Development Lab.

2.0 DEMAND DRIVEN MULTIPLEX LOOP

The "Multiplex Loop" concept is a demand driven mechanism for gathering input data and status from and distributing output data and control to many communications lines on a real time basis. The design was adopted as an economical method of connecting many communication lines to a "controlling processor" while using a minimum of the processor's resources to manage the mechanism. A similar loop concept has also been considered for use in a Data Block Switching Network[1] and Computer Interconnects[2].

The primary functional elements (Figure 1) of the mechanism are the Communications Line Adapters (CLA's) that accomplish character assembly/disassembly and communications line/modem interface; the Input Loop which transports data demands, data and status from the CLA's to the controlling processor; the Output Loop that transports data and control from the controlling processor to the CLA's; the Multiplex Loop Interface Adapter (MLIA) that controls the movement of data demands, data and supervision between the Input and Output Loops and the controlling processor; and the Loop Multiplexer(s) (Loop Mux or LM) that provide access to the Input and Output Loops by the CLA's. Up to 8 Loop Multiplexers each containing a maximum of 32 CLA's may be connected in one loop.

References:

1. Pierce, J. R., "Network for Block Switching of Data", BSTJ, Vol. 51, No. 6, July-August 1972, pp. 1133-1145.

2. Coker, C. H. , "An Experimental Interconnection of Computers Through a Loop Transmission System", BSTJ, Vol. 51, No. 6, July-August 1972, pp. 1167 1175.

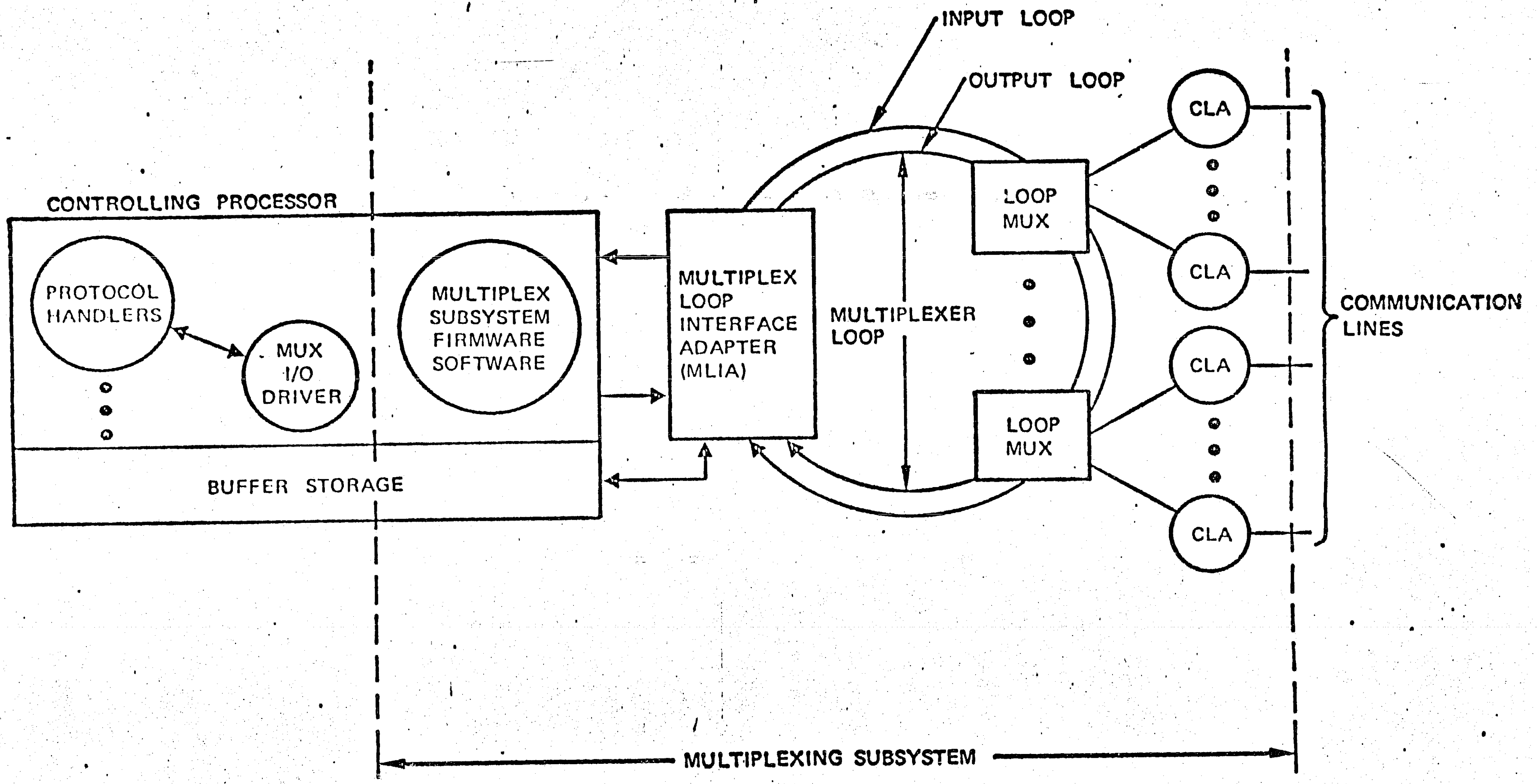


FIGURE 1. FUNCTIONAL ELEMENTS OF THE MULTIPLEX SUBSYSTEM

Multiplex action on the Input Loop occurs when the Multiplex Loop Interface Adapter issues a "loop end" control signal on the Input Loop. As the signal propagates around the loop, each Loop Multiplexer in turn has an opportunity to put data and/or supervision from the CLA's on the loop. The Loop Multiplexers that have information from one or more CLA's to place on the loop monitor the loop for the "loop end" signal, removes the "loop end" signal, place the information from the CLA's on the loop, and replace the "loop end" signal. The Multiplex Loop Interface Adapter transfers the information received on the input side of the Input Loop to buffer storage for processing by the controlling processor.

The Multiplex Loop Interface Adapter takes addressed information from buffer storage under direction of the controlling processor and transmits this information on the Output Loop. The Loop Multiplexer receives the addressed information blocks and presents the address to all CLA's. The CLA recognizing its address is selected and the information is transferred from the Output Loop to the CLA.

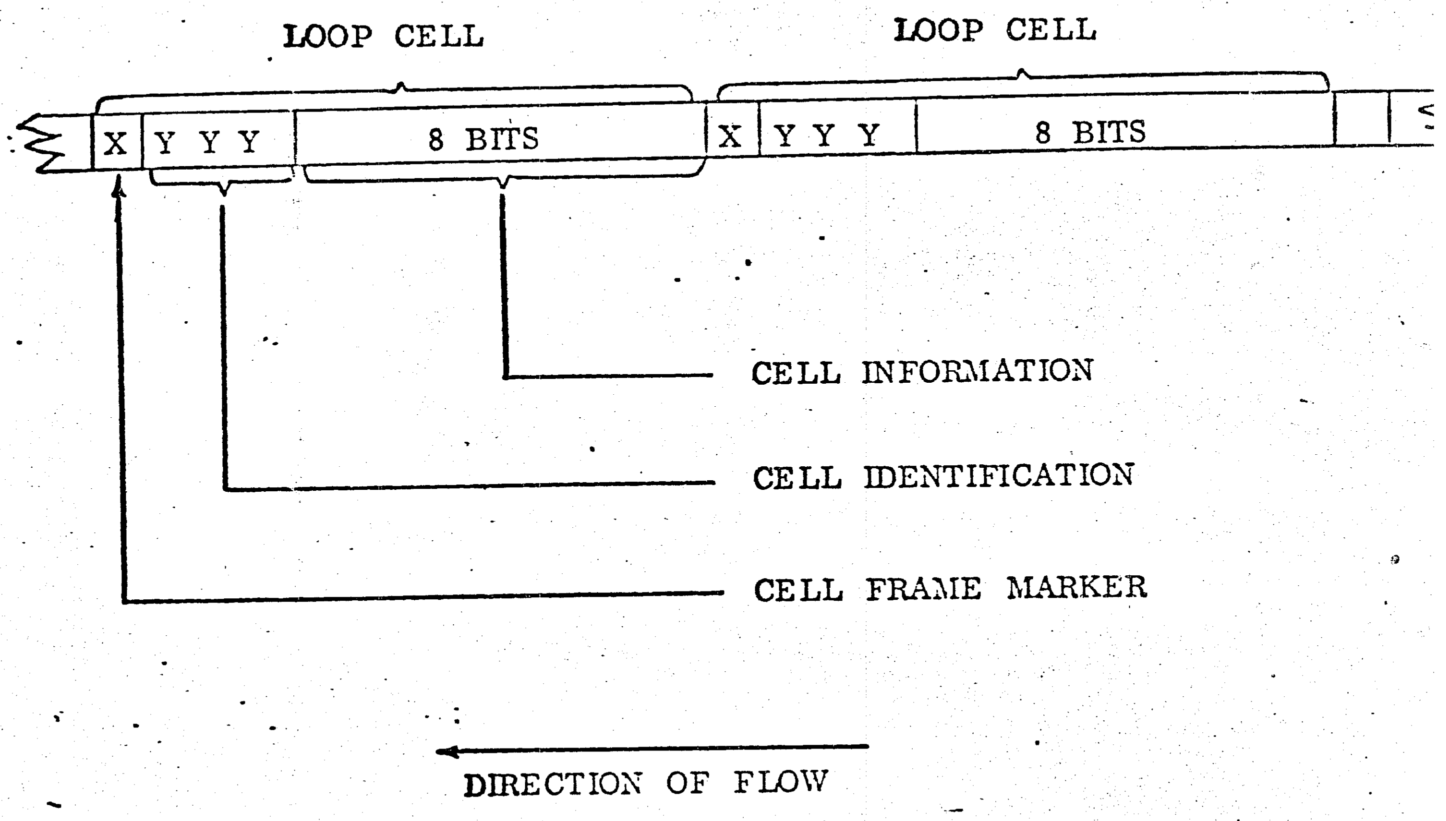
2.1 Loop Transmission

Data transmission on the multiplex loops is serial at a fixed rate of 20 MHz. The method for information framing is illustrated in Figure 2. Every twelfth bit is a cell frame marker that defines a 12-bit loop cell. The cell frame marker is followed by a cell identification field (3 bits) whose contents define the meaning of the remaining field (8 bits) of the loop cell (see Figure 3).

A contiguous group of loop cells starting with a CLA address and ending with a Cyclic Redundancy Checksum (CRCS) is called a line frame. A line frame contain data and/or supervision related to a single CLA. Line frames are made up of four different types of loop cells; "address", "data", "supervision", and "Cyclic Redundancy Checksum (CRCS)". Line frames carry data and supervision loop cells on the loops between the CLA's and the controlling processor.

The "CLA address" loop cell carries Output Data Demands (ODD's) as well as a unique binary number (address) of the CLA which is independent of the CLA's physical location within the system. An Output Data Demand is a request from a CLA to the controlling processor for data to be sent to the CLA for subsequent transmission on the communication line by the CLA.

"Data" loop cells contain up to 8 bits of information that is to be transmitted to or has been received from communication lines. The number of data characters per line frame is limited by the hardware to a maximum of 16, but best performance is obtained by sending only one instance of data and/or supervision per line frame. This will normally be one data and two supervision characters if supervision is present at all.



LOOP CELL FRAMING

FIGURE 2

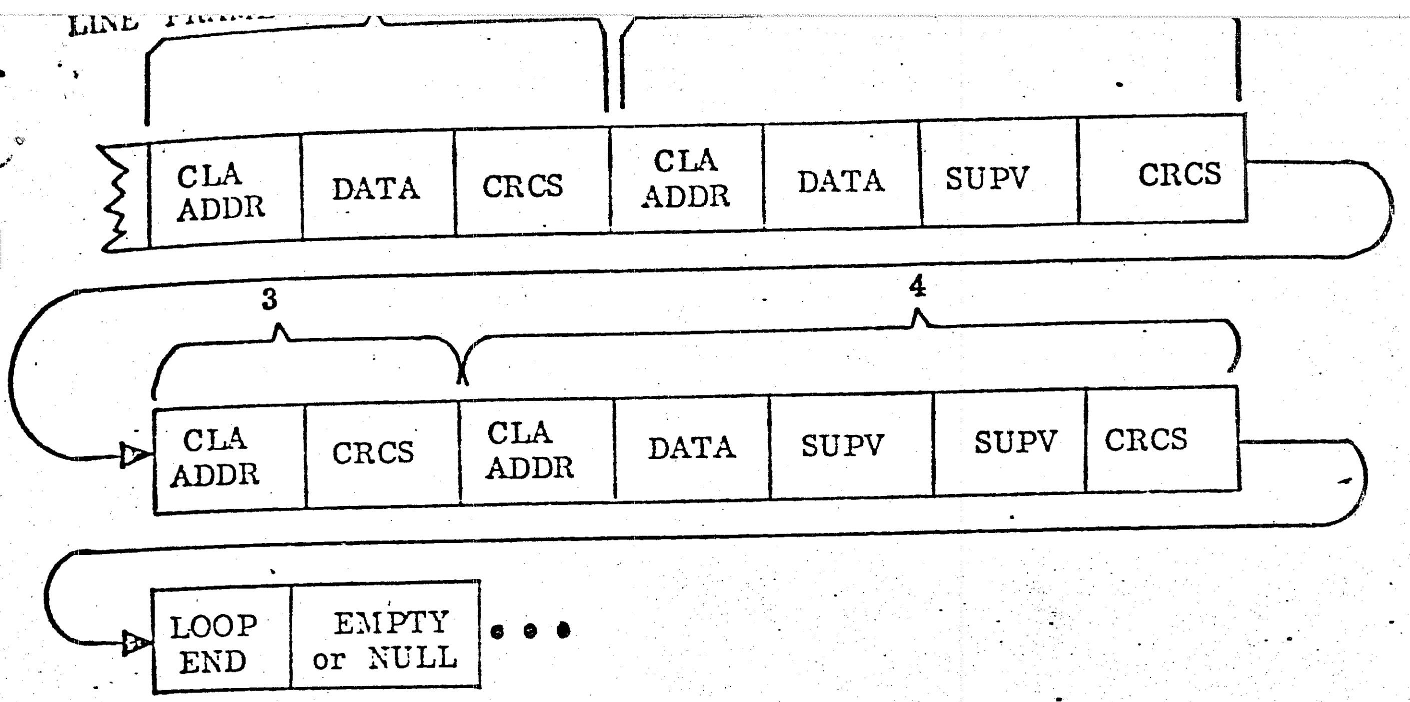
TYPE	FORMAT NO.	0	1	2	3	4	5	6	7	8	9	10	11	
		CELL MARKER	CELL IDENTIFICAT			CELL INFORMATIO								
LOOP MANAGEMENT FORMATS	0	1	0	0	0	0	0	0	0	0	0	0	0	EMPTY
	0	1	0	0	0	1	0	1	0	1	0	1	0	NULL
	1	1	0	0	1	F_3	F_2	1	1	1	0	0	0	LOOP END
	1	1	0	0	1	F_3	F_2	0	0	0	1	1	1	LOOP END-RESTA
RESERVED FORMATS	2	1	0	1	0	R	R	R	R	R	R	R	R	UNDEFINED
	3	1	0	1	1	R	R	R	R	R	R	R	R	UNDEFINED
LOOP USAGE FORMATS	6/7	1	1	1	F_1	A	A	A	A	A	A	A	A	CLA ADDRESS
	4	1	1	0	0	D	D	D	D	D	D	D	D	DATA
	5	1	1	0	1	S	S	S	S	S	S	S	S	SUPERVISION
	6	1	1	1	0	C	C	C	C	C	C	C	C	CRCS

Apply to Input Loop Only:

F_1	=	0	- No Output Data Demand
	=	1	- Output Data Demand
F_2	=		- Undefined (transmit as zeros)
F_3	=	0	- Do not input from secondary lines
	=	1	- Input from secondary lines.

A	=	CLA address; may be primary or secondary
D	=	DATA
S	=	Supervision status from CLA(s) or commands to CLA(s)
C	=	Cyclic Redundancy Checksum. Accumulation over all preceding bits of the line frame according to the polynomial $X^8 + X^7 + X^6 + X + 1$
R	=	Reserved (presently undefined)

FIGURE 3. MULTIPLEX LOOP INFORMATION FRAMING



EXAMPLE LOOP BATCH

A "loop batch" must be constructed according to the following rules:

1. May contain many line frames.
2. Each line frame must start with the CLA address and end with the CRCS.
3. Each line frame could consist of any number of DATA and/or SUPV characters, but best performance will occur if only one instance of data and/or supervision is sent within a line frame (multicharacter buffers in CLA's should be loaded or unloaded one character at a time).
4. A supervision character which is associated with a data character must follow that data character(s) immediately.
5. Null cells may appear any place within the loop batch to resolve timing problems.
6. Empty cells may appear outside the loop batch only.
7. A new line frame may be added to a loop batch on the Input Loop after detection of the "loop end" followed by at least one "empty" cell. Once the addition of a line frame has started, it may continue, replacing either "empty" or "null" cells but a new line frame may not be started unless an "empty" is present on the loop.
8. "Address", "data", and "supervision" information cells are transmitted and received on the loop as the highest order bit first.

LOOP BATCH FORMAT

FIGURE 4

and the controlling processor. Supervision may be modem, line, CLA status generated by the CLA, or CLA commands generated by the controlling processor.

The "CRCS" loop cell signifies the end of the line frame and carries an 8-bit Cyclic Redundancy Checksum on all preceding bits of the line frame according to the polynomial $X^8 + X^7 + X^6 + X + 1$.

Three loop management cell types are defined to control information flow on the loops. They are: "loop end", "null", and "empty".

The "null" loop cell is used to control the information rate of the loop and may be inserted on the loop to resolve timing conflicts. "Null" cells are normally ignored by all devices on the loop when encountered.

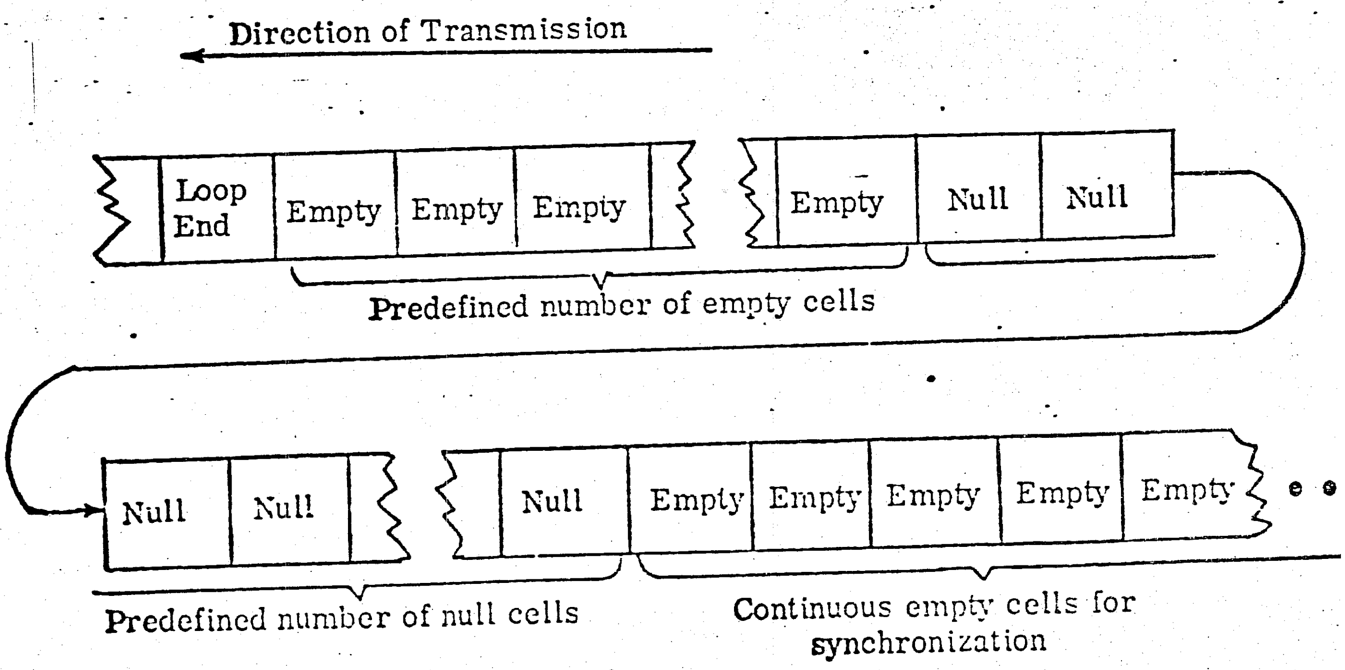
The "empty" loop cell is used to acquire and maintain loop cell frame synchronization. It also is used to signify that information line frames may be placed on the Input Loop, replacing the empty cells, provided the empty cells are preceded by the "loop end" cell.

The "loop end" cell is used to signify that information line frames may be placed on the Input Loop by the Loop Multiplexers. It also signals the end of a train of line frames. A train of line frames followed by a "loop end" cell is called a "loop batch". An example "loop batch" is shown in Figure 4.

A special "restart loop end" cell is used to signal the Loop Multiplexers and CLA's that the previous "loop batch" was received in error on the Input Loop. The restart action taken by these devices is described in paragraph 2.2. No information may be placed on the loop when a "restart loop end" cell is detected.

2.1.1 Sequence of Operation for Transfer from the CLA to the Controlling Processor (Input)

1). The MLIA generates and transmits an empty train as shown below:


Direction of Transmission
Loop End
Empty
Empty
Empty
Empty
Null
Null
Predefined number of empty cells
Null
Null
Null
Empty
Empty
Empty
Empty
Empty
Predefined number of null cells
Continuous empty cells for synchronization

EMPTY INPUT LOOP BATCH

2). A Loop Multiplexer (LM) monitors the serial bit stream for a "loop end" cell followed by one "empty" cell. All bits received are passed on to the next LM or MLIA on the serial loop.

3). When the "loop end", "empty" sequence is detected, the LM removes the "loop end" and selects each CLA that has a demand for input of "data", "supervision or "output data demand" (ODD).

4). When selected, the CLA transfers to the LM in a parallel manner one loop cell at a time starting with the "CLA address".

5). As each loop cell is received from the CLA the LM places the cell on the Input Loop replacing the cell (empty or null) that appears on the loop next.

6). The LM begins accumulation of a Cyclic Redundancy Checksum (CRCS) on each cell received from a CLA.

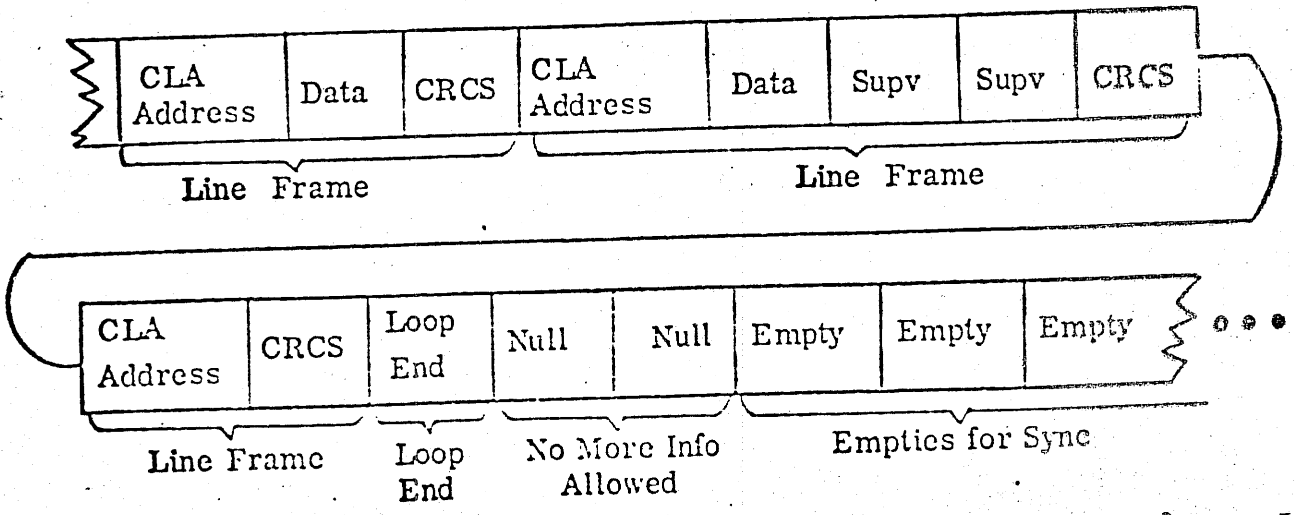
7). The CLA signals the LM that the last loop cell has been transferred and the LM places the accumulated CRCS on the loop.

8). The next cell on the loop is then interrogated to determine if an "empty" exists permitting more information to be transmitted within this loop batch.

9). If the next cell is an "empty", the next CLA with a "demand" is selected and the operation continues until all CLA's have been serviced or a non-empty cell is detected on the loop. The LM then replaces the "loop end" cell and drops loop synchronization.

10). The "loop end" then propagates to the next LM, if any exists, which performs in a manner identical to the above sequences.

An input loop batch containing information is shown below:



11). The MLIA receives the loop batch containing data and/or supervision from the CLA's and stores the information in memory for distribution and processing by the controlling processor.

12). The MLIA checks the CRCS on each line frame received and initiates "restart" action if a line frame is in error.

13). The MLIA times out each "loop end" cell transmitted and initiates restart action if a "loop end" cell is transmitted and is not received in the specified period of time.

14). The MLIA will not generate another "loop end" until the previous "loop end" is received or timed out, thus only one loop batch may be present on the Input Loop at one time.

15). The controlling processor examines each loop cell stored in memory by the MLIA. A data character is examined to determine if any special action or translation is required for that character.

16). The controlling processor then stores the character in the appropriate input data buffer and identifies any character that requires special action.

2.1.2 Sequence of Operation for Transfer from the Controlling Processor to the CLA (Output)

1). Output Loop action is initiated by receipt of an Output Data Demand (ODD) from a CLA or by a command issued by a protocol handler. Data characters will normally be output only after receipt of an ODD from the CLA but "supervision" may be output to the CLA at any time.

2). The controlling processor receives ODD's from the MLIA and constructs line frames in the standard formats (illustrated in Figure 3) according to the information that is available for output to the CLA. A higher performance version of the MLIA may also construct line frames after receipt of an ODD without processer intervention.

3). The MLIA computes the CRCS on each line frame and transmits the line frame on the Output Loop in a serial manner at a fixed rate of 20 megabits per second.

4). The Loop Multiplexer maintains synchronization on the Output Loop at all times. When the LM detects a "CLA address" on the loop, it will present that CLA address to all CLA's connected to the LM on a common buss. Any CLA recognizing its address signals the LM and subsequent loop cells in that line frame (excluding the CRCS) are transferred to that CLA one loop cell per transfer.

5). The LM computes the CRCS on each line frame received and notifies the CLA if an error is detected.

6). The MLIA monitors the input side of the Output Loop to maintain synchronization and to timeout the "loop end" cell transmitted with each loop batch. The "loop end" timeout function is done for diagnostics reasons only and is not an integral part of the loop error protection (refer to Section 2.2 for loop error procedures).

2.2 Multiplex Loop Error Control

The requirement for Multiplex Loop error control is to provide a method of detecting errors that might occur on a loop (which could be up to 100 feet in length) that would prevent introducing an undetected error rate that is higher than that expected on communications circuits using conventional error detection methods between the CLA and the terminal.

The Multiplex Loop Interface Adapter, with the cooperation from the Loop Multiplexer, is responsible for the integrity of the information on the loops. The "Cyclic Redundancy Checksum" loop cell and the "loop end" cell are employed to detect errors in loop batches and to recover from such errors. The procedures used for the Input and Output Loops, respectively, are described below.

2.2.1 Input Loop Error Procedure

The MLIA will start a timer when each "loop end" cell is transmitted. If the timer expires prior to receipt of the valid "loop end" cell on the input side of the Input Loop, or a CRCS error is detected on any of the line frames received, the MLIA will transmit a special "restart loop end" cell which indicates an error on the previous batch. This "loop end" cell will notify all Loop Multiplexers that the last loop batch was not received correctly.

The MLIA will interrupt the controlling processor whenever an input error occurs and the controlling processor will count errors for diagnostic and maintenance purposes. The controlling processor will count successive errors and take action to turn the loop down and/or activate a secondary loop when the primary loop is inoperable.

The Loop Multiplexer will remember which CLA's used the last input batch for "data", "supervision" and/or "Output Data Demand". If the next "loop end" received is a "normal loop end", the previous loop batch is assumed to have been received correctly by the controlling processor and the Loop Multiplexer clears all memory of which CLA's used the previous loop batch. If the next "loop end" received is a "restart loop end", the previous loop batch is assumed to have been received in error by the controlling processor and the Loop Multiplexer sends an "input error" signal to those CLA's (if any) that used the previous batch. The initial product CLA's will recognize the "input error" signal and generate supervisory status to the protocol handler that an "input error" occurred. Future, more complex CLA's could initiate other line restart action when an "input error" signal is received such as "NAK the block", etc.

[illegible]

The Loop Multiplexer will generate a CRCS cell for each line frame placed on the Input Loop and the MLIA will check the CRCS on each line frame received. The MLIA will ignore all information on the loop following a CRCS error until a "loop end" timeout occurs. After timeout, a "restart loop end" is then transmitted to the Loop Multiplexers as described above.

If the MLIA loses sync or detects any other input error condition, it will interrupt the processor with the appropriate reason for interrupt.

The controlling processor will count loss of sync, lost "loop ends" and CRCS errors for diagnostic purposes. The controlling processor will notify the higher level function that the loop is inoperative when error rates reach a given threshold.

2.2.2 Output Loop Error Procedure

The MLIA will transmit line frames and loop batches on the Output Loop. Each loop batch will be followed by a "loop end" cell even though the "loop end" is not required for information transfer or error detection. The MLIA will start a timer when each loop batch is output and if the timer expires before "loop end" is detected on the input side of the Output Loop, an interrupt will be generated to the controlling processor with the appropriate reason for interrupt.

The controlling processor will count missing "loop ends" for diagnostics and maintenance purposes.

The MLIA will generate a CRCS cell on each line frame on output but will not check CRCS on the input side of the Output Loop.

The Loop Multiplexer will check CRCS on each line frame received for a CLA connected to the Loop Multiplexer and will provide an "output error" signal to the CLA if a CRCS error is detected.

The initial product CLA's will send supervisory status to the "protocol handler" indicating an "output error" has occurred. The "protocol handler" will then take the appropriate action to restart the line. Note that the initial product CLA's may have already acted on supervisory commands or data received prior to receipt of the "output error" signal from the LM. Future more complex CLA's could be designed to ingore any information received with an "output error" and take other line restart action such as cause an invalid character or CRC to be transmitted.

The controlling processor will timeout receipt of "Output Data Demand" from each active output line. If an "Output Data Demand" is not received from the CLA in some predefined time, the protocol handler will be notified and initiate the appropriate restart action.

2.3 Mux Loop Backup for a Dual CYBER 70 Communications Subsystem

Backup operation in a dual system environment provides for the servicing of Communication Line Adapters (CLA's) in the event of a communications processor Loop Multiplexer, Multiplex Loop Interface Adapter or loop failure.

In the normal mode of operation, "loop end" cells with flag $F_3 = 0$ will be generated on the Input Loop. In this case, the secondary Loop Multiplexer logic is transparent to the line frames and each Loop Multiplexer services its primary CLA's. See Figure 6.

In the event of a failure of one of the components listed above, the backup controlling processor, via its MLIA, will generate "loop end" cells with flag $F_3 = 1$. The secondary Loop Multiplexer logic will recognize this flag and will service Input Loop requests from the CLA's of the failed subsystem, thus both primary and secondary information will be transferred to a single controlling processor on the same loop. This processor will service both primary and secondary lines until normal operation is restored.

In a dual system environment, CLA addresses must be unique to the total system and not just to each subsystem and the total number of CLA's must not exceed 256.

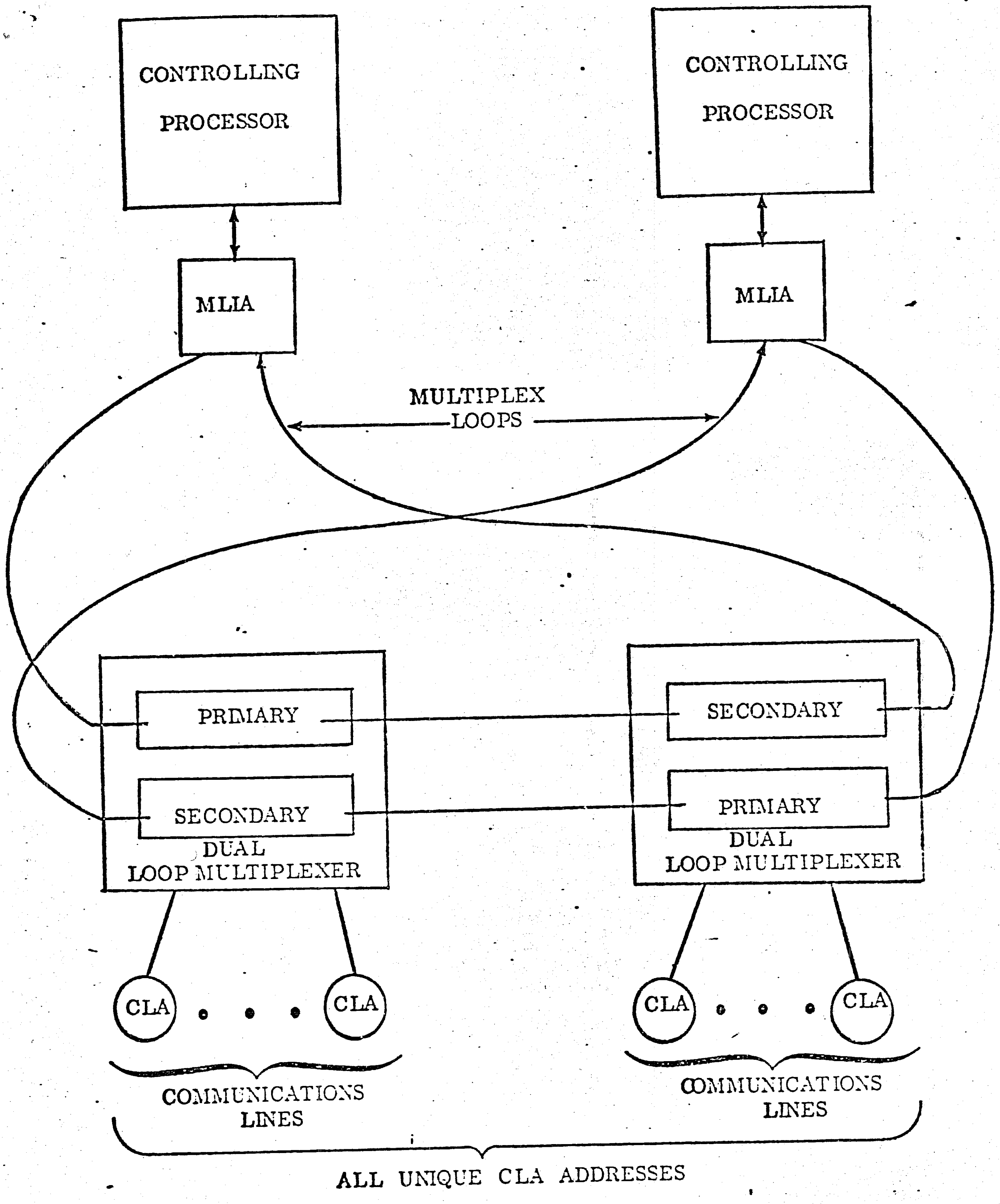
2.4 Multiplex Loop Capacity

The rate of information transfer on the multiplex loops is normally a small percentage of the total loop transfer capacity. Since cost goals dictate the provision for only one character buffering in most CLA's, the Multiplexing Subsystem has one character time to respond to a CLA's demand for the next output character or demand for the next input characer to be picked up.

The Multiplex Loops running at 20 MHz have the capacity to transport up to 555,000 data characters per second. The CYBER 70 Communications System is designed for two different equipment configurations to provide a more precise matching of communications requirements with cost and performance: (a) The low performance system, which could be rated up to 10,000 CPS thruput, would use less than 1% of the loop capacity: (b) The high performance system rated at 30,000 CPS would use less than 6% of the loop capacity.

Operating the loops at a very low percentage of their total capacity minimizes the delays in access to the loop by the CLA's. Since the loop is empty most of the time, the "loop ends" (empty loop batches) will be placed on the input loop more frequently and the CLA may place requests for output data or may place input characters on the loop without a long wait for information from other CLA's to pass. A high speed serial loop rate also minimizes loop propagation time after the information has been placed on the loop.

It is important, then, that the Multiplex Loop capacity be much higher than the rated system thruput to ensure a fast response time to CLA Output Data Demands and system response to input characters. Although the loop rate has a small effect on the line speeds and mix of lines that the system may service, it should not be used as an indication of what the communications system thruput or line mix capabilities might be as these are a function of many other elements of the system, such as firmware processing time, memory speeds, applications dependent software, etc.



DUAL CYBER 70 MULTIPLEXING SUBSYSTEMS

FIGURE 6

3.0 MULTIPLEXING SUBSYSTEM OVERVIEW

An overview of the Multiplexing Subsystem hardware, firmware and software functional elements is shown in Figure 7. The general functions performed by each of the elements are described below.

3.1 Communications Line Adapter (CLA)

The Communications Line Adapter is a hardware element that interfaces one communications line or port to the remainder of the system (usually via a modem). It assembles characters on input (serial to parallel conversion) and places demands for the characters to be transported to the higher level function. It issues demands for output characters to be transferred from the higher level functions and disassembles the characters (parallel to serial conversion) for transmission on the communication lines.

The CLA also stores CLA and modem status and places demands for this status to be transported to higher level functions when a change in status occurs. CLA and modem commands are received from the higher level functions by the CLA to provide control of the CLA and communications line.

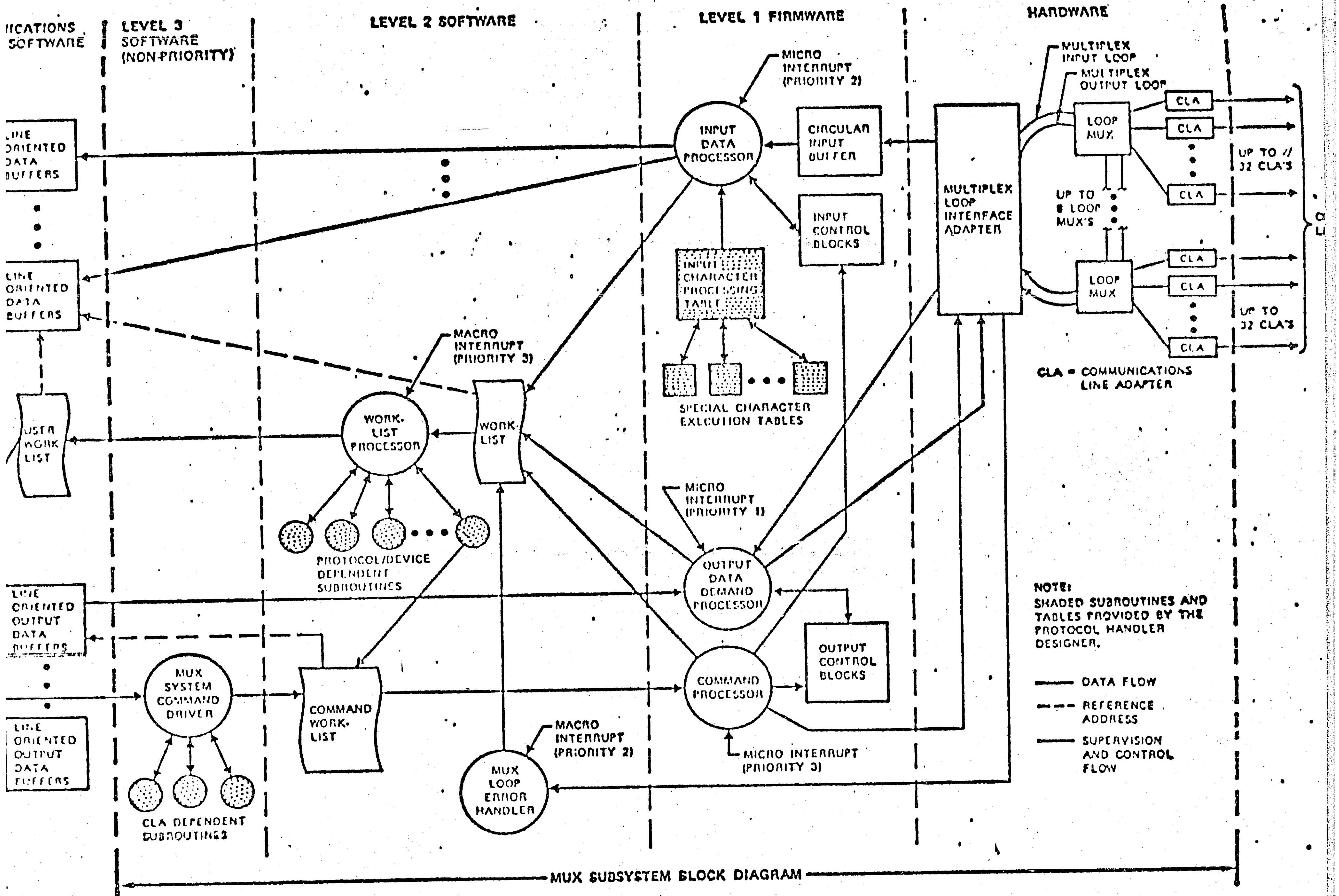
Up to 256 CLA's may be included in one communications system and a different type of CLA may be required for different line types. The characteristics of two of the types being developed for the initial product are given below.

3.1.1 Asynchronous CLA

- Half duplex, full duplex, or echoplex operation
- Code length -5, 6, 7, 8 or bits (also 8 bits + parity)
- All standard speeds to 9600 baud plus others
- Input and output speeds may be different
- Even, odd, or no character parity checking and generation
- Stop bit duration - 1 or 2 units
- Self test mode (loop back)
- All of above selectable by program command
- Break detection and generation
- Data transfer overrun detection

3.1.2 Synchronous CLA

- Half and full duplex operation
- Code length - 6 or 8 bits
- Frame synchronization on 2 sequential characters established by software
- Self test mode (loop back)
- All of above selectable by program command
- Speeds up to 56K bps determined by modem - provisions for external clock source



ICATIONS SOFTWARE
LEVEL 3 SOFTWARE (NON-PRIORITY)
LEVEL 2 SOFTWARE
LEVEL 1 FIRMWARE
HARDWARE
LINE ORIENTED DATA BUFFERS
LINE ORIENTED DATA BUFFERS
USER WORK LIST
LINE ORIENTED OUTPUT DATA BUFFERS
LINE ORIENTED OUTPUT DATA BUFFERS
MUX SYSTEM COMMAND DRIVER
CLA DEPENDENT SUBROUTINES
COMMAND WORK-LIST
WORK-LIST PROCESSOR
MACRO INTERRUPT (PRIORITY 3)
WORK-LIST
PROTOCOL/DEVICE DEPENDENT SUBROUTINES
MUX LOOP ERROR HANDLER
MACRO INTERRUPT (PRIORITY 2)
MICRO INTERRUPT (PRIORITY 2)
INPUT DATA PROCESSOR
CIRCULAR INPUT BUFFER
INPUT CONTROL BLOCKS
INPUT CHARACTER PROCESSING TABLE
SPECIAL CHARACTER EXECUTION TABLES
MICRO INTERRUPT (PRIORITY 1)
OUTPUT DATA DEMAND PROCESSOR
OUTPUT CONTROL BLOCKS
COMMAND PROCESSOR
MICRO INTERRUPT (PRIORITY 3)
MULTIPLEX LOOP INTERFACE ADAPTER
MULTIPLEX INPUT LOOP
MULTIPLEX OUTPUT LOOP
LOOP MUX
LOOP MUX
UP TO 8 LOOP MUX'S
CLA
UP TO 32 CLA'S
UP TO 32 CLA'S
CLA = COMMUNICATIONS LINE ADAPTER
NOTE: SHADED SUBROUTINES AND TABLES PROVIDED BY THE PROTOCOL HANDLER DESIGNER.
DATA FLOW
REFERENCE ADDRESS
SUPERVISION AND CONTROL FLOW
MUX SUBSYSTEM BLOCK DIAGRAM

FIGURE 7

3.2 Multiplex Loops

The Multiplex Loops consist of two independent coaxial cable pairs that provide a serial connection starting at the Multiplex Loop Interface Adapter (MLIA) and continuing to each Loop Multiplexer on the loop and terminating at the Multiplex Loop Interface Adapter. The loops are driven by the MLIA using DC signalling at a rate of 20 MHz. Each loop consists of a separate clock and data line. The LM(s) monitor information on the loop and regenerate the information as received in all cases except when information is to be added to the Input Loop. In this case, the LM replaces either "empty" or "null" cells received on the Input Loop with information cells.

3.3 Loop Multiplexer (LM)

The Loop Multiplexer provides the hardware to connect Communications Line Adapters (CLA's) to the serial multiplex loops. The CLA's physically reside within a Loop Multiplexer. One (1), 2, 3, or 4 CLA's can be "packaged" per plug-in module; this module is inserted into one of 16 available positions within the Loop Multiplexer. Up to 32 CLA's may be accommodated by a Loop Multiplexer and up to 8 Loop Multiplexers may be connected to the same loop. The LM provides two serial interfaces to the Multiplex Input Loop and Multiplex Output Loop, and a parallel interface to the connected CLA's using a common bus with individual control lines for each CLA.

The LM obtains Input Loop synchronization whenever any CLA connected has placed a demand for input of data, supervision or an output data demand via one of 32 unique request lines (one for each CLA). The LM obtains synchronizatior by looking every bit time for the unique "loop end"/"empty" loop cell combination. After obtaining synchronization, the LM will select one of the requesting CLA's via one of 32 unique select lines. Information is then transferred from the CLA, in loop cell format, one cell at a time via the common bus. As each cell is received from the CLA, the cell is placed on the Input Loop and a Cyclic Redundancy Checksum (CRCS) is accumulated for each cell. Information is placed on the input loop displacing either "null" or "empty" cells until an end signal is received from the CLA at which time the accumulated CRCS is placed on the loop completing the "line frame" for that CLA.

The next CLA with a request for input is then selected and the operation is repeated until no more CLA's have requests for input or a "non empty" cell is detected on the Input Loop signifying no more input is allowed on this "loop batch". The LM then replaces the "loop end" cell on the loop and drops loop synchronization.

The LM records which CLA's place information on the Input Loop for each loop batch. The record is used to notify those CLA's that used the loop batch of a possible loop error that was detected by the MLIA (See Section 2.2.1).

The output section of the Loop Multiplexer receives contiguous loop cells from the preceding LM or MLIA on the Output Loop. All cells received are always passed to the next LM or MLIA on the loop. The LM utilizes "empty" cells to synchronize with the data stream. When synchronized, the LM looks for a CLA address as a frame boundary and passes loop cells within the frame to the CLA's. All CLA's connected to the LM are connected to a common bus and each CLA has a unique address. When the LM has received a "CLA Address" loop cell, it is presented to all the CLA's on the common bus. The CLA with the corresponding address is selected and connects itself to the output bus. Subsequent "loop cells" containing data and/or commands are received from the Output Loop and transferred to the selected CLA until the LM detects the CRCS loop cell signalling the end of the "line frame". The Loop Multiplexer checks the CRCS received against the CRCS accumulated for the line frame, and if a difference is detected, the selected CLA is notified of the error prior to deselection.

3.4 Multiplex Loop Interface Adapter (MLIA)

The Multiplex Loop Interface Adapter (MLIA) provides the interface between the Multiplex Input and Output Loops and the Multiplex Subsystem controlling processor. The MLIA provides the serial to parallel and parallel to serial conversions and procedural control necessary for information flow between the controlling processor and the Loop Multiplexer and its associated CLA's via the multiplex loops.

The MLIA generates and transmits clock, "loop end", "empty" and "null" cells (empty loop batchs) on the output side of the Input Loop to solicit information from CLA's connected to the loop. It then monitors the input side of the Input Loop for 1) clock, 2) loop synchronization, 3) information placed on the loop by LM's that must be stored in memory, and 4) "loop end" cells for error control. The MLIA checks the CRCS and stores each "line frame" received on the Input Loop into the controlling processor's circular input buffer in a direct memory access fashion (without processor intervention) for subsequent processing and distribution of the information by the controlling processor firmware/software programs. After receipt of the "loop end" cell on the input side of the Input Loop, another empty loop batch ("loop end" followed by "empties" and "nulls") will be generated immediately without processor intervention, thus providing the maximum rate of "loop end" generation without allowing more than one "loop end" cell on the loop at any one time.

The MLIA also detects Output Data Demands (ODD's) received from the CLA's on the Input Loop and provides for special priority processing of the ODD's. Processing of the Output Data Demands requires that the next character be obtained from the appropriate line oriented output data buffer and transmitted on the Output Loop. The ODD processing function may be performed by the controlling processor firmware logic or by the MLIA itself, depending on the MLIA hardware configuration. Two version of the MLIA are planned in order to provide a cost/performance match to the number of lines and line speed mix that is terminated by the communications system. The high performance version of the MLIA processes Output Data Demands without processor intervention while the low performance MLIA detects the ODD and flags the controlling processor firmware that priority processing is required. All other functions of the low and high performance MLIA's are the same.

The MLIA transmits line frames on the output loop on command from the controlling processor or as determined by the MLIA logic itself (high performance version only). Output line frames contain CLA address, data characters, CLA commands and CRCS. The CRCS is generated for each "line frame" by the MLIA and is transmitted as the last cell of each line frame.

The MLIA monitors the input side of the Output Loop for loop synchronization and loop batch timeout to be used for error recording and diagnostic purposes.

3.5 Controlling Processor

The controlling processor is that processor which controls and executes the Multiplex Subsystem functions. It will reside within the communications processor (MP17) in a low performance configuration and will reside in a separate peripheral controller configuration of the MP17 in a high performance configuration.

The functions performed by the controlling processor for the Multiplex Subsystem are organized into three levels according to priority, frequency of occurrences, protocol/device dependency (ease of change) and complexity of the function. The basic functions performed by each level are described below.

3.5.1 Level 1 Processing

Level 1 processing provides for those character processing functions that must be performed on each character on input and those functions that must be done within one character time on output. Level 1 processing will be implemented by a combination of hardware, firmware and memory tables.

Input characters stored in a circular input buffer in memory by the MLIA are distributed to line oriented input buffers. Special input character recognition and character sequence recognition defined by protocol/device dependent tables. are performed as the characters are transferred to the line buffers. Also, character counting for "end of logical input" recognition, CRC/LRC and parity checking, and code translation are performed when required.

Supervision received from the CLA's is placed in work lists for processing by higher level Multiplex Subsystem functions.

Since the special Level 1 input functions are protocol and device dependent, these functions will be table driven to provide the addition of a new protocol or dynamic change of the functions without change to the firmware code.

Level 1 output processing is executed when Output Data Demands are received from the CLA's. Output data characters are taken from the line oriented output buffers, transformed into line frame formats, and transferred to the MLIA for transmission on the multiplex output loop. CLA and modem commands received from higher level multiplex functions and protocol handlers will be transformed to line frame format and transferred to the MLIA for output to the CLA's.

Since the Level 1 "output data demand" processing must be completed in one character time for CLA's with only one character of buffering, this time critical code could become the limiting factor on the line connectability of the system. The undesirability of the situation will be overcome by providing optional higher performance MLIA hardware that can execute the ODD processing function without processor intervention. It is important to note that the same situation does not exist with the Level 1 input processing functions since the input buffer provides for more than one character time of buffering.

3.5.2 Level 2 Processing

Level 2 processing provides for input and output work list processing on a priority basis. Functions performed at this level do not take place each character time but must be performed on a priority basis to meet system response time and line efficiency requirements. Level 2 functions will be implemented using 1700 code (running at the high priority 1700 external interrupt level) and main memory tables.

The functions performed will be driven by the work list entries built by the Level 1 processing functions based on special character recognition, end of block recognition, supervision received from the CLA, end of buffer conditions, etc. Provisions will be made for branching to special work list handlers depending on protocol and device type.

Some example functions performed by these handlers are: chaining filled input buffers to input data streams, polling, poll/call response analysis, output of CLA supervision, etc.

Level 2 processing functions will generate work list entries for the protocol handlers for functions requiring their action such as end of transmission, error conditions, etc.

3.5.3 Level 3 Processing

Level 3 processing provides the logic for input and output command processing from the user programs to the Multiplex Subsystem. These functions will be implemented at a low or non-priority 1700 code level. The primary function of thi logic is to transfer logical I/O commands from the user programs to the physical logic to initiate or execute these commands within the Multiplex Subsystem. Example logical commands are: "initiate output", "terminate output", change special character recognition or code set for a particular line, etc. The comman processing will be designed to provide a common user interface which is independent of the multiplex subsystem functional mapping for the low or high performance versions of the system.

4.0 PROPERTIES OF THE MULTIPLEX SUBSYSTEM

The most important design considerations used in the Multiplexing Subsystem definition are listed below. Each of these considerations is reviewed with comments on how the design presented in Sections 2 and 3 fulfills these design goals.

Design Considerations

- Connectability independent of thruput or line placement
- Minimum time dependent code
- Adaptable to both a low performance and a high performance system
- Common protocol/device dependent logic to be shared by all lines, thus allowing low cost line adapters
- Simple method of adding new protocol/device dependent logic

4.1 Connectability Independent of Thruput or Line Placement

Typically, in the multiplexing schemes used today, the system thruput decreases as more lines are connected to the system. This is especially apparent in line scanning multiplexing schemes that share the communications processor for the line scan procedures and/or scan records. The scanning of each line to determine if there is information to transfer causes a fixed overhead rate which is dependent on the number and speed of the lines connected.

The "demand driven" multiplexing scheme does not have any overhead processing to scan each line since the CLA's are designed to signal the controlling processor whenever a character is received from the terminal or the next character is required for output. The "demand driven" system has an overhead rate which is a function only of the traffic load on the communications lines.

Other disadvantages of scanning multiplexing schemes that will be eliminated by the "demand driven" multiplexing scheme are listed below:

The complexity of balancing the system performance for a desired line service rate and system thruput to the number and mix of the lines connected.

A limit on the number of lines that can be connected and successfully scanned exists and this limit could be less than the number needed to generate a traffic load that is consistent with the rated thruput of the system.

The difficulty in designing a scanning mechanism that is fast enough to scan large numbers of high-speed lines and still be cost effective for a low cost, low performance system. This means more than one hardware device is required to cover all ranges of desired performance.

4.1.1 Multiplex Loop Concept

The basic differences in the "demand driven" multiplexer and the "scanning" multiplexer are shown in Figure 8. The Multiplex Loop is a mechanism for resolving contention between the CLA's for access to the Multiplexing Subsystem as shown in Figure 9. Several other methods that were considered to resolve this contention are discussed below.

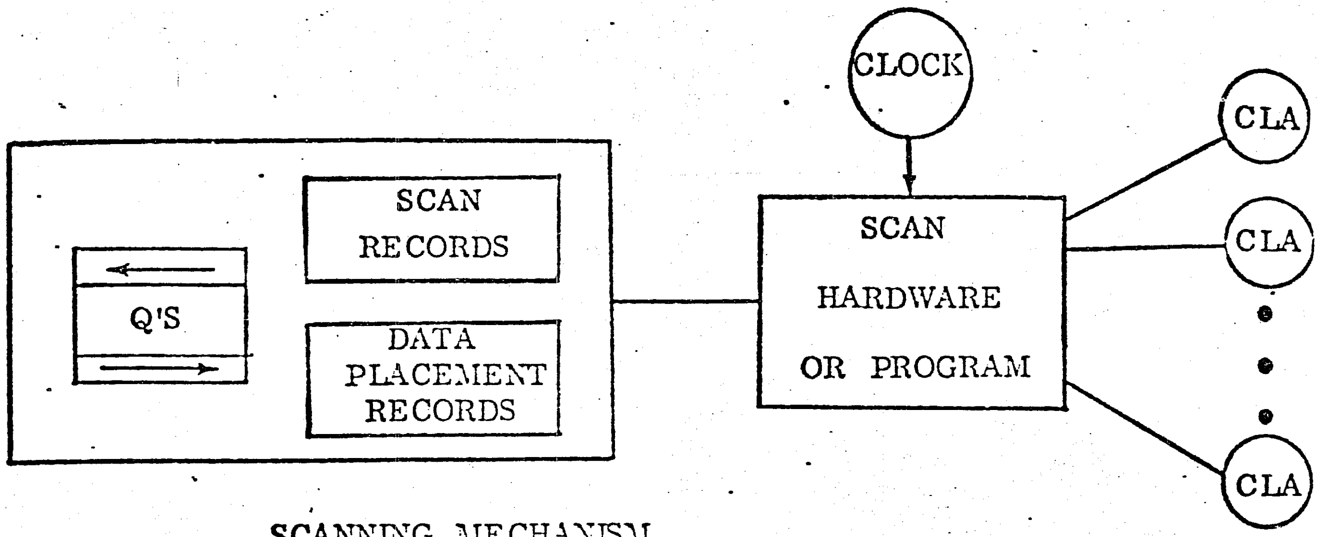
a). Time Division Multiplexing - This method was discarded because it is not consistent with the "demand driven" and connectibility independent of thruput concepts. The time division scheme requires the CLA to wait for its assigned time slot before transferring information. Only a limited number of time slots would be available thus limiting the number of lines that could be connected to the system.

b). Common Buss with Scan Records - This method was discarded because it has an overhead rate to scan each CLA which is a function of the number of lines connected and therefore could not meet the design goals.

c). Common Buss with Request Lines - This method would connect all CLA's to a common parallel buss with a separate request line to the control logic from each CLA. The CLA would raise its request line when it had a "demand" for information transfer to or from the multiplexing system. Common control logic would select one of the CLA's with a "demand" according to predefined priority rules. This scheme could have been adapted and would have met the design objectives. The Multiplex Loop was choosen over the Common Buss with Request Lines technique because the common parallel buss with one request line for each CLA would have physical configuration constraints that would not be imposed by the serial Multiplex Loop. The serial loop has the potential to economically connect many local or remote devices independent of their physical proximity to the communications processor.

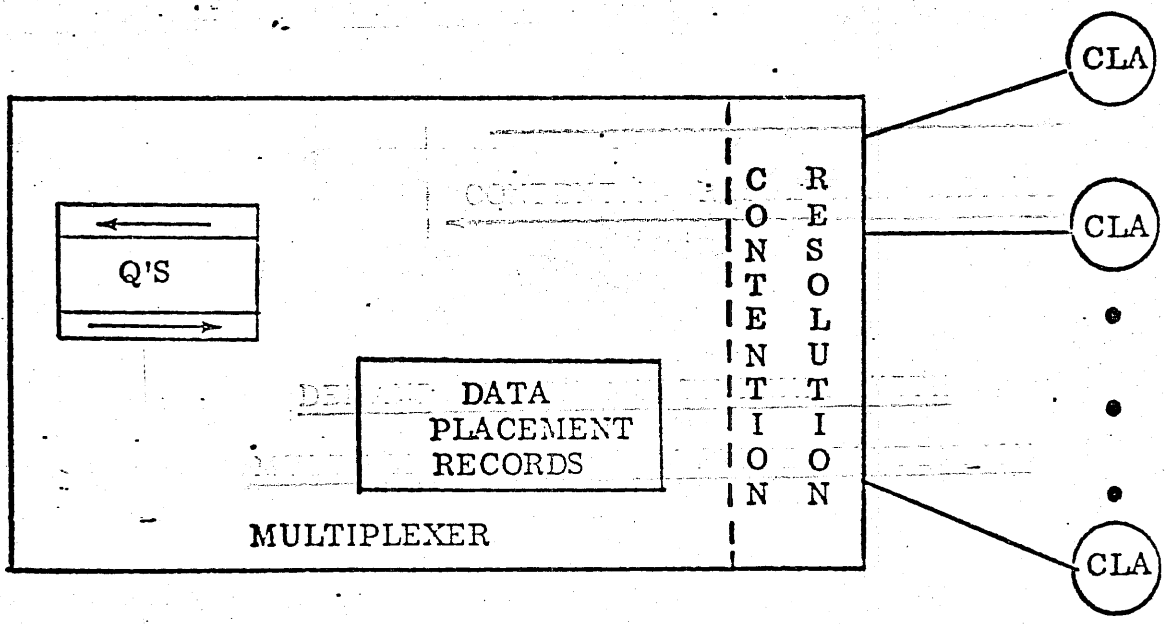
The Multiplex Loop, as defined, allows the CLA access to the higher level Multiplexing functions to present demands for input or output without waiting for its time slot or scan cycle. With the Multiplex-Loop, the CLA does have to wait for a "loop end" signal but this delay is not a function of the number of lines connected; as pointed out earlier, it is dependent only on the traffic rates of the system.

The initial product Multiplex Loop mechanism, using coaxial cable, will allow for a loop up to 100 feet in length. This will allow the Loop Multiplexer and CLA equipments to be located close to the communication line terminations for some installation. The Multiplex Loop method could also be economically adapted to a much longer modem driven loop where the coaxial cable could be run to many potential terminal locations in a large factory or campus environment.



SCANNING MECHANISM

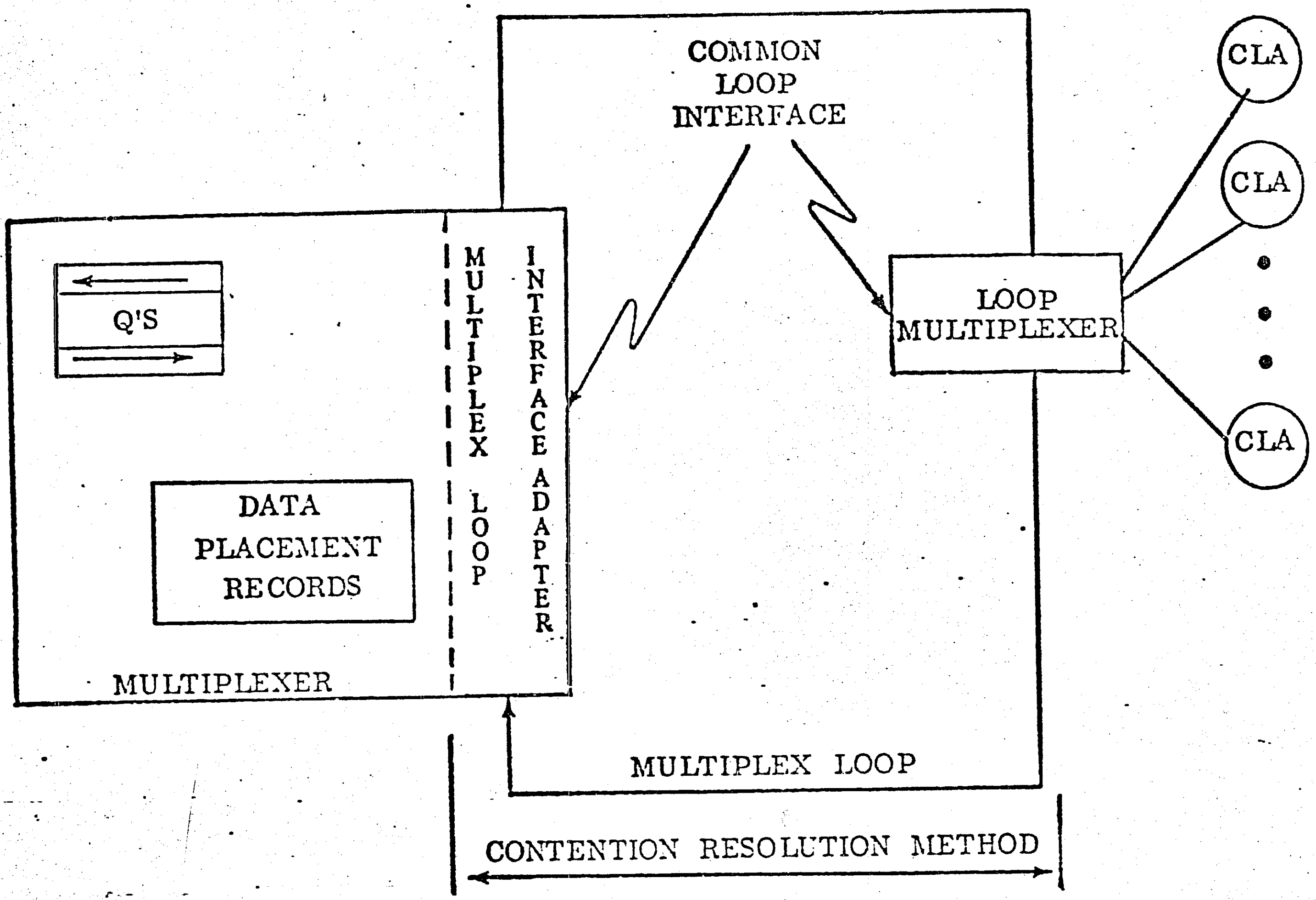
- SCAN LOGIC SAMPLES EACH CLA FOR INPUT OR OUTPUT DATA/SUPERVISION



DEMAND DRIVEN MULTIPLEXER

- CLA'S GENERATE DEMANDS FOR OUTPUT DATA/SUPERVISION
- CLA'S GENERATE DEMANDS FOR INPUT DATA/SUPERVISION TRANSFER

FIGURE 8



DEMAND DRIVEN MULTIPLEXER WITH

MULTIPLEX LOOP TO RESOLVE CONTENTION

- CLA'S GENERATE DEMANDS FOR OUTPUT DATA/SUPV
- CLA'S GENERATE DEMANDS FOR INPUT DATA/SUPV TRANSFER
- THE MULTIPLEX LOOP RESOLVES CONTENTION AND TRANSPORTS DEMANDS, DATA/SUPV BETWEEN THE MULTIPLEXER AND THE CLA'S

FIGURE 9

4.2 Minimum Time Dependent Code

The Multiplex Subsystem has been designed to eliminate,as much as possible, time dependent code. Assuming the functional mapping shown in Figure 7, the only time critical function is the Level 1 output function which must output a character within one character time after receipt of an Output Data Demand (ODD). This function will be executed at the firmware level and be given the highest priority in the processing schedule. No main memory level code will exist which has to be executed with one character time.

Preliminary analysis shows the Level 1 output functions should present a zero probability of data underrun at the CLA for the Low Performance Version (LPV) using the maximum line configuration and thruput goals. These Level 1 functions may, however, present some unacceptable probabilities of data underrun in the High Performance Version (HPV) at its maximum rating if these function were not moved from the firmware to the hardware. The analysis results, then, have instigated a development plan that includes two MLIA hardware options, one that performs the Level 1 output functions and one that does not.

4.3 Adaptable to Both Low Performance Low Cost and High Performance System

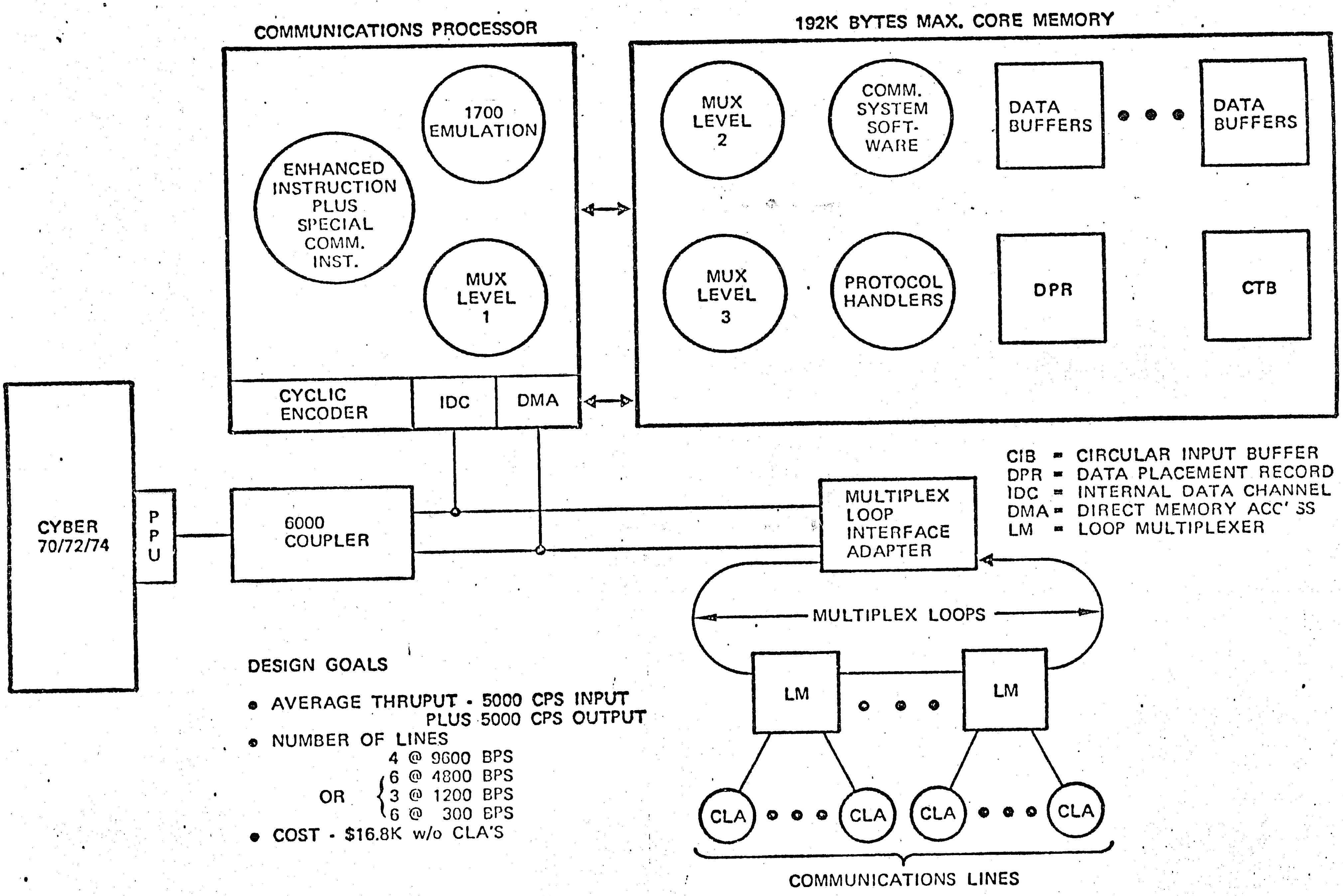
The mapping of Multiplex Subsystem functions must be such that these functions may time share the communications processor in the Low Performance Version and also be moved to a separate multiplexing processor for the High Performance Version (HPV).

The functional mapping and performance/cost design goals for the Low Performance Version (LPV) and High Performance Versions (HPV) communicatior systems are shown in figures 10 and 11.

Note that the primary differences in the LPV and HPV functional mapping is the availability of 100% of the Multiplex Loop Controller (MLC) processor time to perform Multiplex Subsystem function in the HPV, while these functions should average no more than 20 to 30% of the processor time in the LPV although higher short peaking conditions could be accommodate without adverse effect.

4.4 Common Protocol/Device Dependent Logic to be Shared by All Line Adapters (CLA's) Thus Allowing Low Cost Line Adapters

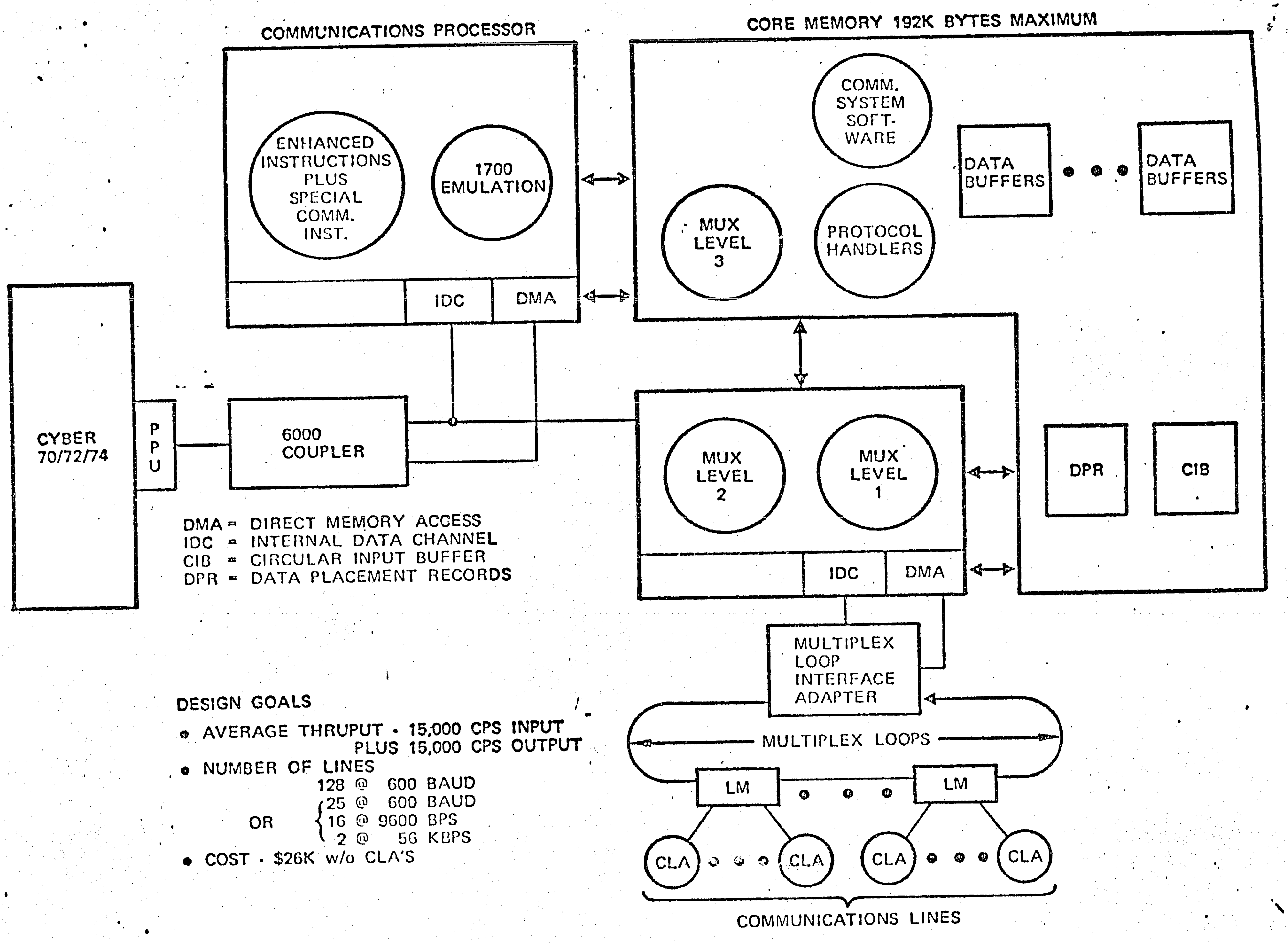
The Level 1 processing outlined in Section 3 includes many of the protocol dependent functions currently done in the line adapters in the CYBER 1000. This approach uses common logic to perform the functions that were previously duplicated in each line adapter which should prove to be more economical. (A more detailed design and cost estimates of these functions must be completed before a cost comparison could be made.) This approach provides the ability to add new protocols without a development effort for new line adapters.



DESIGN GOALS

- AVERAGE THRUPUT - 5000 CPS INPUT PLUS 5000 CPS OUTPUT
- NUMBER OF LINES
  4 @ 9600 BPS
  OR { 6 @ 4800 BPS, 3 @ 1200 BPS, 6 @ 300 BPS }
- COST - $16.8K w/o CLA'S

FIGURE 10. LOW PERFORMANCE SYSTEM



DMA = DIRECT MEMORY ACCESS
IDC = INTERNAL DATA CHANNEL
CIB = CIRCULAR INPUT BUFFER
DPR = DATA PLACEMENT RECORDS

DESIGN GOALS

- AVERAGE THRUPUT - 15,000 CPS INPUT PLUS 15,000 CPS OUTPUT
- NUMBER OF LINES
  - 128 @ 600 BAUD
  - OR
    - 25 @ 600 BAUD
    - 16 @ 9600 BPS
    - 2 @ 56 KBPS
- COST - $26K w/o CLA'S

FIGURE 11. HIGH PERFORMANCE SYSTEM

4.5 Simple Method of Adding New Protocol/Device Dependent Logic

The entire Multiplex Subsystem design employs extensively subroutine and table driven logic techniques to provide for minimum complexity and interference with existing programs when a new protocol is added. It is envisioned that a customer could add new devices or protocols to the communications system using the Compiler (PASCAL) language for higher level function generation and using tables entries to add the Multiplex Subsystem functions required to handle the special protocol/device characteristics. It will be a design goal to eliminate the necessity for modifications to the firmware when a new device or protocol is added to the system.