

406

INTERFACING INCREMENTAL PLOTTER WITH EC 1020 B COMPUTER SYSTEM

Dissertation submitted in partial fulfilment
of the requirements for the degree of
MASTER OF PHILOSOPHY

SATYA NARAYAN SINGH

124p. + fig.

School of Computer and Systems Sciences
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067

1981

CERTIFICATE

This is to certify that the dissertation
" INTERFACING INCREMENTAL PLOTTER WITH EC 10203
COMPUTER SYSTEM " submitted for the partial
fulfilment of the requirements for the degree of
Master of Philosophy, by Satyanarayan Singh, is a
record of bonafide work. The research work embodied
in this dissertation has been carried on in School
of Computer and Systems Sciences, Jawaharlal Nehru
University, New Delhi-110067. The work is original
and has never been submitted in part or in full for
any degree or diploma of any University.

B.S. Under
Supervisor

S Singh
Candidate

N.P. Munkh
Co-Supervisor

N.P. Munkh
Dean

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI

1981

ACKNOWLEDGEMENT

It is my pleasure to acknowledge the wide variety of assistance the staff members, colleagues, friends, and others have provided me in finishing the work of this dissertation.

I am particularly indebted to my supervisors, Prof. (U.P. Mukherjee and Dr. D.S. Khurana) for the valuable help they have given me althrough the development of this work. Words can hardly describe the gratitude I owe to Dr.P.Biswas and Dr. A.R.Majumdar for their constant guidance and encouragements given to me throughout the research work. Special thanks are also due to Mr. D.K. Roy for being always helpful in the laboratory work.

Mr. Ashok Kumar, Mr. J.P. Singh, Mr. S.K.Sinha and other friends deserve my sincere thanks for sparing their valuable time for sketching the diagrams given in this dissertation and the proof reading work. I am also grateful to my parents and family members for their love and affection, which helped me a lot in my studies.

Last but not the least, I wish to sincerely thank my wife for her unflailing support, patience and co-operation without which it would have been extremely difficult for me to complete this work.

I also wish to acknowledge gratefully the financial assistance U.G.C. had given me during my tenure as student in J.N.U.

New Delhi
5.1.82


SATYANARAYAN SINGH

TABLE OF CONTENTS

	PAGE
ACKNOWLEDGEMENTS	
SYNOPSIS	
CHAPTER - 1 INTRODUCTION	1
CHAPTER - 2 GENERAL INTERFACING CONSIDERATION	3
2-1 Preliminaries	3
2-2 Interface Functions	4
2-3 Concept of the Standard Interface Module	6
2-4 Factor Affecting On Interface Design	7
2-4.1 Data Transfer Techniques	7
2-4.2 I/O Structure of the Computer	9
2-4.3 I/O Device Characteristics	10
2-5 Specification of a Computer Interface	11
2-6 Incremental Plotter Interfacing with a Computer	12
CHAPTER - 3 I/O SYSTEM OF EC 1020/30B COMPUTER - BUK CHANNEL	
3-1 Organisation of EC-1020/B Computer	13
3-2 Input/Output System	17
3-2.1 I/O Interface	18
3-2.2 Channels	18
3-2.3 I/O Instructions	21
3-2.4 I/O Operation Initiation	23
3-2.5 I/O Command	25

3-2.6	I/O Terminator	29
3-2.7	I/O Interruption	30
3-2.8	Basic Procedure For a Data Transfer Operation	31
3-3	Multiplexor Channel	32
3-3.1	Basic Data Path And Block Diagrams	32
3-3.2	Microprogram Control of the I/O System	33
3-3.3	Multiplexor Channel Block Diagram And Data Flow	33
3-3.4	Hardware of the MUX Channels	38
3-4	General I/O Interface Sequence Multiplexor Channel.	39

CHAPTER -4

	System Component Description	45
4-1	System Block Diagram	45
4-2	I/O Interface Structure and Signal Sequences.	47
4-2.1	Line Definition	50
4-2.2	System EC-1020B Interlocking Rules	56
4-2.3	Priority Assignment by Sel-Out Routing	57
4-2.4	Important Signal Sequences	60
4-3	GT-150 Incremental Plotter	69
4-3.1	Function	69
4-3.2	Operating Principle	70
4-3.3	General Specifications	72

	PAGE
4-3.4 Input Requirement	73
4-3.5 Operational Check-out and Operating Procedure	74
4-4 Buffered Plotter Controller	76
4-5 Interfacing Requirements	78
4-5.1 Interfacing Requirements- Incremental Plotter	78
4-5.2 Interfacing Requirements- EC-1020B Mux Channel	79

CHAPTER -5

Interface Design Description	81
5-1 Block Diagram of Buffered Controller	81
5-2 Division of the Interface logic	84
5-2.1 Interface States	84
5-2.2 Buffer	86
5-2.3 Command Structure	86
5-2.4 Sense Byte & Status Bytes	86
5-2.5 IN-TAG signals	87
5-2.6 Device Address and Priority	87
5-2.7 Main Block in Interface Control	88
5-3 Logic Diagram Notation	94
5-4 Functional Description of PCB Cards	96
5-4.1 Bus-Out Buffer Register	97
5-4.2 Address Match	97
5-4.3 Parity Check	98
5-4.4 Accepting Command Code	98

	PAGE
5-4.5 Address, Command and Data States	99
5-4.6 Operational-In And Status-In Signal	99
5-4.7 Service-In, Address-In, Duffer-Write-Pulse and Bus-In-Priority Bits	101
5-4.8 Directing the register to Bus-In	104
5-4.9 Buffer Write Control ckt.	104
5-4.10 Buffer Read/Write Control Circuit	105
5-4.11 255 Bytes Buffer	107
5-4.12 Buffer Output Control ckt.	107
5-4.13 Plot Signal Generation	108

CHAPTER -6

Conclusion	
6-1 General Consideration	109
6-2 Future Extensions and Improvement	113
6-2.1 Resolution of Plotting	113
6-2.2 Hardware Technology	114
6-2.3 Plotter ckt. Improvement	114
6-2.4 Card-Layout	115
6-2.5 Buffer Size	116
6-2.6 Interrupt Handling Facility	116
6-3 Software Consideration	116
6-3.1 Graphics Software - General Strategy	116
6-3.2 IOCS Software Specification	120

REFERENCES

- APPENDIX - A Logic Diagram Implementation
- APPENDIX - B PCB Card-Layout
- APPENDIX - C IC Chip Pin Assignment

LIST OF FIGURES

<u>Figure</u>	<u>Title</u>
CHAPTER-3	
3.1	General Organisation of a Digital Computer
3.2	I/O Interface
3.3	General Block Diagram of the CPU & I/O System
3.4	Block Diagram of MUX Channel
3.5	General I/O Sequence.
CHAPTER-4	
4.1	Interconnection; Channel To Plotter
4.2	Line Definition
4.3	Priority Assignment By Select-Out Routing
4.4	Initial Selection
4.5	Write (O/P) Data Transfer
4.6	Ending Sequence-Channel Terminated
4.7	Plotter Assembly.
CHAPTER-5	
5.1	Selection Block
5.2	Schmitt Trigger for Closing Switches
5.3	Schmitt Trigger for Opening the Switch
5.4	RESET CKT.
APPENDIX A	Logic Diagrams of Implemented Interface ckts.
APPENDIX B	PCB Card-Layouts
APPENDIX C	Integrated ckt. Chips.

SYNOPSIS

As the digital computers are becoming increasingly popular, the need for the better presentation of its output has been constantly felt amongst the users. Since the conventional output device of the computer, i.e. Printer, can only present the data processing results in tabular form, the interpretation and analysis of result data remains a tedious work. The computer graphics is the solution of this problem. For the rapid translation of digital outputs into computer graphics for quick visual interpretation and analysis, the digital plotters are the best suited devices. The plotter can work on-line with the computer for computer graphics, if it is suitably interfaced with the host computers.

The incremental plotters are regarded as one of the slowest peripherals in a computer system. The objective of this project is to design the interface for an indigenous incremental plotter (KELTRON) to be connected to the multiplexor channel of the EC 1020 system. The major objective of the design is to have a maximum possible throughput efficiency in the multi-programming environment. This complete buffered interface supports multiplexor channel operation in the burst mode and allows the particular memory partition (stored output of the plot-program) to be released within a comparatively insignificant duration. As the EC 1020 interface follows the standard IBM 360/370 protocol,

so, with the minor modifications, the KILTRON GT-150 plotter can be connected on-line with any of the EC, IBM, or UNIVAC computer systems following similar protocol. In this thesis, most of the hardware circuitry has been designed and implemented in the laboratory and the detailed requirements of the IOCS software has been suggested.

I N T R O D U C T I O N

Increasing demand on digital computer for use in large variety of fields has necessitated the development of specialized interfaces so that they could be connected to different kind of external devices for processing jobs of different kind. These external devices can be standard computer units such as disks, tape-driven, CRT displays, printers, and plotters; they can be various electrical, medical, or other instruments that employ the computer for ON-LINE data processing; or they can be industrial equipments that employ the computer to supervise operations such as an automatic oil refinery or a computerized steel-rolling mill.

One can have the required interface for a particular application in any of the following three ways:

- i) It could be purchased entirely from the manufacturer;
- ii) It could be assembled using commercially available ckt. cards;
- iii) It could be designed from scratch and could then be built with individual integrated ckt. chips.

The third option has been followed in this work and the primary aim of this work has been to build up IC 1020/30B capability to support different applications in the field of Graphics. This has been done by interfacing

HFLTRON's GT 150- a drum type incremental plotter having six data input lines with EC 1020/30B system.

The project forms part of a laboratory development effort whose ultimate aim is to provide a graphics facility to the EC 1020/30B users in the areas of Research and development.

CHAPTER - 2GENERAL INTERFACING CONSIDERATIONS.**2-1 Preliminaries**

Input/output devices provide a means of communication between the computer and the outside world. A computer can have more than one I/O devices connected to its I/O unit. Communication with any of I/O devices is established by selecting that device. I/O system of a computer should meet the following requirements;

1. Computer should not be tied down to the peripheral devices;
2. Computer should be independent of nature of I/O device i.e I/O system should be able to communicate with any of devices through proper interfaces;
3. It should account for difference in the type of information presented to or by I/O device, e.g the system might have to perform word assembly before the data from a character oriented Input device could be transferred to the computer or word-disassembly for data word received from computer before sending to an output device;
4. The system should provide the way to write programs in such a way as to recover from unexpected conditions.

5. The system should have provisions to allow I/O devices to transfer data at their convenience regardless of memory availability at a particular moment e.g. buffer storage.
6. The addition or removal of any number of devices should not affect the computer from functioning normally.

The Interface should be built to take care of most of the above goals in hardware or leave some of them to be achieved by software so that the ratio of cost of hardware required to the improvement achieved is as small as possible.

2-2 Interface Functions

The following functions are performed by a typical interface between an I/O device and computer;

1. To accept data from the computer or I/O device.
2. To transmit data to the computer or I/O device.
3. To execute the commands of the computer.
4. To transmit current status of operation.
5. To encode or decode data in terms of the required recording or reproducing methods.
6. To monitor overall device operation.
7. To account for difference in data transfer rates of the two separate systems by providing buffer storage.

During this temporary storage period, code translation, word-assembly or disassembly, block formation, editing or any other data processing that may be required prior to information transfer is performed.

The interface may provide some local autonomous control functions in addition to the commands received from the computer. Other facilities desired are to try to correct the corrigible errors before informing the computer, provisions for recovery from the unexpected conditions, minimize the time the computer remains tied up during the data transfer operation and to handle, may be more than one I/O devices.

Now a days manufacturers not only supply a peripheral device but also a generalized Device control system with it, which performs almost all functions stated above except say 1, 2, & 4, thus simplifying interfacing.

The functions to be performed by a typical interface in such a situations would be:

1. To receive data from the computer and perform data word-disassembly, if required and transmit to the device control system.
2. To receive data from the device control system and perform data word-assembly and transmit to the computer.
3. To receive a command word and decode the operation to be performed OR to receive each command on a separate line and communicate to the device control system the operation required.

It may check out the validity of the command received at that particular moment.

4. To receive the device control system status and transmit to the computer on its request.
5. To provide means of communicating error conditions which require quick action of the computer.
6. Speed difference between the CPU and the peripheral is taken care of either by the device control system, or could be provided by the interface.

2-3 Concept of the standard Interface Module.

New concepts are being evolved now a days moving towards standardization of computer interface. Historically computer interface have been built with very little standardization. Standardization has following obvious advantages:

1. Saving of money and manpower because of non-duplicacy of efforts in interface design.
2. Switching of peripherals from one computer system to another is simple, e.g. in case a computer system is to be replaced, it is just necessary to plug-in the standard interfaces.
3. Easy maintenance of peripherals e.g. a peripheral to be repaired could be plugged out and repaired off-line or it could be connected to a small computer (mini or micro) for on-line testing.

standardisation of computer interfaces is achieved by:

1. Fixing the physical dimensions of interface modules.
2. Standardizing paths or buses called data highways or data ways. The connector on back of the modules is plugged into this highway and each wire in the highway is reserved for a specific signal type. The signal characteristics and functions are also specified.

The above concept of computer interfacing may require a very simple interface as a plug on computer cabinet which supplies the appropriate signals to appropriate points of data way or a control module to generate appropriate signals.

2-4 Factors affecting on Interface Design.

Design of an interface between an I/O device or a device control system and a computer is affected by a number of factors. The important factors are :

1. Modes of data transfer available on the computer.
 2. I/O structure of the computer.
 3. I/O device or device control system available.
- The maximum data input or output rates with or without critical timings for data transfer etc.

2-4.1 Data Transfer Techniques.

There are several methods for transferring data between a computer and the outside world. These are :

- a) Programmed data transfer.

- b) Programmed data transfer with interrupt.
- c) Direct Memory Access transfer.

Interface design for an I/O device is influenced by the data transfer technique to be implemented, the choice of which depends upon the I/O device characteristics (discussed later) to be interfaced with the computer, maximum data transfer rate with which I/O device can communicate, and critical data transfer timings. It may be required to have in an interface for an I/O device, all of data transfer modes. It may be the case that an I/O device provides high data transfer rates but data transfer timings are not critical. Clearly in such a situations, the choice of data transfer mode is also dictated by cost comparisons of different modes of implementation.

In the programmed transfer method, the peripheral is serviced at the convenience of the computer, i.e., when over the I/O instructions happen to be reached in the normal program sequence. By contrast, the interrupt method places the data transfer at the convenience of the external device, i.e., service is initiated through what amounts to an I/O subroutine as soon as the peripheral requests it instead of when the computer gets around to it.

Direct memory access, DMA, is a very quick and efficient method for transferring large blocks of information such as when data is passed to or from a disk. It is similar to the interrupt method in that service is initiated immediately, but instead of having a moderate-sized software subroutine supervise the transfer, it is done entirely by

special hardware. Thus, the data are transferred much quicker at the expense of more complicated and costly hardware.

2-4.2 I/O Structure of the Computer.

For connecting I/O devices to the computer, there are two systems:

1. BUS SYSTEM: In this system the I/O Bus will have a number of data lines, address lines, and control lines which are shared by the devices. Depending on the address a particular device will be selected and the control signals will activate the device. Adding a new device, it is just sufficient to have a proper device controller which can readily be interfaced with the I/O Bus. The CPU is not required to be modified. The computer has at least one I/O instruction operation code. Typically the op. code, the device number and a few control bits form an I/O instruction.

Whenever CPU interprets an I/O instruction it sends control signals on different control lines of the I/O bus, depending on the control bits of the I/O instruction. If the computer word length is insufficient to accommodate the op-code, the device number and a few control bits, it usually uses one of the working registers, generally accumulators, to hold the device number or the control bits. Any I/O operation to be performed on a given device will require at least two I/O instruction to be executed in this typical order.

1. First I/O instruction to select the device.
2. Second I/O instruction to perform the required operation.

Different combinations of these control bits give rise to different combination of control signals on I/O bus, each of which could be used to perform a specific function.

The interface must have a device selection logic, an instruction decoding logic, and the logic to perform the required function. Complexity of the interface depends on the size of the instruction set chosen to handle the new device and functions to be performed by each instruction in this instruction set.

2. Radial System: In this system, the control signals are different for different devices. So whenever a new device is to be added, the computer CPU is required to be modified.

2-4.3 I/O Device Characteristics:

Interface design depends on the characteristics of the I/O device to be interfaced or the characteristics of the device controller logic available with it.

The I/O devices are Binary coded devices or alphanumeric devices. The various I/O devices differ widely in nature. The main characteristics influencing the interface design are:

1. Nature of signal available, e.g. analog or digital.
2. Data encoding, decoding and formatting required.
3. Speed of the I/O device.
4. Critical or non-critical data transfer timings etc.

If the I/O device has been provided with a device controller logic, then the interface design, depends on the characteristics of the device controller logic, and is usually simplified.

2-5 Specification of a computer interface.

The interface designer should include the following points for specifying the interface developed:

1. Detailed information about the device being linked to the computer, alongwith the interfacing requirements on both the sides.
2. Command implementation: Instruction set associated with the interface. Specific instructions, very clear explanation of each instruction.
3. Transfer rates: The user on specifying transfer rates of the interface must also give careful attention to the amount of program manipulation required before the data arrives at the interface and is stored in the computer memory or on the peripheral device, and the effect of presence of the other devices.
4. Maintenance and packaging: Maintenance is usually thought of as two activities:
 - a) Preventive maintenance - Anything from periodic replacement of mechanical components to exercise

equipment on regular bases.

- b) Repair maintenance: Specifications should give details of checkout programs that are envisioned. Specifications should call out test point that can be accessed from the front panel or back panel or any display devices e.g. light emitting diodes. Specifications should ask for repair parts list. From repair point of view, interface should be plug-in type. Finally, specifications call for packaging details, dimensions, connectors, front panel arrangements, logic diagrams etc.

2-6 Incremental Plotter interfacing with a computer.

The incremental plotter is an electromagnetic device which accepts digital data on its input lines and moves the pen past a paper (drum type or flat bed) to produce graphical representation of the data inputs. There are two types of plotter- Drum type and Flat bed type. In drum type plotter paper moves one step at each digital signal bit received on Y-line. It can move in forward direction if signal is comes on + Y line, and it moves backward if the - Y line receives a signal. Pen moves along the length of the drum i.e. at an angle of 90° to the paper movement. Pen can move either to the left or right depending on the signal received either on - X line or + X line respectively. There are two more lines - PEN UP and PEN DOWN which keeps pen up and down respectively.

CHAPTER - 3INPUT/OUTPUT SYSTEM OF EC 1020/30B COMPUTER-MUX CHANNEL3-1 Organisation of EC 1020 Computer

The general organisation of a typical computer system like EC 1020 can be represented as shown below. The heart of the system is the Control Processing Unit (CPU), shown as comprising of a main storage, which holds both program and data, an arithmetic - logic unit (ALU), which contains processing circuitry such as an adder, shifter, and a few fast registers for holding the operands, and the instruction currently being processed. One part of the CPU is a set of routing circuits which provide paths between storage and the ALU and Input/Output controllers or channels. In this type of system, many storage or Input/Output devices may be wired to one channel, but only one device per channel can be transmitting information from or to main storage at any one time. This is, of course a restriction on the number of devices that can operate concurrently. This is imposed because of the economy of sharing common paths to main storage and simplicity in controlling movement of information between the devices and storage.

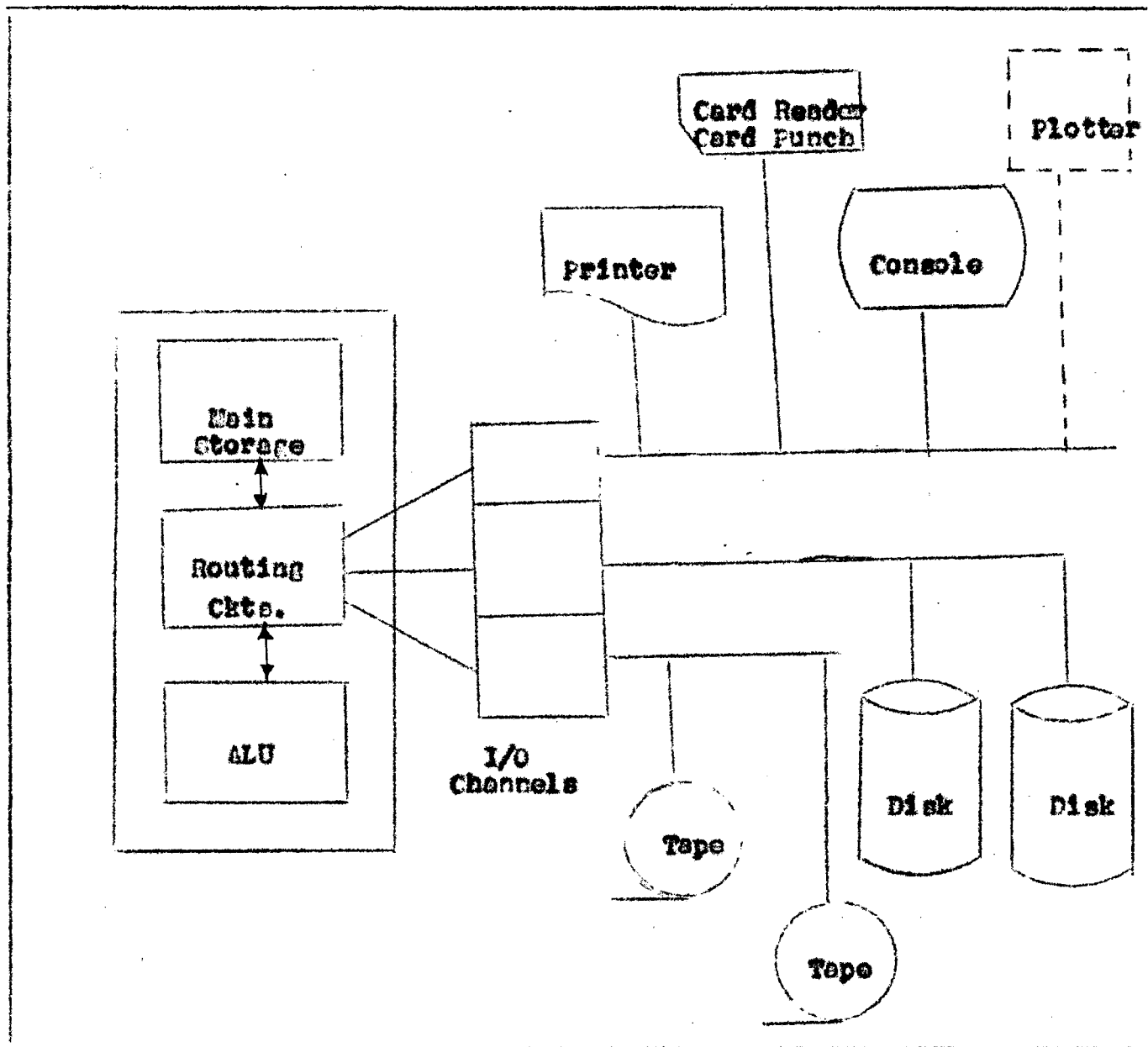


Fig.3.1 General Organisation of a typical digital computer.

The major parts of a computer may be described as follows:

1. Storage - This is a means for storing a rather large volume of information and a simple economical access mechanism for routing an element of information to/from storage from/to a single point(register). Storage is usually available in several versions, even in the same system (main storage and secondary storage), these vary in access time, capacity, and cost.
2. Data Flow - The switching networks that provide paths for routing information from one part of the computer to another.
3. Transformation - The circuits for arithmetic and other data manipulation. This function is usually concentrated in a single Arithmetic - Logic Unit(ALU). The centralization provides economy since a single set of fast expensive circuits is used in time sequence for all operations. Transformation circuits operate on information obtained from storage by control of the data-flow switching.
4. Control - This is a general term that includes the important function of performing time sequences of routings of information through the data flow. The control function appears on many levels in a computer. Usually the control is organised

as a set of time sequences, or cycles. Each cycle period is commonly (but not always) divided into equally spaced time units called intervals. The term "cycle" refers to a specific type of sequence for selections on the data flow performed in a succession of clock intervals. For example, there is an instruction - fetch cycle during which an instruction containing information about a transformation is brought from storage to an ALU register. At each clock interval within the cycle, an elementary operation is performed such as routing the storage location of the instruction to the storage access mechanism, or routing of the instruction obtained to an ALU register.

5. Input/Output - Since information in the processor and storage of the computer are represented by electric signals, devices are provided to convert information from human-generated to machine-readable form on input, and in the opposite direction on output. A very common scheme for performing this transducer function uses a punched card. An operator reads the information from hand written or typed documents and enters the information on a key-board of key-punch machine. This machine translates the key strokes into holes on the card. The cards are then sent to the card reader,

which contains the necessary equipment to READ the cards, i.e., sense the hole positions and translate them into the internal electric-signal representation. The results of the processing in CPU, can be communicated to the user by means of some other transducer called printer. In the printer, a translation from internal electric signal form to the human readable character set form takes place.

The punched card, printers, and its associated machines are examples of Input/output devices. Other devices available include type writers, punched paper tape, CRT displays, analog-digital converters and plotters.

3-2 Input/Output Systems

Input/Output operations involve the transfer of information to or from main storage and on I/O device. Input/output devices include such equipment as card-readers and punches, magnetic tape units, disk storage, teletype devices, printers, teleprocessing devices, plotters, and process control equipment.

The I/O device operation is regulated by a control-unit. The control unit function may be housed with the I/O device as in the case of printer, or a separate control unit may be used. In all cases, the control unit function provides the logical and buffering capabilities necessary to operate the associated I/O device. From the programming point of view, most control unit functions merge with I/O device functions.

3-2.1 Input/output Interface:

All communications between the control unit and the channel takes place over a connection called the I/O interface. The I/O interface provides and an information format and control signal sequences that are independent of the type of control unit and channel and provide a uniform mean of attaching and controlling various types of I/O devices.

3-2.2 Channels:

The channel directs and controls the flow of information between I/O devices and main storage. It relieves the CPU of the task of communicating directly with the devices and permits data processing to proceed concurrently with I/O operations. The channel accepts control informations from the CPU in the format supplied by the program and changes it into a sequence of signals acceptable to a control unit. After the operation with the device has been initiated, the channel assembles or disassembles data and synchronises the transfer of data bytes over the interface with main storage cycles. To accomplish this, the channel maintains and updates an address and a count that describe the destination or source of data in main storage. When an I/O device provides signals that should be brought to the attention of the supervisor, the channel again converts the signals to a format compatible to that used in CPU.

A channel may be an independent unit, complete with necessary logical and storage capabilities, or it

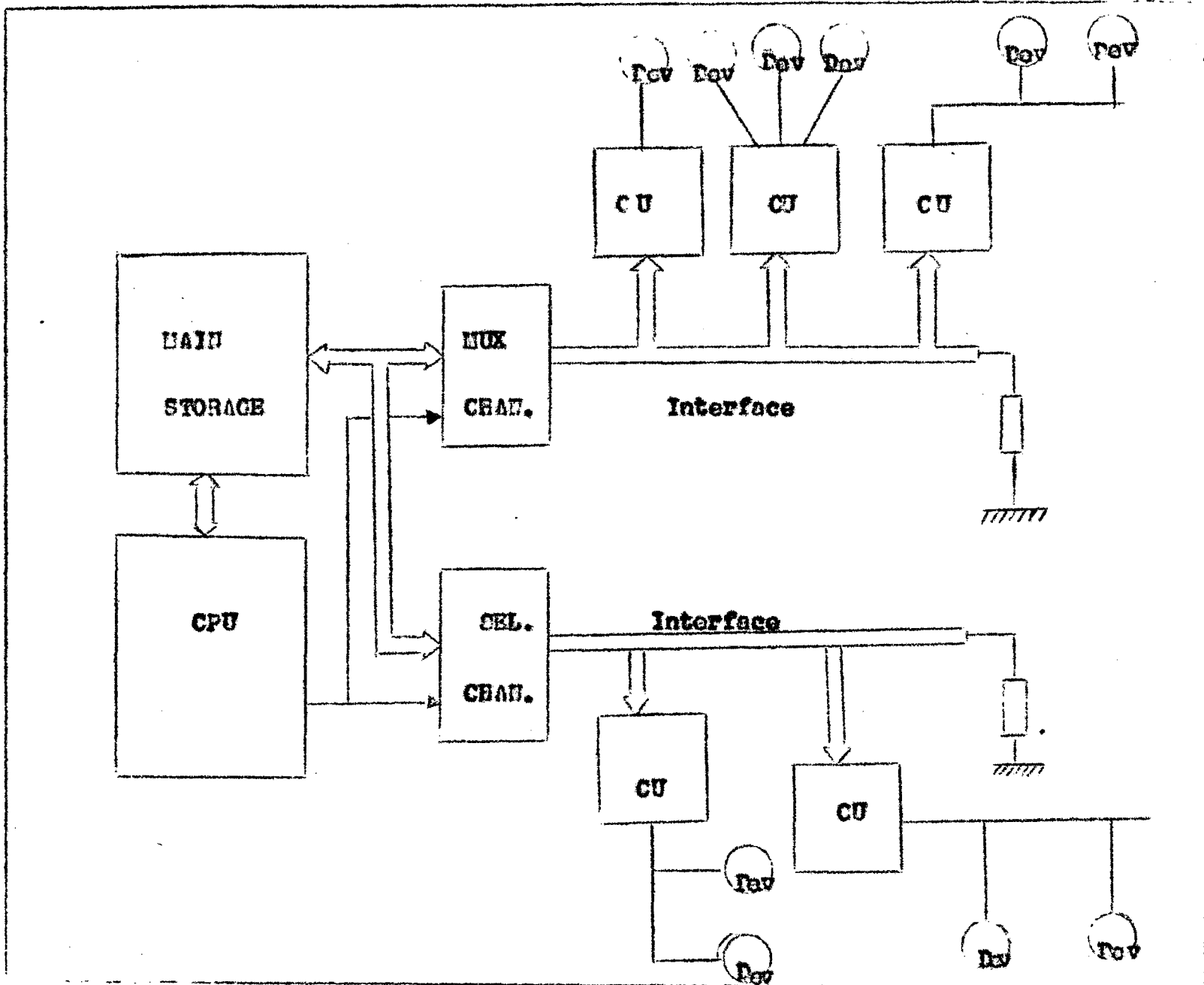


Fig.3-2 I/O Interface.

may share CPU facilities and be physically integrated with the CPU. The channel facilities required for sustaining a single I/O operation are termed as a SUBCHANNEL. The subchannel consists of the channel storage used for recording the addresses count, status and control information associated with the I/O operation. The mode in which a channel can operate depends upon whether it has one or more subchannels. In most of the computer, there are two types of channels.

1. SELECTOR CHANNEL

2. MULTIPLEXOR CHANNEL

Channels have two modes of operation: BURST and MULTIPLEX. In the Burst mode, the data transfer facilities of the channel are monopolized for the duration of transfer of a burst of data. Other devices attached to the channel cannot transfer data until the burst ceases. The selector channel functions only in the burst mode and so it has only one subchannel.

The multiplexor channel functions in either the burst mode or in the multiplex mode. In the multiplex mode, the multiplexor channel can sustain concurrent I/O operations on several subchannels. Bytes of data associated with different I/O devices are interleaved and routed to or from the desired locations in main storage. The I/O interface is time shared by a number of concurrently operating I/O devices, each of which uses its own subchannels

3-2.3 INPUT/OUTPUT INSTRUCTIONS

I/O operations are initiated and controlled at three levels by information with three type of formats: INSTRUCTIONS, COMMANDS, and ORDERS. Instructions are decoded and executed by the CPU and are part of the CPU program (first level). Commands are decoded and executed by the channels and initiate I/O operations, such as reading and writing (2nd level). Both Instructions and commands are fetched from main storage and are common to all types of I/O units, although the modifier bits in the command code may specify device dependent conditions for the execution of a data transfer operation at the device.

Functions peculiar to a device are specified by orders. Orders are decoded and executed by I/O devices. The control information specifying an order may appear in the modifier bits of a control command code, may be transferred to the device as data during a control or write operation, or may be made available to the device by other means.

The CPU controls I/O operations by means of four I/O instructions:

<u>Name</u>	<u>Mnemonic</u>	<u>Type</u>	<u>Code</u>
1. START I/O	SIO	SI,C	9C
2. TEST I/O	TIO	SI,C	9D
3. HALT I/O	HIO	SI,C	9E
4. TEST CHANNEL	TCH	SI,C	9F

TH 814

All I/O instructions use the following SI format:



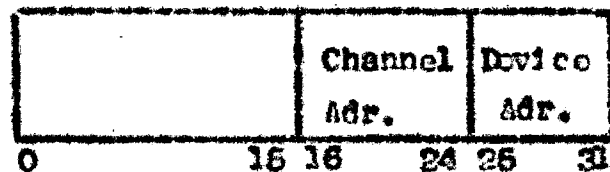
Where, Bits 0-7 contains instruction code

Bit 8-15 are ignored

Bit 16-19 BI field (designates a basic register)

Bit 20-31 DI field (displacement)

The sum obtained by the addition of the content of register BI and the content of the DI field identifies the channel and the I/O device. This sum has the format:



Bit positions 0-15 are ignored.

START I/O (SIO) instruction: This is used to initiate an I/O operation when CPU encounters an SIO instruction, it activates, the channel and sends the device address to the channel. Channel intern initiates the device selection sequence which results in the issuing of a command.

On multiplexor channel, SIO causes the addressed device to be selected and the operation to be initiated only after the channel has serviced all outstanding requests for data transfer for previously initiated operations. This instruction is executed only when CPU is executing in the supervisory mode.

HALT I/O(HIO) instruction:

HIO instruction terminates a current I/O operation at the addressed sub-channel or channel. The subsequent state of the channel depends on the type of the channel. The instruction HIO is used only when the CPU is executing in the supervisor state.

TEST CHANNEL(TCH) instruction:

Execution of the TCH instruction sets the condition code in the PSW to indicate the state of the channel addressed by the instruction. The resulting condition code indicates one of the following channel available, interruption condition in channel, channel working, or channel not operational.

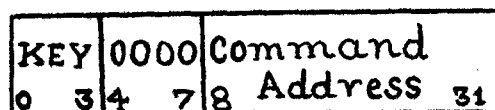
3-2.4 I/O operation initiation:

An I/O operation is initiated by a SIO instruction. If the necessary channel and device facilities are available, SIO is accepted and the CPU continues its program. Then after channel independently governs the I/O device specified by the instruction. Successful execution of SIO instruction causes the channel to fetch a CHANNEL ADDRESS WORD(CAW) from the main storage location 72. The CAW specifies the byte location in main storage where the channel program begins.

The format of the CAW is shown below. Bits 0-3 specify the storage protection key that will govern the I/O operation. Bits 4-7 must contain zeros. Bits 8-31 specify the location of the first CHANNEL COMMAND WORD(CCW).

The byte location specified by the CAW is the first of eight byte of information that the channel fetches from the main storage. These 64 bits of information are called channel command word (CCW). Only the SIO instruction may cause the channel to fetch CCW's.

CAW:



One or more CCW's make up the channel program that directs channel operation. A channel command word can specify one of six commands:

READ

WRITE

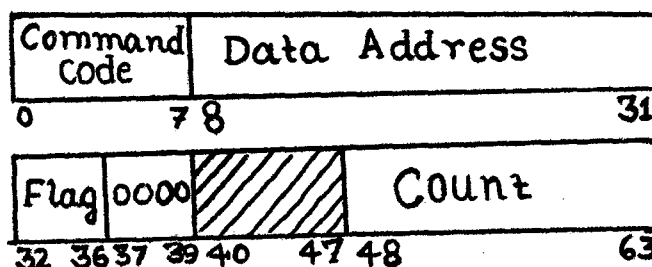
READ BACKWARD

CONTROL

SENSE

TRANSFER IN CHANNEL

If more than one CCW is to be fetched, the CCW's are to be fetched sequentially, except when transfer in channel is encountered. The Figure given below shows the format of the CCW



The Command code specifies the operations to be performed (Read, Write, Rewind etc.).

The Data Address specifies the first byte location in main storage for a data transfer type of operation.

The Flag bits may specify chaining to another CCT, suppression of a possible incorrect length indication, etc. The count specifies the number of bytes for a data transfer operation.

3-2.5 I/O Commands:

The command code, bit positions 0-7 of the CCT, specifies to the channel and the I/O unit the operation to be performed. The two low order bits (or when these bits are 00, the four low order bits) of the command code identify the operation to the channel. The channel distinguishes among the following four operations:

- Output forward (write, control)
- Input forward (Read, Sense)
- Input backward (Read backward)
- Branching (Transfer in channel)

The channel ignores the high order bits of the command code. Commands that initiate I/O operations (Write, Read, Read backward, Control, and Sense) cause all eight bits of the command code to be transferred to the I/O unit. The command code assignment is listed in the following table. The symbol 'X' indicates that the bit position is ignored; K identifies a modifier bit.

Code	Command
XXXX0000	Invalid
XXXX0100	Sense
XXXX1000	Transfer in Channel
XXXX1000	Read backward
XXXX1001	Write
XXXX1010	Read
XXXX1011	Control

The modifier bits specify to the unit how the command is to be executed. They may cause, for example, the unit to compare data received during the Write operation with data previously recorded, and they may specify such conditions as recording density and pointy. For the control command, the modifier bits may contain the order code specifying the control function to be performed.

Whenever the channel detects an invalid command during the initiation of a command, the program-check condition is generated when the first CCW designated by the CAC contains an invalid command code, the status portion of the CCW with the program-check indication is stored during the execution of START I/O.

WRITE: A Write operation is initiated at the I/O device, and the subchannel is set up to transfer data from main storage to the I/O unit. Data in storage are fetched in an ascending order of addresses, starting with the address specified in the CCW. Bit positions 0-6 of the CCW contain modifier bit (mmmmmm01).

READ: A read operation is initiated at the I/O unit, and the subchannel is set up to transfer data from the unit to main storage. Data in storage are placed in an ascending order of addresses, starting with the address specified in the CCW.

READ BACKWARD: A Read Backward Operation is initiated at the I/O unit, and the subchannel is set up to transfer data from the unit to main storage. The channel places the bytes in storage in a descending order of addresses, starting with the address specified in the CCW, but

fetches CCW's in an ascending sequence.

CONTROL: A control operation is initiated at the I/O unit, and the subchannel is set up to transfer data from the Main storage to the I/O device. The I/O unit interprets the data as control information. The control information is fetched from the main storage in ascending order of addresses, starting with the address specified in the CCW. A control command is used to initiate at the I/O unit an operation not involving transfer of data, such as backspacing or rewinding magnetic tape or positioning a disk access mechanism. For more control functions, the entire operation is specified by the modifier bits in the command code, and the function is performed over the I/O interface as an immediate operation.

A control command code containing scraps for the six modifier bits is defined as No-operation. The No-operation order causes the addressed unit to respond with channel end without causing any action at the unit. Other operations that can be initiated by means of the control command depend on the type of I/O device. Bit positions 0-5 of the CCW contain modifier bits.

SENSE: A sense operation is initiated at the I/O unit, and the subchannel is set up to transfer data from the I/O unit to main storage. The data are placed in storage in an ascending order of addresses, starting with the address specified in the CCW.

Data transferred during a Sense operation provide information concerning both unusual conditions detected in the last operation and the status of the device. The status information provided by the sense command is more detailed than that supplied by the unit status byte and may describe reasons for the unit check condition.

For most I/O unit, the first six bits of the first sense data byte(sense byte 0) are common to all I/O units that have this type of information. The six bits are independent of each other and are designated as follows:

<u>Bits</u>	<u>Designations</u>
0	Command Rejest
1	Intervention Required
2	Bus Out Check
3	
4	Data check
5	Overrun

The meaning of the individual bits are self explanatory.

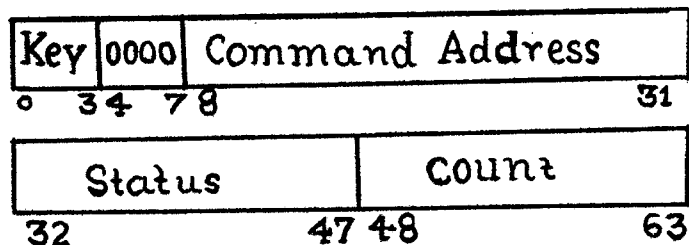
TRANSFER IN CHANNEL: The next CCW is fetched from the location designated by the data address field of the CCW specifying transfer-in-channel. This command does not initiate any I/O operation at the channel, and the I/O device is not signaled of the the execution of the command. The purpose of the transfer-in channel command is to provide chaining between CCW's not located in adjacent double word locations in an ascending order of addresses.

3-2.6 Input/Output Terminator:

I/O operations terminate with the device and channel signaling end of operation and a request for an I/O interruption.

A command can be rejected during an attempt to execute a START I/O, by such conditions as busy conditions, a channel programming error etc. The condition code set in the Program Status Word (PSW) by an unsuccessful, SIO instruction will indicate one of the following: that a channel status word (CSW) has been stored to detail the conditions that preclude the initiation of the I/O operation, that the equipment is busy, or that the addressed equipment is not operational.

The Channel Status Word (CSW) provides information about the termination of an I/O operation. It can be formed or reformed by start I/O, Test I/O, Halt I/O, or by an I/O interruption. The instruction test channel does not affect CSW. The format of the CSW is as shown below:



Key field contains the protection key used in the last operation.

Command Address: specifies the (location +8) of the last CCW used.

Status field contains a unit status byte and a channel

status byte. The unit status byte may indicate one or more conditions- such as control unit end, channel end, Device end, attention, busy, status modifier, unit check and unit exception. The channel status byte may indicate a channel programming error, a channel data check, a channel control check, protection check, interface control check, chaining check, program controlled interruption, incorrect length, Count field specifies the residual count of the last CCW used.

3-2.7 I/O Interruption:

I/O interruptions are caused by termination of an I/O operation or by operator intervention at the I/O device. An I/O interruption stores the current PSW in the I/O old PSW location, and places the I/O new PSW in control of the system. The I/O new PSW, when made current by an I/O interruption, may cause CPU interrogation of the channel status word, or take whatever action is considered appropriate by the programmer.

An I/O interruption request may be initiated by an I/O interruption condition in a device, a control unit, or a channel. When a channel has multiple I/O interruption requests pending it establishes a priority sequence for them before initiating an I/O interruption request to the CPU conditions responsible for I/O interruption are request remain pending in the I/O devices or channels until they are accepted by the CPU.

3-2.8 BASIC PROCEDURE FOR A DATA TRANSFER OPERATION

A START I/O instruction is used to initiate data transfer to or from an I/O device. To perform such an operation, it is necessary for the programmer to:

1. Establish a channel command word (CCW) or a list of CCW's in main storage.
2. Load the channel address word (CAW) with the address of the first byte of the first CCW in the channel program.
3. Load the channel and device address in the START I/O instruction to be used for the operation.
4. Set the system mask to disable all channels for I/O interruptions.
5. Issue the START I/O instruction.
6. Test the condition code established in the current PSW by termination of the START I/O.

Condition code 0 indicates that the I/O operation has been initiated and that the channel is proceeding with its execution. If an I/O interruption is desired upon termination of the operations, the pertinent channel mask bit must be set to 1 (an appropriate I/O new PSW must have been established previously).

Condition code 1 indicates that a channel status word (CSW) has been stored; its status bytes should be examined to determine why the desired operation was not initiated.

Condition code 2 indicates that the channel or subchannel addressed by the START I/O instruction was found to be busy with a previously initiated operation. If an

I/O interruption from the operation already in progress is desired, the channel mask bit must be set to one.

Condition code 3 indicates that the addressed equipment is not operational; a message to the operator may be initiated.

Between the time a START I/O instruction is decoded by the CPU, and the time the CPU is released by the channel with condition code 0 set in the current PSW, the channel performs many functions. The CAW must be fetched, the first CCW must be fetched, the CAW and CCW must be tested for validity, etc. After a START I/O results in condition code 0, the operation continues until terminated. Termination of an I/O operation causes a request for an I/O interruption.

3-3 MULTIPLEXOR CHANNEL:

3-3.1 BASIC DATA PATHS AND BLOCK DIAGRAMS

General block diagram of the CPU and I/O channels is given in Fig 3.3. The channel and the CE panel are connected to the CPU by the block of channel control (BCC). BCC consists of assembly logic and some other schemes (for example, register for channel control-RCC) which will be discussed later.

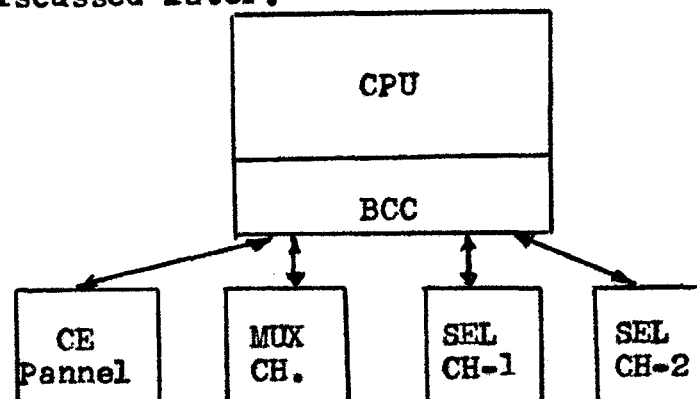


Fig.3-3 General Block diagram of the CPU & I/O System.

3-3.2 MICROPROGRAM CONTROL OF THE I/O SYSTEM:

The data and control information transfer between the channels and the external devices is realized by microprograms and control circuits. Microprogram control is applied for all channels. In the selector channels, a special control Ckt. is used for data transfer. The I/O microprograms are located in the 1st block of ROM and use control signals from the microprogram block of the CPU.

3-3.3 MULTIPLEXER CHANNEL BLOCK DIAGRAM AND DATA FLOW:

EC 1020 multiplexor channel (EC) coordinates and controls the work of a great number of small rate I/O devices: the maximum transfer rate in multiplex mode is 14-26 Kbytes/sec. and in burst mode this rate is upto 200 Kbytes/sec.

The block diagram for multiplexor channel is shown in Fig.3.4. The multiplexor channel has a comparatively small own hardware because during the execution of channel operation it shares the hardware of the CPU and so a computing process at the same time cannot be realized. The multiplexor channel uses the multiplexor and the local storage. In the multiplexor, which is a part of the main storage, the subchannels are situated. In the subchannels the UCW (unit control words) are stored. The local storage is a complementary hardware in the execution of the channel microprograms.

In the multiplexor channel, the control and the data transfer are realized partly hardware and partly microprogram. The own hardware of the multiplexor channel includes several registers and control schemes:

- register of output information RR2, which controls the BUS-OUT interface lines. This register consists of 8 informational bits and of one parity bit. The functional schemes of the register RR2 and other multiplexor channel registers are described in paragraph 3.4.
- register of input information RR3. This register is connected with the BUS-IN interface lines and 8 informational bits and one parity bit.
- control register for in-bound interface lines RR4, connected with the in-bound tag and control lines. The register has the following bits:

RR4:

0 - (IFC)	Interface free condition.
1 - (TOPL-IN)	Trigger for the interface signal OPL-IN
2 - (TADR-IN)	Trigger for the interface signal ADR-IN
3 - (TSTA-IN)	" " " STA-IN
4 - (TSRV-IN)	" " " SRV-IN
5 - (TEEL-IN)	" " " SEL-IN
6 - (TRRQ-IN)	" " " REQ-IN
7 - (TSEL U)	Trigger for control unit selection. This shows that selection is initiated by the channel.

- register for output channel control RR6, which is connected with out-bound interface tag and control lines. The register consists of the

following bits:

RRQ

0	-	(TCCL-OUT MC)	Trigger for the interface signal	SEL-OUT
1	-	(TADR-OUT MC)	" " "	ADR-OUT
2	-	(TCMD-OUT MC)	" " "	CLD-OUT
3	-	(TSRV-OUT MC)	" " "	SRV-OUT
4	-	(TOPL-OUT MC)	" " "	OPL-OUT
5	-	(TSUP-OUT MC)	" " "	SUP-OUT
6	-	(TSEL-OUT MC)	" " "	SEL-OUT
7	-	(TCDEC)	Trigger for chain data	

- error register RRE. This register fixes the error situations during the operations of the multiplexer channel and has the following bits:

RRI

0	-	(TMISCC)	trigger for multiple interface signals check
1	-	(TISCC)	trigger for interface signals check
2	-	(TIPC)	trigger for interface parity check
3	-	(TOCC)	trigger for output ALU lines C check
4	-	(TCDC)	trigger for channel data check
5	-	(PCCC)	potential channel control check
6	-	(PIOC)	potential interface operation check

- error register RRE. This register indicates an invalid response, slow response or no

response from I/O device to a tag-out
interface signal and has the following bits:

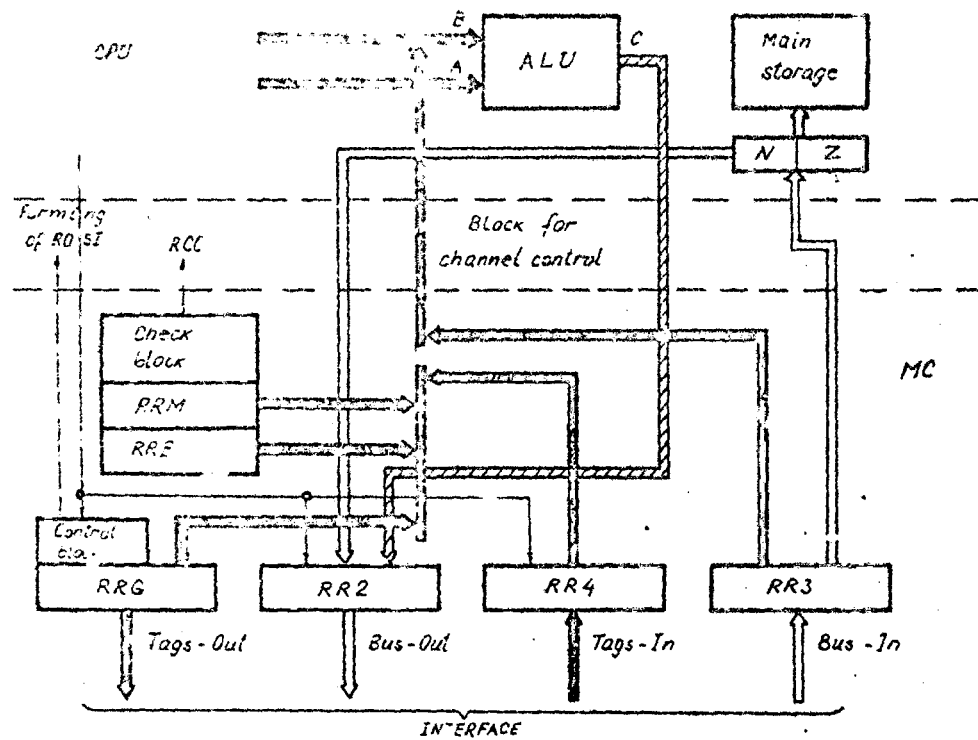
BBE

-
- 0 - (T1ISEL - OUTEC) first trigger for interface operation
interrupt after the signal SELOUT
 - 1 - (T2ISEL - OUTEC) second trigger for interface operation
interrupt after the signal SELOUT
 - 2 - (T1ISRV - OUTEC) first trigger for interface operation
interrupt after the signal SRV-OUT
 - 3 - (T2ISRV - OUTEC) second trigger for interface operation
interrupt after the signal SRV-OUT
 - 4 - (T1ICLD - OUTEC) first trigger for interface operation
interrupt after the signal CLD-OUT
 - 5 - (T2ICLD - OUTEC) second trigger for interface operation
interrupt after the signal CLD-OUT
 - 6 - (T1OPL - IDEC) Trigger for interface operation
interrupt after the signal OPL-IO
 - 7 - (T1OIEC) Trigger for interface operation
interrupt.
-

The channel hardware also includes some
control schemes:

- Scheme for forming ROS interrupt (ROSI) signal,
- Scheme for forming RMB(reset M/C errors)
signal etc.

The data transfer between the main storage and
the control unit is done via the input information register



Figure

RR3 and output information register RR2, connected with the register RZ of the main storage. The data transfer to the register RR2 and from the RR3 is microprogram controlled.

The input of the register RR2 is also connected by the assembly schemes in the ECC to the output C of the ALU and for that reason the output of the RR3 is connected by the assembly logic in the ECC to ALU-B input.

The transfer of control signals to the corresponding interface lines from the channel is from the register RR6. Some triggers of this register are set as bits of RR1 register. For analysing the contents of the RR6 register, its output is connected to the input 'B' of the ALU.

The interface signals from the external device are transferred for analysis via register RR4 to the input "B" of the ALU. The error registers RR8 and RR9 are also connected to the input "B" of the ALU and their contents can be analysed by the microprogram.

All registers in the next mux channel are connected either to input "D" or to output "C" of the ALU. This connection requires the maximum no of register bits to be eight without a parity bit.

3-3.4 HARDWARE OF THE MUX CHANNEL

The mux channel hardware can be divided into three main groups:

1. The first group includes all schemes for maintaining the standard interface dialogue.

Registers RR0 and RR4 and some control triggers belong to this group.

2. The second group includes registers for servicing the data flow from the main storage to external devices and vice versa. Registers RR2 and RR3 belong to this group.

3. Registers RRE and RREI are in the third group and serve for checking and handling errors in channel operations.

3-4 General I/O Interface Sequence - Multiplexor Channel

The I/O interface provides an information format and a signal sequence common to all I/O units. The interface consists of a set of lines that can connect a number of control units to the channel. Except for the signal used to establish priority among control units, all communications to and from the channel occur over a common bus, and signal provided by the channel is available to all control units.

Control unit is a piece of hardware that provides the logical capabilities necessary to operate and control an I/O device and adapts the characteristics of each device to the standard form of control provided by the channel. The control units accept control signals from the channel, controls the timing of data transfer over the I/O interface, and provides indications concerning the status of the device.

The I/O interface (channel to control unit) is the communication link between the CPU channel and the various

I/O control units in the EC 1020 system. It employs information formats and control signals sequencing to provide uniform means for attaching and controlling various type of control units.

This interface is so designed that it provides:

- a degree of consistency of I/O programming over a wide range of control units.
- ready physical connection to EC 1020 system channels of control units designed by any manufacturer to operate with this interface.
- ability to physically accommodate future control units designed to meet the parameters of this interface.
- an interlocked interface operations that is not very time - dependent, this feature increases the range of control units that may be attached.
- an operation applicable to both multiplex and burst mode operations as well as many control operations and channel-to-channel transmissions;
- upto eight control units serviced per set of lines.

At any one time, only one control unit can be logically connected to the channel. A control unit remains logically connected to the interface until it transfers the information it needs or has, or until the channel signals it to disconnect.

The rise and fall of all signals transmitted over the interface are controlled by interlocked responses.

Interlocking removes the dependence of the interface on ckt. speed, and makes it applicable to a wide variety of ckts. and data rates. Further, interlocking permits connecting control units of different ckt. speeds to a single channel.

The I/O activity commences with the START I/O instruction (96) which is encountered in the program. It indicates the channel address together with the address of the device which is to be used. This device address represents the chosen peripheral device (Module), as well as the control unit through which it operates. The device address is placed on the bus-out to all connected control units and by means of patch-cards, the chosen one is identified.

After addressing the device, its fate, or the condition of its readiness, must be reported to the processor. This is achieved by the status byte, which is prepared in the control unit and sent to the channel, and by the two bit condition code, which is prepared in the channel and sent to the processor. The flow chart shows the various situations and the status or conditions which can be reported.

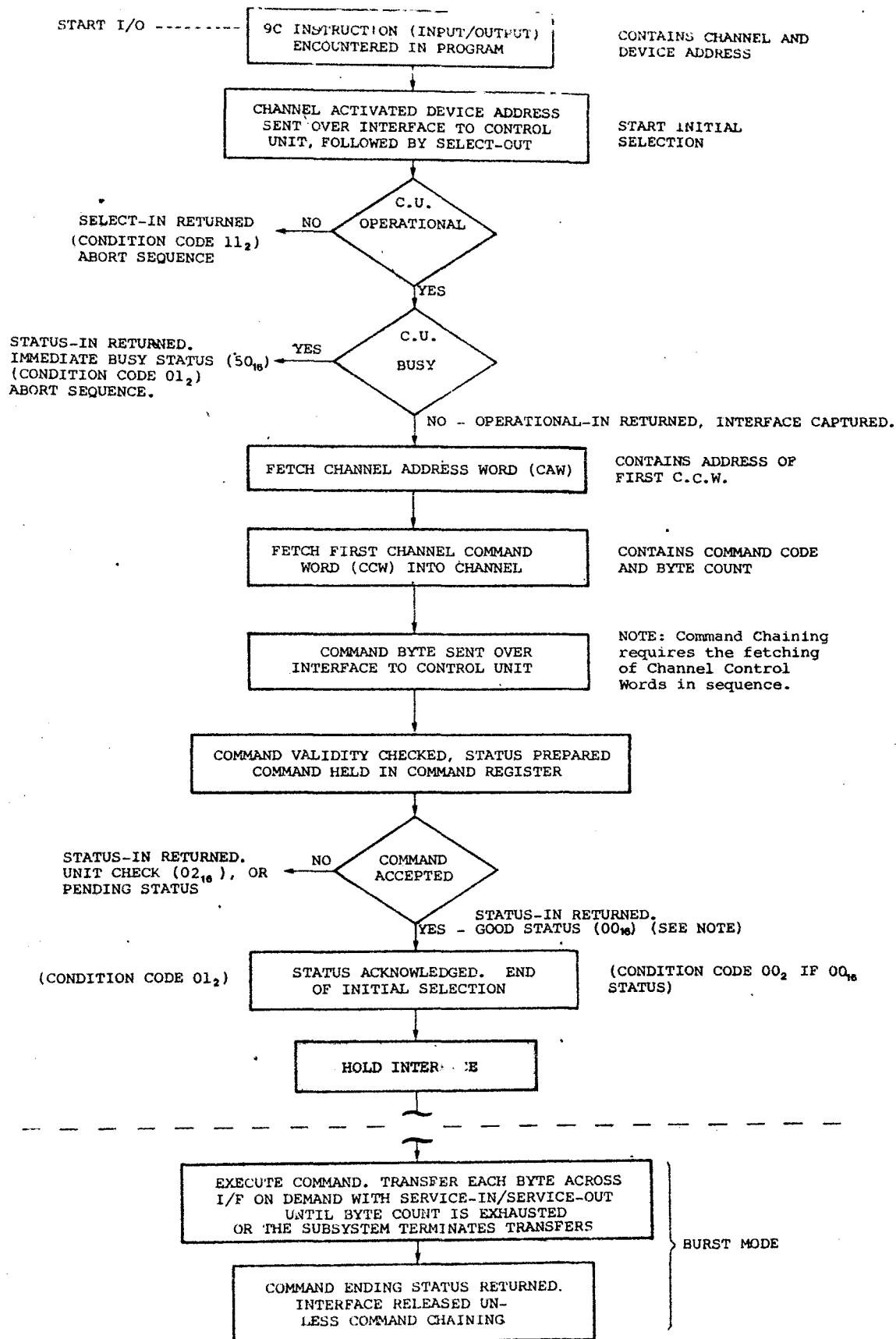
Further activity depends upon control words which have been previously prepared and reside in memory. These words namely CAW and CCW, are used to provide the control of all activity with the current selected device. The CAW points to the first CCW which when fetched into the channel, provides the necessary controlling parameters. It contains the command code, as per the device instructional repertoire, with certain flag bits, which indicate special operations.

The flag bits may specify, for example, whether the command is self standing or is issued as a part of a chain of commands. The CCV also contains the address where the data is to be located in memory or where data is to be placed during the current transactions. The amount of data bytes which have to be transferred with the current command is also specified in the CCV i.e. the byte Count. The CCV is fetched into the channel and is executed in order to initiate operation. It continually updates this counter as each byte transfer takes place. It is therefore maintained within the channel, always to reflect the current situation.

Prior to the start of any data transfer, the command byte, extracted from the CCV, must be delivered to the control unit and checked for validity, relative to the addressed module. If the command is invalid, the status of unit check(02₁₆) is returned. Alternatively, any status condition which may exist (pending) for the selected module is reported by means of the status byte at this time. If the command is accepted, the status returned depends upon the type of command which was issued. Certain command not involving a data transfer, may be executed almost immediately. Therefore, the status byte returned at this time may contain command - ending status. If the command involves a data transfer, a zero status will be returned which indicates an acceptance to proceed into the execution phase.

The acknowledgement of status by the channel signifies the end of the Initial-selection phase and since data is to be transferred in the burst mode of operation,

GENERAL I/O SEQUENCE



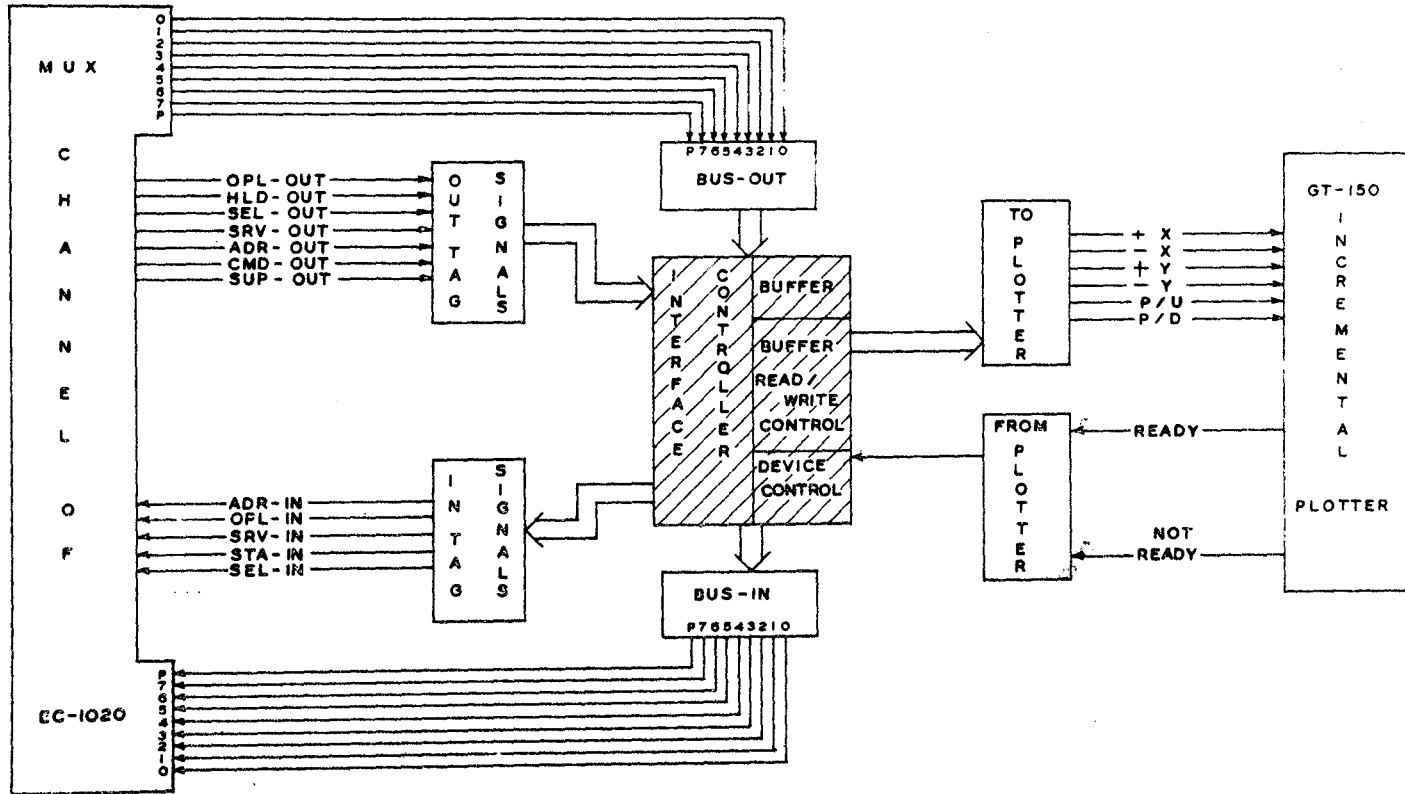
the interface with the selected device is held until the completion of the data transfer.

CHAPTER - 4SYSTEM COMPONENTS DESCRIPTION

4-1 System Block Diagram: The complete system of I/O interfacing can be divided into three portions; Channel portion, controller portion and device portion, Channel provides the standard set of lines (signals) to the controller. These lines can be grouped in the following general way:

- | | |
|-------------------------------------|--|
| Information lines | <ul style="list-style-type: none"> - BUS-IN and BUS-OUT used to transfer an eight bit byte plus one parity bit, in parallel. |
| Control or Tag lines | <ul style="list-style-type: none"> - ADDRESS-IN and ADDRESS-OUT SERVICE-IN and SERVICE-OUT STATUS-IN COMMAND-OUT Used to define the contents of the Bus-lines which, at different times, can carry any of the following: <ul style="list-style-type: none"> The Device Address The Command Data Status(In only) |
| Selection and Priority Lines | <ul style="list-style-type: none"> - SELECT-OUT and SELECT-IN HOLD-OUT REQUEST-IN OPERATIONAL-OUT OPERATIONAL-IN |

INTERCONNECTION : CHANNEL TO PLOTTER



Different combinations of these signals convey different message to the controller. The main function of the controller is to understand these messages and act accordingly. All the signals, except data on the BUS-OUT, are meant for controller. Controller accepts data from the channel and sends it to device in a format acceptable to it. Thus data formatting is done by the controller.

The I/O device also has standard set of lines. Some of them Input and others output. It is operated by these input lines and informs its status to the controller with the help of output lines. The system block diagram is shown in Figure 4.1. There are six input lines two output lines in this device. The input lines are: +X, -X, +Y, -Y, pen up and pen down. The two output lines are: Ready and Not ready.

4-2 I/O Interface Structure and Signal Sequence:

The I/O interface(channel to control unit) is the communication link between the CPU channel and the various I/O control units in EC 1020 system. The following are the important features of this standard interface:

- i) Ability to physically accommodate future control units designed to meet the parameters of this interface;
- ii) An interlocked interface operation that is not very time-dependent; this feature increases the range of control units that may be attached;
- iii) An operation applicable to both multiplex and burst mode, operation as well as many control operations and channel to channel transmissions.

iv) Upto eight control units serviced per set of lines.
 Informations in the form of data, status and sense information, control signals, and device addresses, are transmitted over the time and function shared lines of interface. These 34 lines are shown in the figure 4.2.

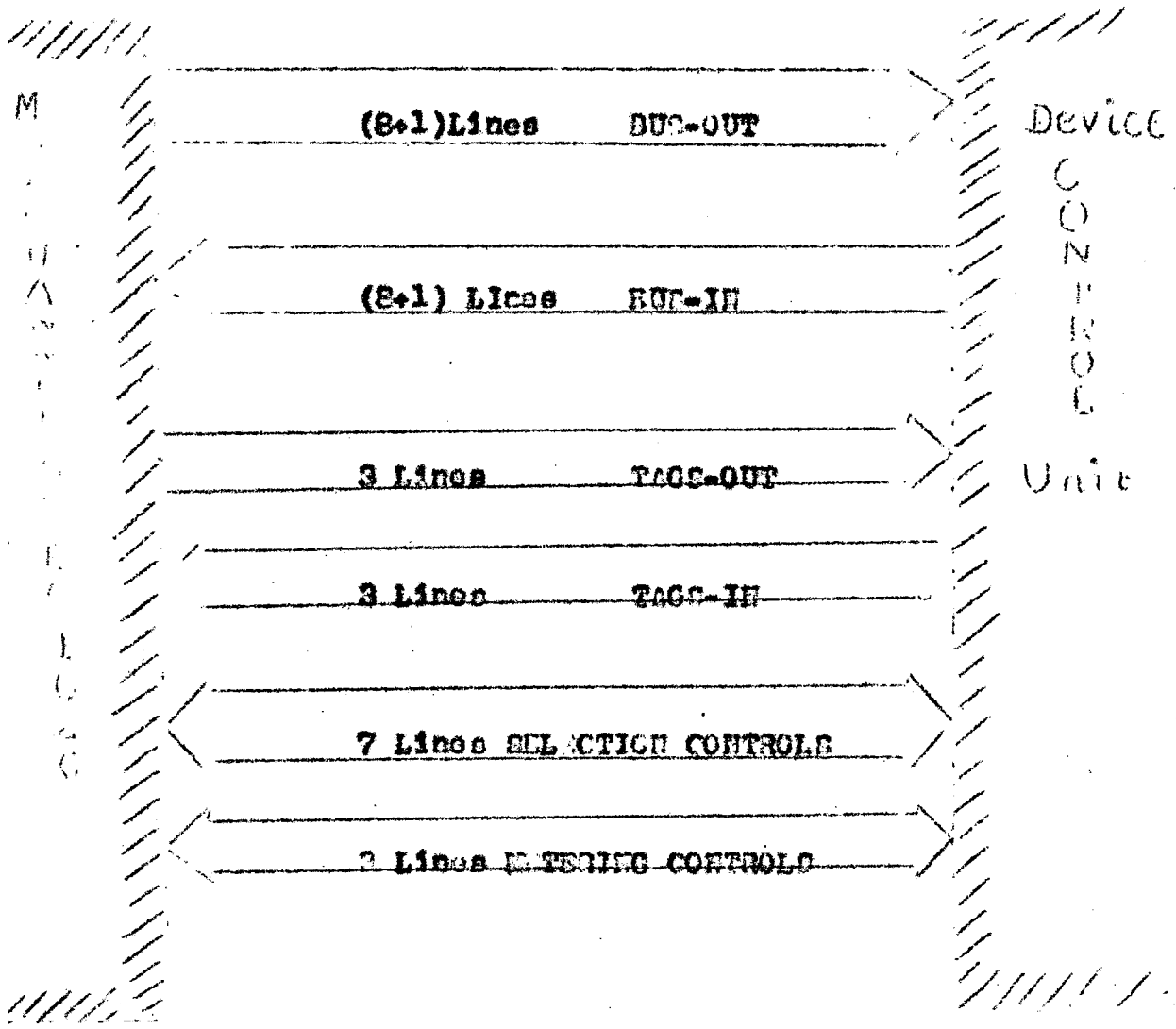


Fig. 4.2

Except for signals used to establish selection control, all communications to and from the channel occur over a common bus; i.e. any signal provided by the channel is available to all control units. At any one time,

however, only one control unit can be logically connected to the channel. Selection of a control unit for communication with the channel is controlled by a signal (passing serially through all CUE) that permits sequentially, each control unit to respond to the signal provided by the channel. A control unit remains logically connected to the interface until it transfers the information, it need or has, or until channel signals it to disconnect.

The rise and fall of all signals transmitted over the interface are controlled by interlocked responses. Interlocking removes the dependence of the interface on ckt. speed, and makes it applicable to a wide variety of circuits and data rates. Further, interlocking permits connecting control units of different ckt speeds to a single channel.

The I/O interface lines are grouped in the following way:

- BUS-OUT - used to transmit information (data, device address, commands, control orders) from channel to the control unit.
- BUS-IN - used to transmit information (data, selected device identification, status information, sense data) from device control unit to the channel.
- PAGE-OUT - used for identifying the type of information on the BUS-OUT (data, address, command, or control

- information). They are: ADR-OUT, CMD-OUT, SRV-OUT.
- TAGS-IN** - used for identifying the type of information on the BUS-IN (date, device identification, status, sense byte). They are: ADR-IN, STA-IN, SRV-IN.
- SELECTION CONTROLS** - used for the scanning of, or the selection of, attached I/O units. They are: OPL-OUT, OPL-IN, HLD-OUT, SEL-OUT, SEL-IN, SUP-OUT, REQ-IN.
- METERING CONTROLS** - used for the conditioning of usage meter located in the various attached units.

4-2.1 LINE DEFINITION:

Each Bus is a set of 9 lines consisting of eight information lines and one parity line. Information on the buses is arranged so that bit position 7 of a bus always carries the low order bit within an eight-bit byte. The highest order bit is in position 0 and intervening bits are in descending order from position 1 to position 6.

4-2.1.1 TAGS-OUT: It consists of three lines -

ADDRESS-OUT(Adr-out) - This is a tag line from the channel to all attached control units. It provides two functions.

1. I/O device selection - Adr-out is used to signal all the control units to decode the I/O device address on BUS-OUT. The CU when recognises the address, must respond by raising its Opl-In line when its , incoming sel-out rises with

address out still up. The rise of Adv-out precedes the rise of set-out by a minimum of 400 ns and follows the placing of the I/O unit address on Bus-Out by at least 250ns.

2. Disconnect Operation - If Hld-out is down and Adv-out rises or Adr-out is up and Hld-out falls, the presently connected CU must drop its Opl-In, thus disconnecting from the interface.

COMMAND-OUT(Cmd-Out) - This is a tag line from channel to all attached I/O devices used to define the content on the Bus-Out. The presence of this signal on the interface conveys different meaning in different situations. In the case of channel initiated sequence, the rise of this signal indicates that the content of the Bus-out is a command byte. While in the case of device initiated sequence, the rise of Cmd-out signal does not define the content of the Bus-out but simply indicates to 'Proceed' further.

SERVICE-OUT(Srv-Out) - This is also meant for defining the content of the Bus-out. When Srv-out rises in response to Srv-In during the execution of a write or control command, the nature of the information on Bus-out depends on the type of operation. For example, during a write operation it contains data to be recorded by the I/O device. During a control

operation, it can specify an order code or a second-level address within the control unit or I/O device.

4-2.1.2 TAGG-IN:

It consists of three lines:

ADDRESS-IN(Adr-In) - is used to signal the channel when the addresses of the currently selected I/O device has been placed on the Bus-In. The channel responds to Adr-In by Cmd-Out. The rise of Adr-In indicates that the address of the currently addressed I/O device is available on Bus-In. Adr-In must fall so that Cmd-Out may fall. Adr-In cannot be up concurrently with any other inbound Tag line.

STATUS-IN (Sta-In) - is used to signal the channel when the device has placed status information on the Bus-In. The channel responds with either Srv-Out or Cmd-Out depending on whether it accepted the status byte or not.

SERVICE-IN(Srv-In) - is used to signal to the channel when the selected I/O device wants to transmit or receive a byte of information. The nature of information associated with Srv-In depends on the operations and the I/O unit. During Read, Read Backward, and sense operations, Srv-In rises when information is available on Bus-In. During write and control operations, Srv-In rises

when information is required on the Bus-Out. It cannot be up concurrently with any other inbound tag line.

4-2.1.3 SELECTION-CONTROL:

These lines control the selection of the device and require further description, as follows:

SELECT-OUT (Sel-Out) - The signal is generated in the channel and sent to the control unit placed physically first in line. The action within the control unit, on receipt of this signal, depends upon the configuration of its wired priority patchboard. Two configurations are possible namely high or low priority. The propagation of this signal is discussed in section 4-2.2.

HOLD-OUT(Hld-Out) - This signal generated in the channel, allows the gating of Sel-Out within the control units. Without Hld-Out, Sel-Out wouldn't function correctly. Hld-Out is used to obtain a more rapid response on disconnect than could be obtained by dropping Sel-Out. The reason for this is that the trailing edge of Sel-Out would take time to be propagated to the connected control unit.

SUPPRESS-OUT(Sup-Out) - This signal, generated in the channel performs several functions, for example: it can be used to suppress data or status and to suppress Req-In, from a device requiring connection for the transference of a status or data byte.

Sup-Out also indicates command-chaining, allowing a reselection sequence to take place for the purpose of delivering a further command to the same device. Similar activity occurs in the command retry sequence. Finally, Sup-Out also enables a selective reset sequence if present when Opl-Out is inactive.

OPERATIONAL-OUT(Opl-Out) - This line acts as an interlock for all lines originating at the channel. Except for Sup-Out all lines from the channel are significant only when Opl-Out is active. It indicates that the interface area of the channel is ready. The deactivation of Opl-Out, in conjunction with the absence of Sup-Out, indicates a System Reset action. De-activating Opl-Out during the presence of Sup-Out, indicates a Selective Reset to a currently operating device i.e. one where Opl-In is active. Whenever Opl-Out is dropped, any operation in progress over the interface will cease.

OPERATIONAL-IN(Opl-In) - This line when activated by the selected device, indicates that it is operational and ready to accept a command. This is a positive response by the device to the Sel-Out signal generated in the channel. With this response the interface is captured for the selected device and the sequence can continue. This is true for all I/O activity whether the selection sequence is initiated by the channel or the device.

REQUEST-IN (Req-In) - indicates that the control

unit is ready to present status information or data and thus is requesting a selection sequence. Req-IN should be dropped when Opl-In rises, unless additional selection sequences are required, or when the control unit is no longer ready to present the status information or data. Req-In must in no case fall later than 250ns after the fall of Opl-In, if the sequence satisfies the service requirements of the control unit.

SELECT-IN (Sel-In) - is a line that extends the Sel-Out signal from the jumper in the terminator block to the channel. It provides return path for the sel-out signal.

+6V D.C. SUPPLY - This voltage level on the interface is used to enable a certain part of the interface logic within the connected devices. Thus, effectively, the interface areas in the devices, whilst physically remote from the channel are electrically part of it. This enables certain interface activities to take place independently from the power supply condition of the control unit. For example Sel-Out can still be propagated from a non-operational control-unit.

4-2.1.4 BUS-OUT:

It consists of 9 lines (8 information lines and 1 parity line). Thus on this bus at one time one byte wide information can be transmitted from channel to device control unit. At different times, different type of information can be transmitted on this bus such as data byte, I/O address, commands and control

orders. The nature of the information byte presented over Bus-Out can be determined with the help of Tags-Out signal present and the sequence of events occurred.

4-2.1.5 BUS-IN:

It also consists of 9 lines (8 information lines and 1 parity line). On this bus, at one time, one byte wide information can be transmitted from control unit to the channel. At different times, different type of information can be transmitted on this bus such as - data, selected I/O device identification, status information, sense data. The nature of the information byte presented over the Bus-In can be uniquely determined with the help of TAGS-IN signal present and the sequence of events occurred.

4-2.2 SYSTEM EC 1020B INTERLOCKING RULES:

The following rules for direct-current interlocking of signals is used in system EC 1020B.

1. No more than one out-tag may be up at any given time, except for ADR-Out, during the interface disconnect sequence.
2. No more than one in-tag may be up at any given time.
3. An in-tag may use only when all Out-Tags are down, except for the control unit busy sequence.
4. An in-tag may fall only after the rise of a responding Out-Tag, except for Sta-In in the

control unit busy sequence.

5. Srv-Out and Cmd-Out may rise only in response to the up level of an in-tag.
6. Adr-Out for a channel initiated selection sequence may rise only when Sel-In and Sel-Out are down at the channel.
7. Once Adr-Out and Sel-Out have risen for a channel initiated selection sequence, Adr-Out must stay up until after the rise of Sol-In or Opl-In or the fall of the Sta-In.
8. Once Adr-Out has risen for the interface disconnect control sequence, it must not drop until Opl-In drops.
9. None of the out-lines, except Sup-Out, have meaning when Opl-Out is down.
10. Sel-Out can rise only if Opl-In and Sel-In are down.
11. Opl-In cannot fall until either:
 - a) Sel-Out falls and an Out-Tag response is sent for the last in-tag of any given signal sequence, or
 - b) Opl-Out falls, or
 - c) An interface disconnect sequence is given.
12. Opl-In cannot rise unless Opl-Out is up and must drop if Opl-Out drops.

4-2.3 PRIORITY ASSIGNMENT BY SEL-OUT ROUTING:

As mentioned earlier, the Sel-Out signal is generated in the channel and sent to the control unit placed physically first inline. The action within the

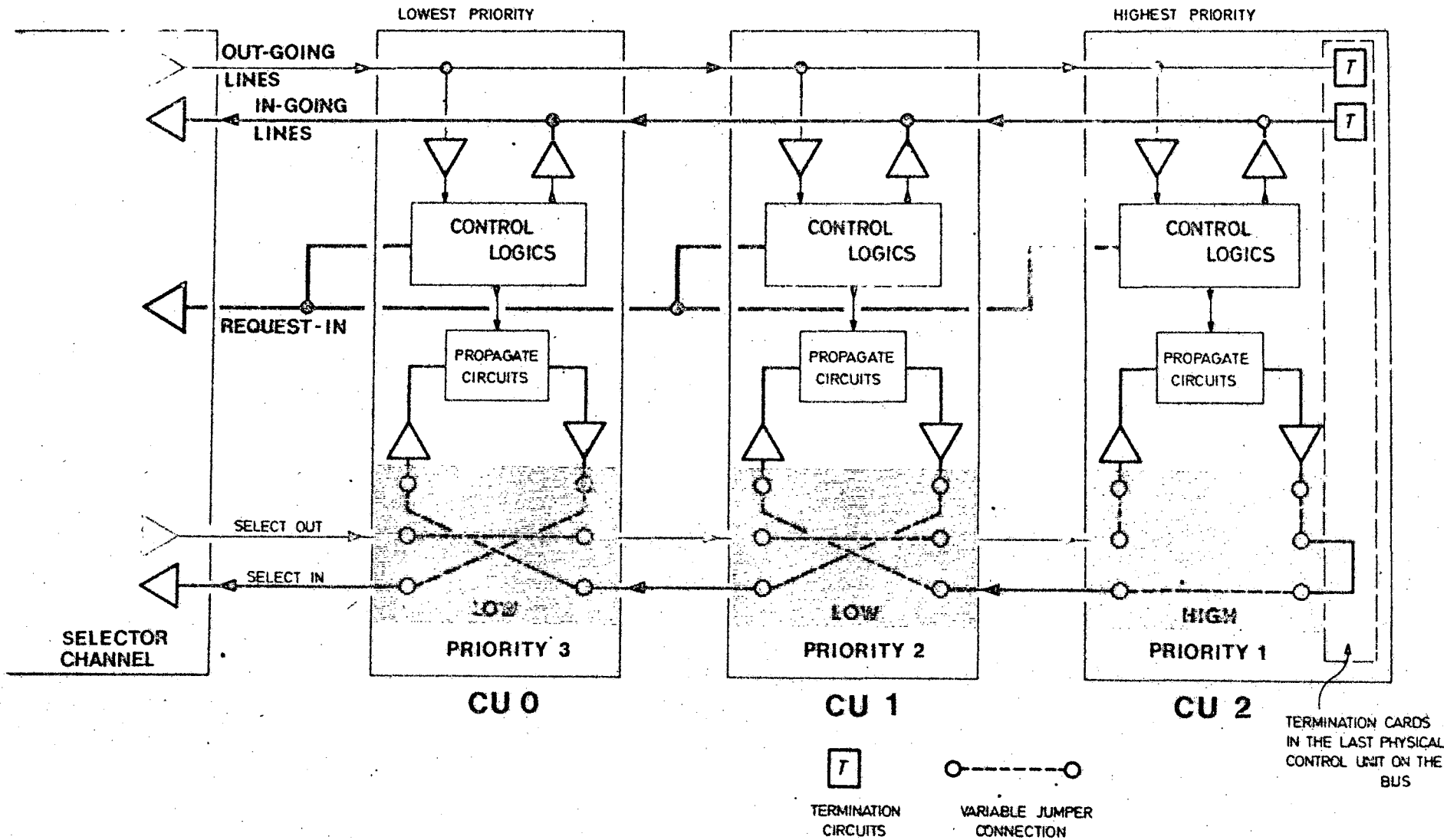


Fig 4.3 PRIORITY ASSIGNMENT BY SELECT-OUT ROUTING

control unit, on receipt of this signal, depends upon the configuration of its wired priority patch-card. Two configurations are possible, namely high or low priority.

In the Channel initiated selection sequence, the sel-out sel-in routing will indicate whether the device as specified by the device address on the Bus-Out, is operational or non-operational. Priority routing is not significant in this case.

However, in the control unit initiated sequence the above mentioned system enables priority to be exercised, if simultaneous requests occur. With low priority configuration, the Sel-Out signal is passed through the first control unit to the second, second to third and so on.

With high priority wiring, the Sel-Out signal is gated such that a device address comparison is sought between the device address patch card logic and the contents of the Bus-Out. This is true for a channel initiated sequence when the ADR-Out line is active. If there is no address compare on the first control unit, the sel-out signal is propagated (re-generated) to the second. If an address compare exists, the propagation of Sel-Out is blocked. As in the case of control-unit initiated sequence, the ADR-Out line is inactive, the control unit requests to transfer a status or data byte, cause the propagation of Sel-Out to be blocked.

If no address comparison, or any request for transfer, exists at any attached control unit, the signal Sel-Out is so propagated as to return to the channel as Sel-In as shown in the figure.

In the figure, three control units are shown connected to the interface cable, with control unit zero (CU0) closest to the channel. The route taken by the Sel-Out signal is also shown in the figure. The priority of control unit is determined by the priority patchcard configuration in each control unit. CU0 and CU1 are wired as low priority, whereas CU2 is wired as high priority. Therefore, Sel-Out is passed through CU0 and CU1 to reach the propagate ckt. of CU2 first. If no address compare exists at CU2, the Sel-Out signal is amplified and passed on i.e. propagated, to the next unit in priority. If address compare exist any further propagation of the signal is blocked.

4-2.4 IMPORTANT SIGNAL SEQUENCES:

As we have already discussed the general I/O sequence in Chapter 3, we will discuss here some of the important I/O sequences in detail pertinent to data transfer from channel to I/O devices (Output case) in Burst mode of Multiplexor channel. In this section the activation of the interfacing signals will be described step by step, relevant to a particular operation. The activities over the interface are shown with the help of block diagrams and Timing diagrams. The block diagrams show the logic activity within the interface area.

The timing diagrams do not attempt to portray signal occurrence or duration to any accuracy but merely to indicate their presence and position, in the manner of a conversation over the interface, during the operation. Time is shown on the horizontal axis.

4-2.4.1 CHANNEL INITIATED SELECTION SEQUENCE:

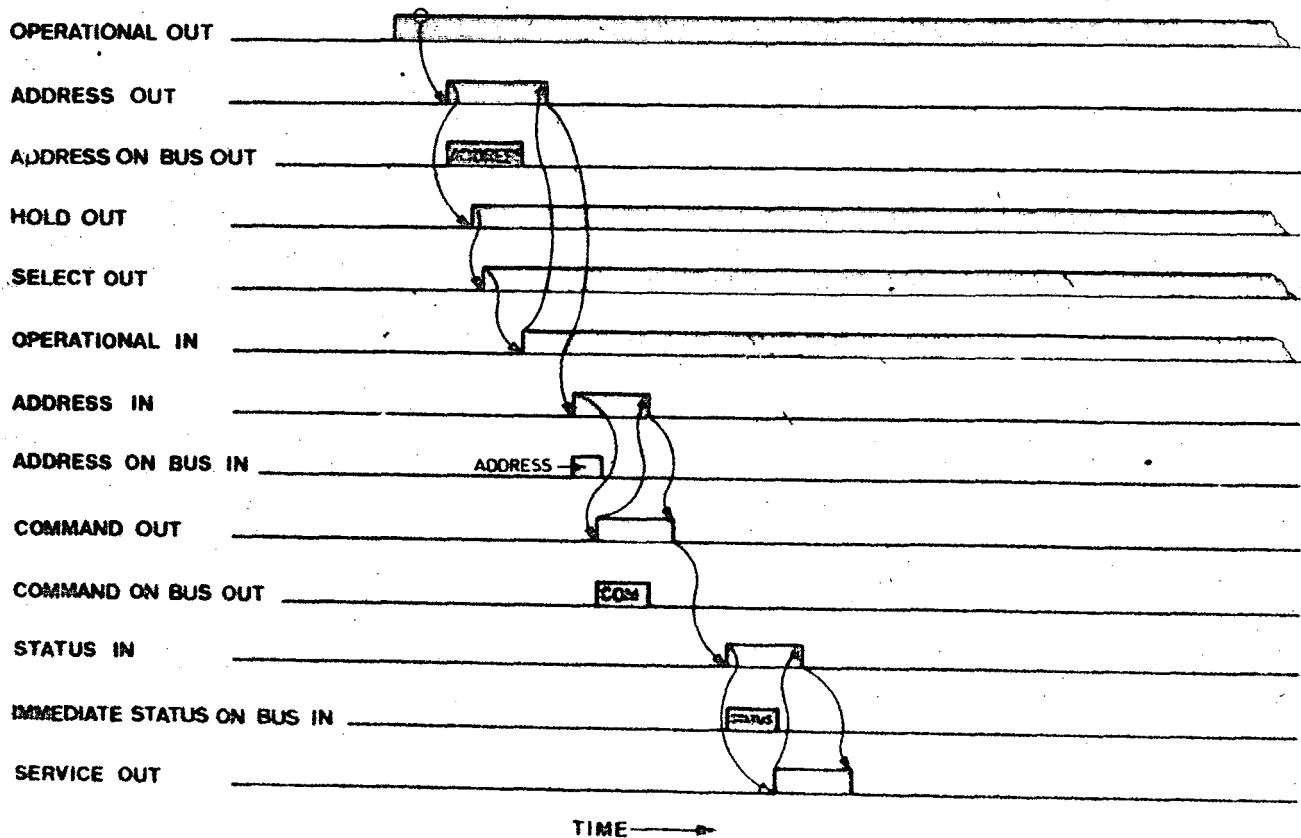
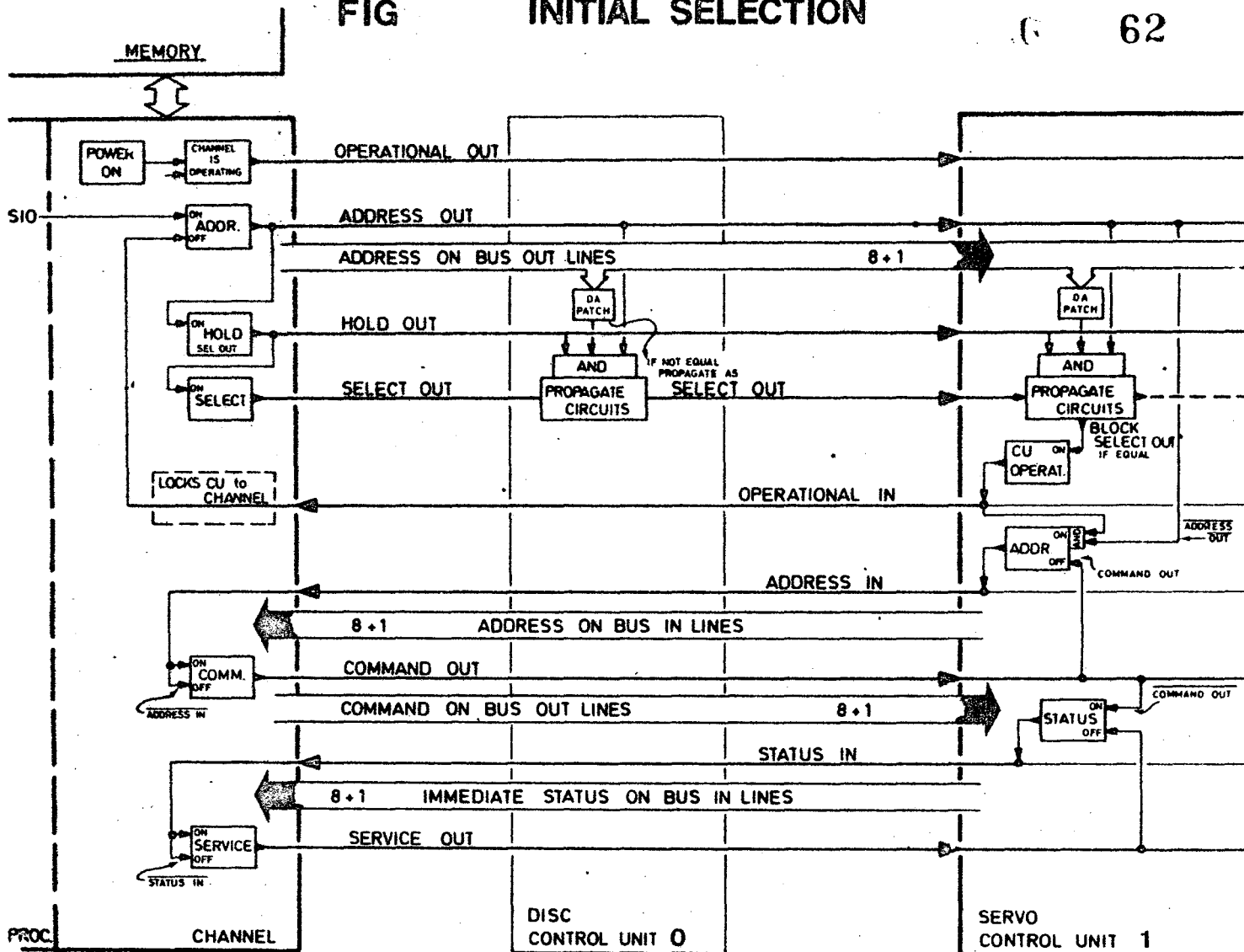
The objects of the operation, to be achieved by this interfacing sequence, are as follows:

1. select the device (control unit 1 and 'Module')
2. Establish its state of availability.
3. Issue a command to it.
4. Transfer and acknowledge its Initial Selection Status.

The channel initiates the sequence by delivering the device address, obtained from the start I/O instruction, to all the control units on the Bus-Out. The signal Adr-Out is raised to define the contents of the Bus-Out as address. To be acceptable, the address must have correct parity.

The channel then issues Sel-Out and when the incoming Sel-Out appears at the addressed control unit, the control unit blocks its propagation and raises the Opl-In line, which indicates to the channel that it has responded to its address, is operational and is not busy with anyother transaction. For simplicity, the block diagram shows only two control units and assumes that although CU0 has higher priority, it is not the one which is addressed. When Opl-In rises,

FIG INITIAL SELECTION



the channel responds by dropping Adr-Out. After Adv-Out falls, the control unit places its own address on the Bus-In and raises Adr-In. For multiplexor channel, Hld-Out with Sel-Out may drop any time after this point.

At this point, the channel checks for the identification address sent by the control unit and on being correct address, the channel command word is fetched from memory in order to obtain the command code which is then placed on the Bus-Out to the selected control unit. The Bus-Out is defined by the Command-Out line. After the command byte is loaded into the register of the control unit, it is examined for validity and possible violation in relation to the addressed module and, based upon the result, the appropriate status byte is prepared. If the command is invalid, the status of unit check (OE_{16}) is returned. If the module, selected by the control unit from the device address, found to be busy performing a previously initiated operation, a 'Busy-Bit' alone is sent in the status byte (IO_{16}) and returned to the channel. Alternatively, any status conditions which may exist (pending) for the ixx selected module is reported by means of the status byte at this time. If the command is accepted, the status returned depends upon the type of command which was issued. Certain commands, not involving data transfer, may be executed almost immediately. Therefore, the status

byte returned at this time may contain command ending status. If the command involves data transfer a zero (00₁₆) status will be returned.

The channel acknowledges the receipt of the status byte by raising the signal *Srv-Out* and this signifies the end of the initial selection sequence.

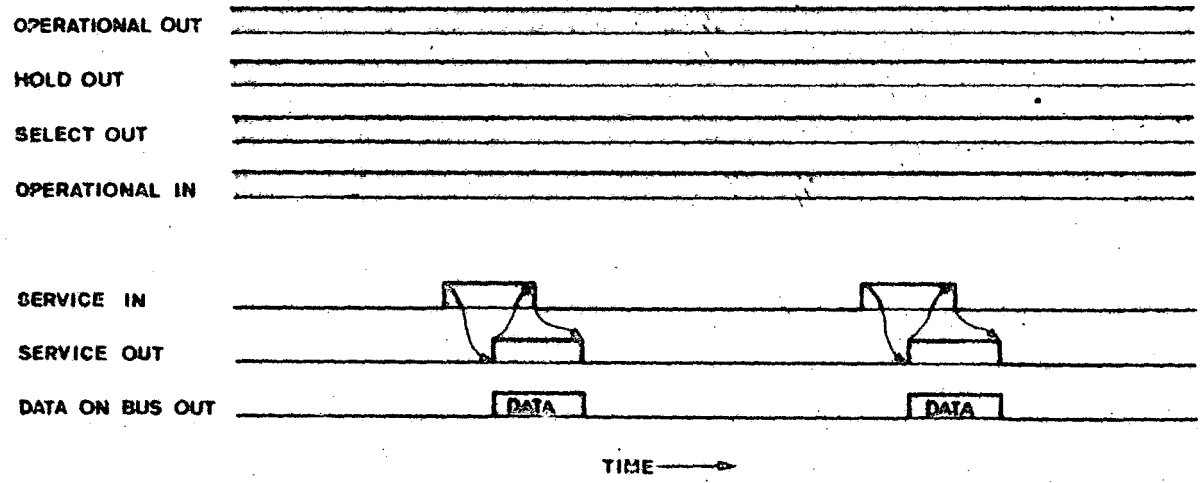
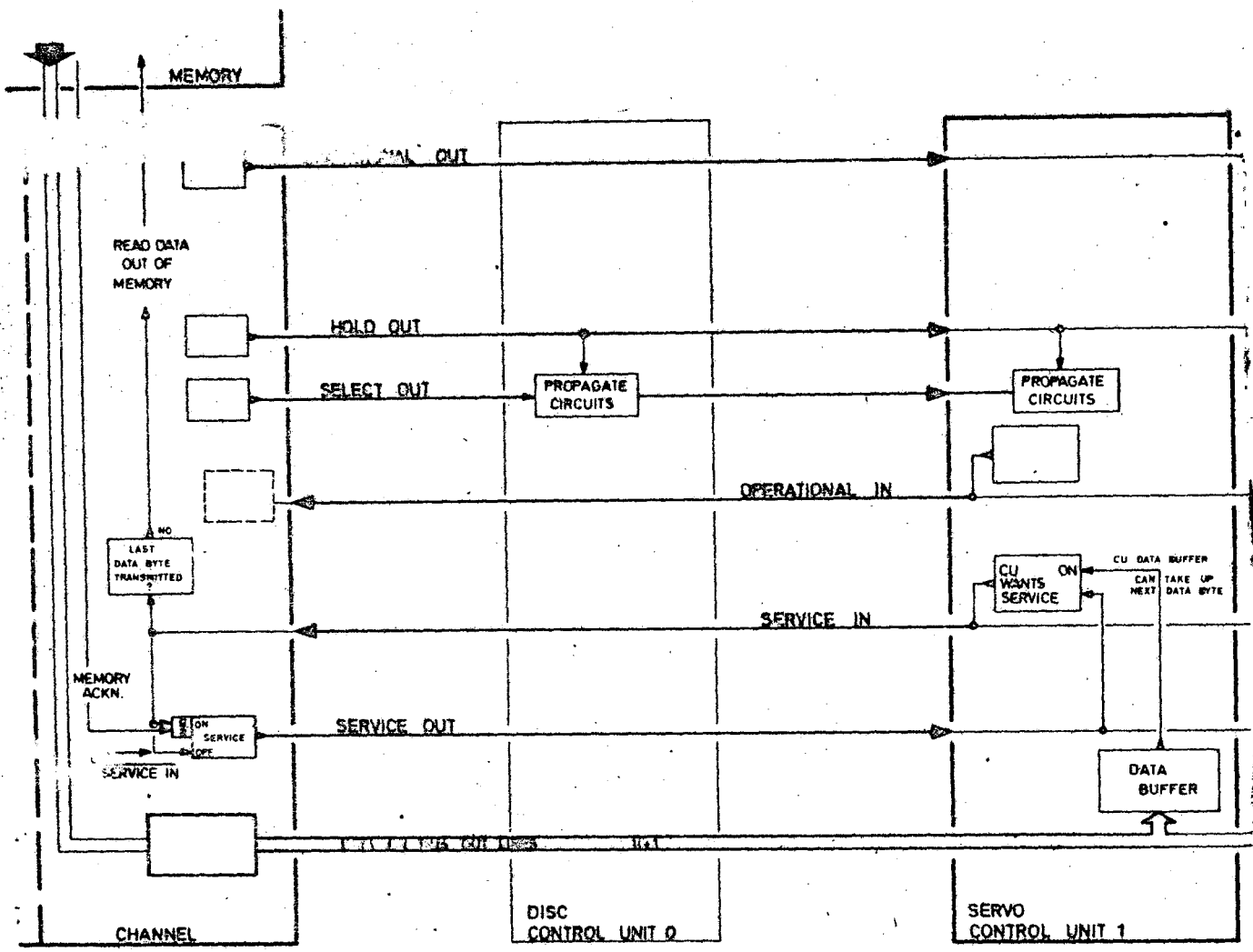
The selected control unit is now free to enter its next phase, which is to execute the loaded command. In order to hold on to the interface for the execution phase, in accordance with the burst mode of operation, the lines *Opl-Out* and *Opl-In* must remain active. An output data transfer if specified by the loaded command, will commence with the first request, which is made by the activation of the *Srv-In* line.

4-2.4.3 WRITE (OUTPUT) DATA TRANSFER SEQUENCE - BURST MODE:

After completion of the Initial Selection Sequence the interface lines *Opl-Out* and *Opl-In* remain active, in accordance with the burst mode of operation. This means that the interface is secured for that device. For example, control unit 1 retains the control of the interface after a satisfactory initial selection. This allows it to proceed with the execution of the loaded command.

If the command belongs to 'Write-Group' of commands, its execution commences when a request is made by the control unit for the first output byte. The request is accomplished by activating the *Srv-In* line, to which channel responds by placing the output byte on the *Bus-Out*, defining it by activation of the

FIG.4.5 WRITE (O/P) DATA TRANSFER



Srv-Out line. Srv-Out can be regarded as the positive response to Srv-In and will be repeated for every byte that is placed on the Bus-Out, required by the command.

The number of bytes transferred across the interface depends upon the values set into two byte counters, one whose value is taken from the CCW and contained within the channel, and the other given by a byte counter in the control unit. Both will be decremented with each byte transferred.

The byte counter in the control unit will have decremented by an amount n to zero, when n bytes have been transferred. At this point the command ending status, given by the channel-end and Device-end bits, is generated and presented on the Bus-In to the channel. The activation of the Sta-In line defines this.

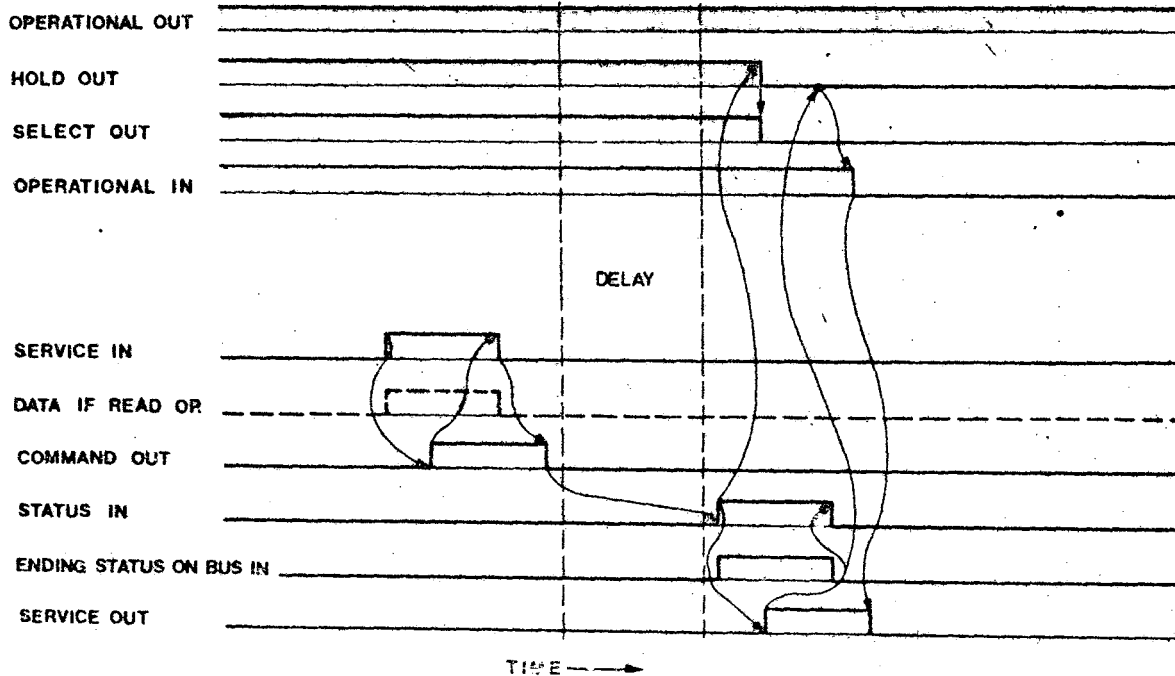
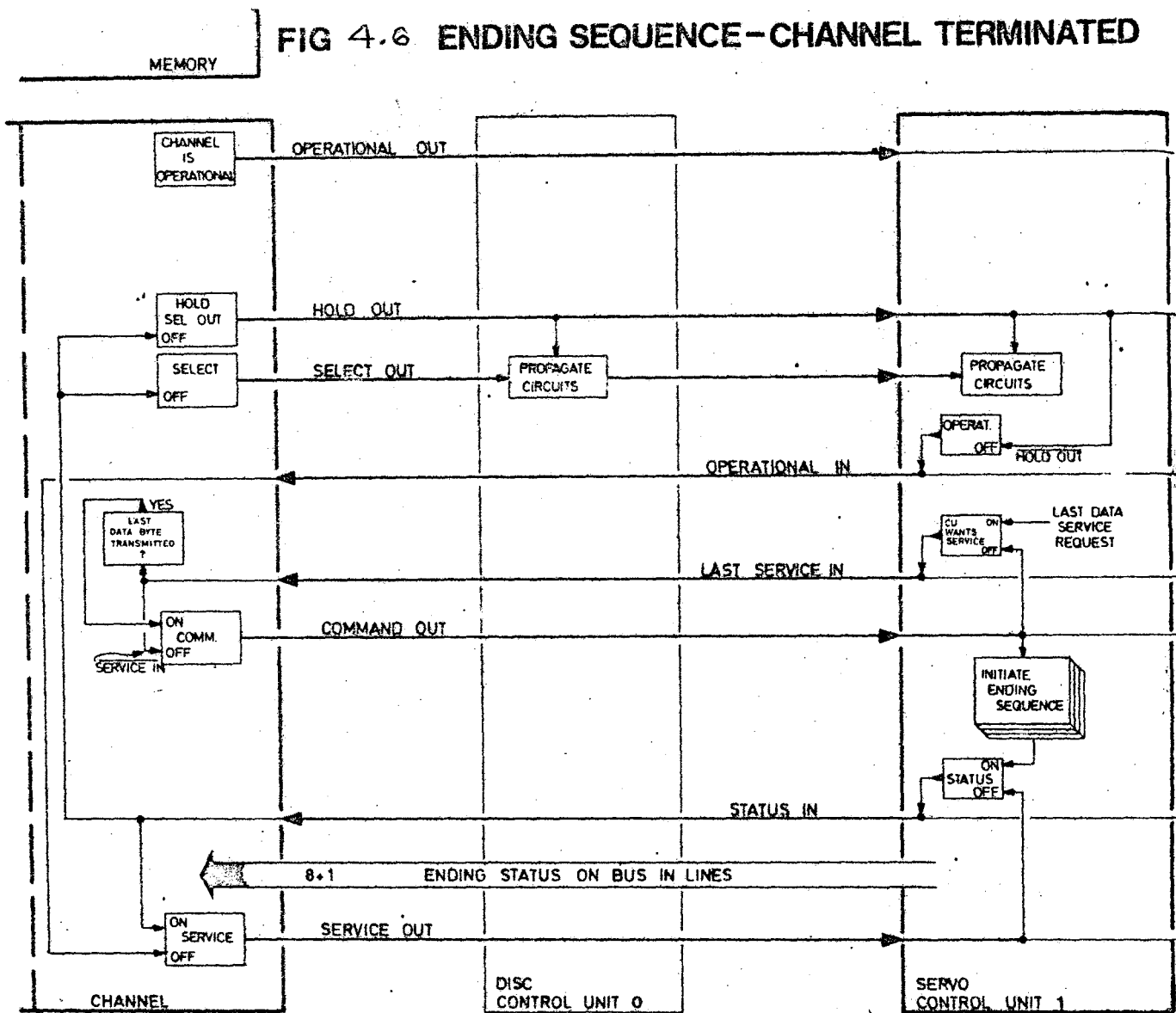
After the status acknowledgement, by the raising of Srv-Out, the channel deactivates the Hld-Out and Sel-Out lines, which results in the dropping of Opl-In and the release of the interface; if command chaining is not specified.

The above sequence is one which allows the transfer of one block of output data over the interface from the channel, to one device only, before termination. A further start I/O instruction must be encountered in memory, in order to transfer a further block to this device.

4-2.4.3 ENDING SEQUENCE DATA TRANSFER TERMINATED BY THE CHANNEL

The ending procedure may be initiated by either the I/O unit or the channel. If the procedure is initiated by

FIG 4.6 ENDING SEQUENCE - CHANNEL TERMINATED



the I/O unit, the end of operation is completed in one signal sequence, assuming that both channel-end and device-end status conditions occur together. If the procedure is initiated by the channel the I/O device may still require time to reach the point where the proper status information is available, in which case a second signal is necessary to complete the ending procedure.

One of three situations may exist at the initiations of the ending procedure:

1. The channel and the I/O unit recognize the end of an operation simultaneously.
2. The I/O unit recognizes the end of an operation before the channel reaches the end of an operation.

For these situations status information is available at the control unit. The control unit places the ending status on Bus-In and raises Sts-In.

3. The channel recognizes the end of an operation before the I/O device reaches its ending point. In this situation whenever the control unit requires service again, it raises the Srv-In line, the channel responds with Cnd-Out indicating STOP. The control unit drops Srv-In and proceeds to its normal ending point without requesting further service.

when the I/O device reaches the point where it normally would send 'channel end', the CU places the ending status on Bus-In and raises sta-in. The channel responds with Srv-Out.

4-3 GT-150 INCREMENTAL PLOTTER:

GT-150 Graphic Terminal is an all-solid state incremental plotter. It is meant for rapid translation of digital computer outputs into computer graphics for quick visual interpretation and analysis. This plotter can be used for on-line operation with a computer for computer graphics. It incorporates the following features:

1. Designed with Integrated circuits.
2. Specially designed Joy-Stick for quick manual positioning of the pen anywhere in the plotting area.
3. Limit switches in all the four directions to provide total safety for the unit.

4-3.1 FUNCTION:

This graphic terminal model GT150 is a medium speed drum type plotter. The plot is drawn along two axes by moving a pen on the X-axis and moving the paper on the Y-axis. The movement of the pen w.r.to the paper or the paper w.r.t. the pen is carried out in small discrete steps or increments. The plotter can be used for producing most forms of graphical display driven from digital input signals. It is designed for rapid translation of digital output into computer graphics for

quick visual interpretation and analysis. A general view of the complete plotter assembly is shown in fig given on the next page.

4-3.2 OPERATING PRINCIPLES:

4-3.2.1 THE PLOTTING SYSTEM:

The plotter logic forms the collecting centre for incoming signals:

- i) From the main control lines via logic level converts.
- ii) From the manual controls.
- iii) From the various plotter interlocks.
- iv) From the internal clock generator, used to produce pulse trains for the movement in manual condition.

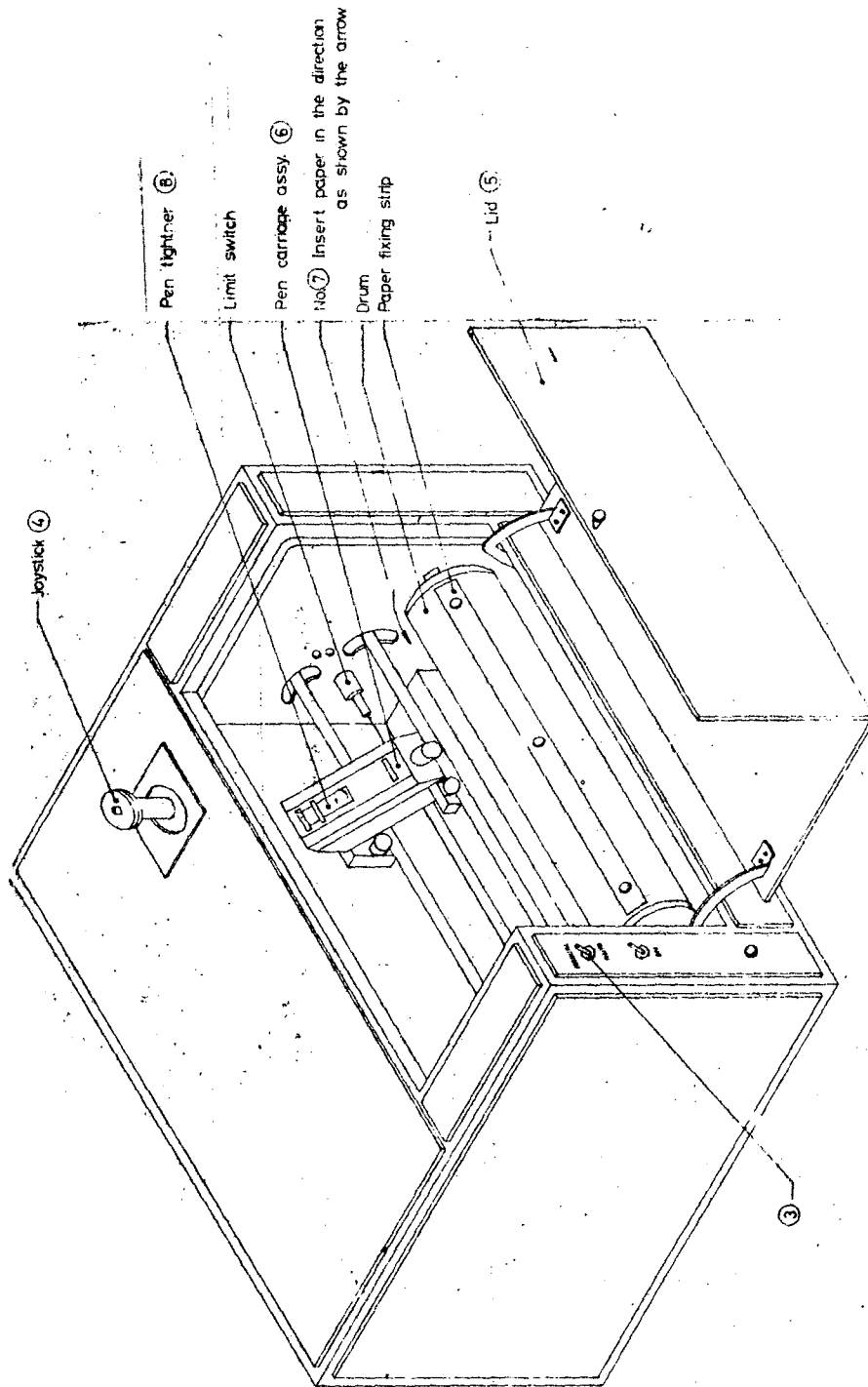
There are three output channels from the plotter logic.

- i) to the X-axis drive module
- ii) to the Y-axis drive module
- iii) to the pen solenoid amplifier.

The modules and amplifier accept signals from the above and convert these into signals of sufficient power to operate the stepper motors or pen solenoid.

4-3.2.2 X-AXIS:

One pulse on one of the two input lines relating to the X-axis (+ X or - X) will via the plotter logic and driver module, causes the X-axis stepper motor to rotate clockwise or anticlockwise by one step. The movement is transferred to the pen carriage on which the pen is mounted. by a steel drive wire and pulleys. Thus the



pen is caused to move one step to the right or left.

4-3.2.3 Y-AXIS:

One pulse on one of the two input lines relating to the Y-axis (+Y or -Y) will, via the plotter logic and driver module, cause the Y-axis stepper motor to rotate clockwise or anticlockwise by one step. This movement is transferred to the plotter drum by pulleys and steel cable.

Thus the drum is caused to advance or retard by one step.

4-3.2.4 PEN CONTROL

A pulse via plotter logic and amplifier on the pen input control lines, will cause the pen solenoid to be energised or de-energised. Due to its own weight the pen falls on to the paper when the solenoid is energised. Due to the action of a spring it is caused to rise up when the solenoid is de-energised.

4-3.2.5 MANUAL CONTROL

All the above movements are possible by manual operation also. They are obtained by feeding a pulse train into the X-or Y motor circuits. The appropriate direction of movement and speed is selected by use of a manual Joy-stick. It is also possible to operate the pen up and down by the Joy-stick.

4-3.3 GENERAL SPECIFICATIONS:

1. Plotting dimensions

Plotting length : X-axis - 275 mm

Y-axis - 210 mm

Paper size : 286 mm x 222 mm

2. Speed and increment: Smallest increment of 0.2 mm at a maxm. rate of 150 steps/sec.
3. Protections : Movements in X- and Y- directions are stopped at the extranities of the plotting area by 8 limit switches.
4. Additional facility: Easily Interchangeable per assembly to obtain graphics in different colours and sizes varying from 0.2mm to 0.8mm in steps of 0.1mm

4-3.4 INPUT REQUIREMENT:

1. Input signal : Operates with TTL compotible signal with positive logic '0' level is 0.8V max and '1' level is 2V minicum. Max. pulse rate acceptable is 150 Hz.
2. Input Code :
- | | |
|--------|-----------------------------|
| 000001 | +X (Right movement for pen) |
| 000010 | +Y (Up movement for drum) |
| 000100 | -X (Left movement for pen) |
| 001000 | -Y (Down movement for drum) |
| 010000 | Pen (up) |
| 100000 | Pen (down) |
3. Power Requirements : 230V AC, 150Watts 50 C/S.
4. Limit Switches : X-axis limit switches are placed at the extenities of the pen travel. Y-axis limit switches

are placed at the extremities of the drum travel.

4-3.5 OPERATIONAL CHECKOUT AND OPERATING PROCEDURE:

(A) Operational Checkout:

(1) Paper loading: Can be done with the m/c ON and it is not necessary to remove the pen. Before inserting the paper the drum has to be positioned by actuating the JOY-STICK, such that the paper fixing strip comes to the top. Lid should be opened and paper fixing strip should be lifted up. Paper should be wrapped around the drum and its two ends should be pressed under the fixing strip by pushing it down.

(2) Pen Loading: Pen tightener should be taken out from the pen carriage. The pen is inserted in the pen holder and tightener is inserted over the pen. Then lid of the plotter is closed. Joy-stick should be operated in order to ensure that the 'PEN UP' and 'PEN DOWN' functions properly and that the pen nib is touching the paper when the pen is down and not touching when the pen is up.

(3) Put the power ON if not ON.

(4) Put the Auto/Manual switch in the manual position.

(5) Move the JOY-STICK to front and rear of the plotter.

(6) Move the Joy stick left and right and see that the pen is moving appx. at 1 step/ sec.

- (7) Depress the Joy-Stick and repeat the above procedure and see that the movements now take place appx. at 180 steps/sec.
- (8) Press the centre knob on the Joy-stick and note that the pen drops into contact with the plotting paper. Again press the knob and see that the pen returns up. A mark having appx. size of the pen stylus should be left on the plotting paper.
- (9) Drop the pen on the plotting paper. Operate the Joy-Stick in all directions possible, with and without depressing the knob, to produce a trace along each axis in turn, noting that a satisfactory trace is produced of consistent width and density. After that return the pen to the up position.
- (10) Put the auto/manual switch in auto position and non- of the above functions will be working when operated manually by the Joy-stick. The function of the plotter in the Auto mode is wholly dependent on the data input on its six lines.

(B) Operating Procedure:

After performing the operational checkout before the start of a plot, the pen should be positioned at the starting position. Then ready to start the automatic plotting (i.e.) just before executing the user plot program, the auto/manual switch should be

placed at auto position.

On completion of the automatic plotting, the plotter should be switched back to the Manual position. Completed plots can be removed after opening the lid and then lifting up the paper fixing strip. Then pen is unloaded and lid is closed.

4.4 BUFFERED PLOTTER CONTROLLER:

The incremental plotter is regarded as one of the slowest peripherals in a computer system. If this device is hooked to the system in multiplex mode of operation on the multiplexor channel without buffering facility, the data transfer will be at a very slow rate and CPU will be highly burdened for the data transfer to this device. This interface is designed in such a way that the speed of the device becomes invisible to the channel. The major objective of the design is to have a maximum possible throughput efficiency in the multiprogramming mode. This completely buffered interface supports multiplexor channel operation in the burst mode and allows the particular memory partition (storing the plot program output) to be released within a comparatively insignificant duration.

This controller can be divided into three logical portions:

- (i) Interface controller
- (ii) Buffering
- (iii) Plot signal generator.

- (i) Interface controller meets the requirement of standard protocol followed by the channel when dealing with the devices associated with it. It also determines the priority of the device on the particular channel. It receives the commands from the channel and takes actions on these commands. Also it sends back reply to the channel when needed. When command involves data transfer i.e. WRITE, it receives data from the channel and generates necessary write pulse to write into the buffer until it becomes completely full. In that condition, it informs the channel that transfer of data should be stopped and device is busy, executing the orders (i.e. taking action on data). Data transfer is also stopped when channel has no more data to transfer. It sends all the information about the status of the device as and when required by the Channel.
- (ii) Buffering is provided in order to make the speed of the device invisible to the channel. Channel assumes that it is transferring data to a high speed device. The size of the buffer can^{be} chosen an optimum one by consideration different factors involved in computer graphics. For the time being a buffer size of 256 bytes has been provide which is an optimum size for almost^{all} practical purposes.
- (iii) From the buffer, plotter driver fetches two consecutive bytes and interpretes different bits of the two bytes. The low order two bits of the first byte contains information regarding the

position and the remaining six bits as X-count data. Similarly, the two low order bits indicate the information about the pen and paper movement and the remaining six bits shows the Y-count data. All these informations are used to send required number of pulse trains on the specified lines of the plotter i.e. (+X, +Y, -X, -Y, Penup, pendown)

4-6. INTERFACING REQUIREMENTS:

For the interfacing of any device to a computer system, the interface logic should cater to the needs of both the computer and the device. Device specification tells about certain information formats to be provided in order to perform the desired function which the device is supposed to do. Similarly, the computer has been designed in such a way that some standard protocol has to be followed by the device in order to hook it to that system. Thus the interface logic should communicate to the CPU (or channel) and gather the informations to be sent to the device. After reformatting it (which is acceptable to the device), it sends it to the device to perform the desired function. Thus before an interface is designed one has to collect two types of informations:

- (i) Information formats supplied by the channel
- (ii) Information format acceptable to the device.

4-6.1 INTERFACING REQUIREMENT -INCREMENTAL PLOTTER:

The incremental plotter accepts inputs on six lines namely +X, -X, +Y, -Y, Pen Up, and Pen Down. The signals (pulse trains) on these lines can be TTL

signals i.e. positive logic '0' - 0 to 0.8V

'1' - +2V to +5V

The maximum pulse rate acceptable is 150 Hz. At a time any one of the lines PENUP AND PEN DOWN should be high. Similarly only one line from each +X & -X and +Y & -Y should be high at a time. The controller should cater to this need of the plotter for smooth and error-free operation.

4-5.2 INTERFACING REQUIREMENTS - EQ 1020B MUX CHANNEL:

The general requirements for the interlocked sequences needed by the mux channel has been discussed in the signal sequences in the beginning of this chapter. In addition to that the electrical level requirements are as follows:

Signal Level - TTL Compatible positive logic

'0' - between 0 to 0.8 V.

'1' - between + 2V to +5 V.

Signal rise time - In the order of 100 ns.

Shielded coaxial cables for connection between bus and the controller.

40 pin connectors - 2 nos.

The signal lines coming to the connectors are shown in the diagram given on the next page. Connector 1 incorporates Data Bus lines - BUS-OUT and BUS-IN, while the connector 2 incorporates the control and Tag lines.

CONNECTOR 1

	A				A				
	4	3	2	1	4	3	2	1	
1	Shield	BusOutP	Shield	BusOut0	1
2	BusOut1	Shield	BusOut2	Shield	2
3	Shield	BusOut3	Shield	BusOut4	3
4	BusOut5	Shield	BusOut6	Shield	4
5	Shield	BusOut7	5
6	BusInP	Shield	BusIn0	Shield	6
7	Shield	BusIn1	Shield	BusIn2	7
8	BusIn3	Shield	BusIn4	Shield	8
9	Shield	BusIn5	Shield	BusIn6	9
10	BusIn7	Shield	10

CONNECTOR 2

	A				A				
	4	3	2	1	4	3	2	1	
1	Shield	Op1Out	Shield	SupOut	1
2	Shield	Shield	Op1In	Shield	2
3	Shield	ReqIn	Shield	SelOut	3
4	Sel In	Shield	4
5	5
6	AdrOut	Shield	CmdOut	Shield	6
7	Shield	SrvOut	Shield	AdrIn	7
8	Sta In	Shield	SrvIn	Shield	8
9	Shield	MtrOut	Shield	MtrIn	9
10	CleOut	Shield	10

CHAPTER - 5INTERFACE DESIGN DESCRIPTION

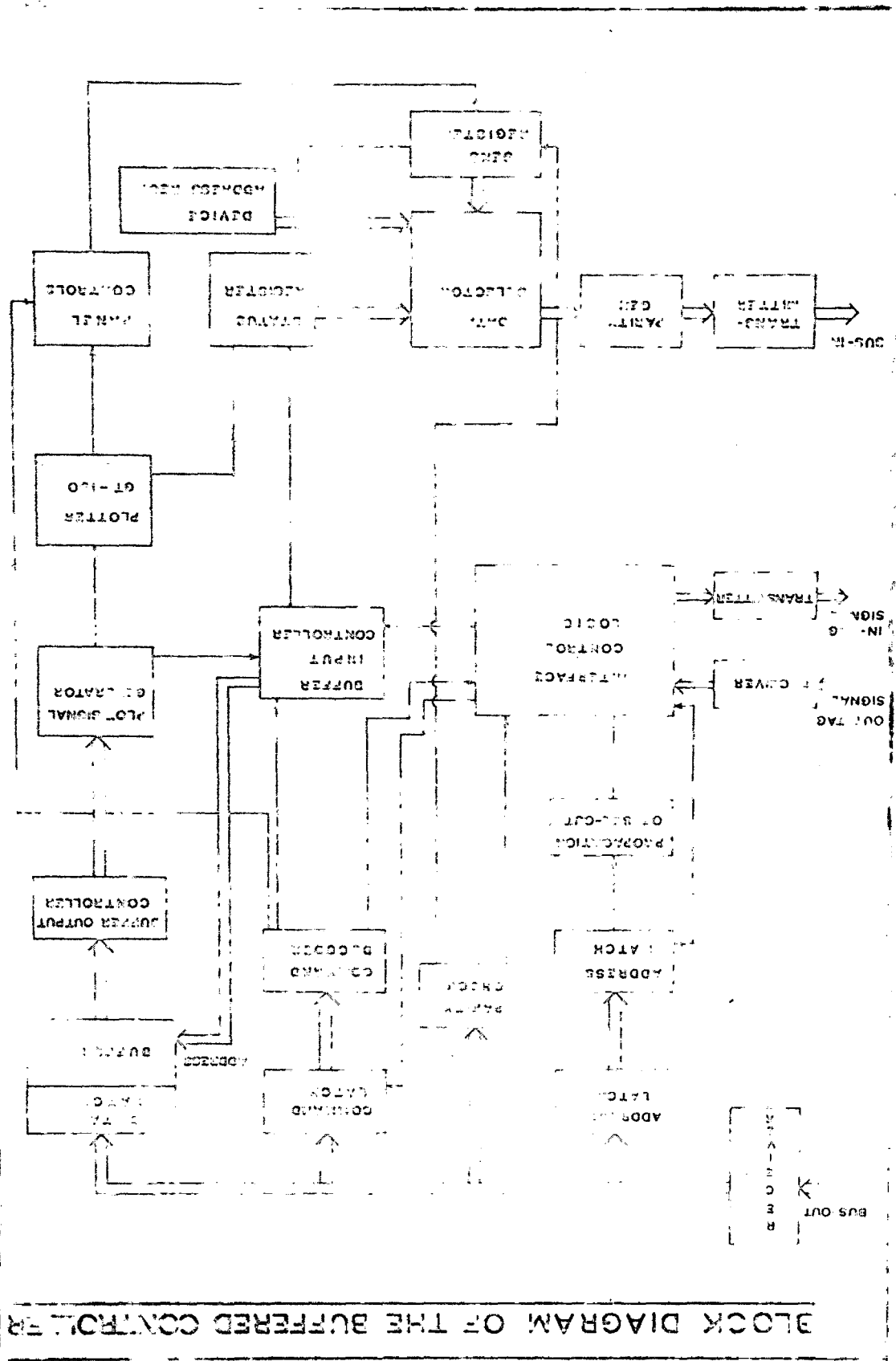
In this chapter a detailed description of the designed interface has been given. With the help of block diagram, the different components of the interface has been described. Then the description about the logic design and chip level implementation has been provided in the subsequent pages of this chapter. For the easy reference of the maintenance Engineer, the physical position of the chips are indicated in the logical description of the various components of the Interface.

5-1 BLOCK DIAGRAM OF THE BUFFERED CONTROLLER:

The schematic block diagram of the buffered controller is shown on the next page. The input lines coming to this controller are: BUS-OUT and OUT-TAG signals. The output from this controller are going to two places:

- (i) to the channel as BUS-IN and IN-TAG signals.
- (ii) to the plotter as +X, -X, +Y, -Y, PENUP, PENDOWN signals.

The BUS-OUT and OUT-TAG signals are received in the RECEIVER and are passed to the appropriate places in the ckt. BUS-OUT signals appear simultaneously to four blocks - ADDRESS LATCH, PARITY CHECK, COMMAND LATCH, and DATA LATCH. OUT-TAG signals appear to the INTERFACE CONTROL LOGIC which indicate the nature of information on the BUS-OUT. So depending on the situation in the INTERFACE CONTROL LOGIC the content on the BUS-OUT are latched to



either ADDRESS MATCH, COMMAND DECODER, OR BUFFER. PARITY CHECK ckt. performs parity checking in all the states and informs the INTERFACE CONTROL LOGIC about the parity error. If parity error exists, the SENSE Register is set to indicate this situation and also the status register is appropriately set.

In the case of O.K. parity, the ADDRESS MATCH & PROPAGATION OF SEL-OUT ckt. blocks the propagation of SEL-OUT signal when the address match takes place. Otherwise, the SEL-OUT signal is propagated to the other device. When the address match takes place and the SEL-OUT signal gets blocked, the INTERFACE CONTROL LOGIC transmits appropriate IN-TAG signals and routes the DEVICE ADDRESS REG to the BUS-IN VIA DATA SELECTOR ckt. for the confirmation of the selection of the device. The channel then sends commands which are received and decoded in the COMMAND DECODER. In the case of WRITE command, the BUFFER INPUT CONTROLLER is informed of the state. Then INTERFACE CONTROL LOGIC sends requests to the channel for data transfer. When data is placed on the BUS-OUT the INTERFACE CONTROL LOGIC informs the BUFFER INPUT controller of the same and it stores this data byte at the current address. Then it increments its address by 1. This way for each data byte transfer, the same action is repeated till the buffer becomes full. At this moment, the busy status is presented to the channel by routing STATUS REGISTER VIA DATA SELECTOR TO BUS-IN. Then BUFFER OUTPUT CONTROLLER fetches two bytes of data from the buffer and sends appropriate signals to the plotter.

6-2 DIVISION OF THE INTERFACE LOGIC:

The electronic circuitry of this interface, which employs latest electronic techniques, can be divided into following functional parts:

- (a) Selection block.
- (b) Error check on bus-out.
- (c) Accepting the command code, data and address.
- (d) Generation of Reset Signal.
- (e) Generation of In-Tag signals.
- (f) Directing the Device Address, Status, and sense register to BUC-IN.
- (g) Setting up the Status and Sense bits.
- (h) Buffer address generation and READ/WRITE control.
- (i) 256 Bytes Buffer.
- (j) Reading the Buffer and Plot Signal generation.

6-2.1 INTERFACE STATES:

The sequence of operation to be executed by the plotter interface may be broadly classified into three states:

- a) Command State.
- b) Data State.
- c) Ending Sequence.

The plotter is of incremental type and can accept only digital signals serially. The device control does the plotting operation using the buffer contents and so the device is practically invisible to the channel (excepting the cases when unusual conditions occur and the plotter gives unit check or equipment check, in such cases status byte is sent and program gets cancelled).

So interface dialogue depends on buffer conditions mainly. The whole idea of buffered device is to minimize the loss of CPU time specially in multiprogramming environment. The design assumes this necessity. The order and data are stored in the buffer. An example of a normal sequence is :

```

FEN-UP  -- WRITE(I X Y)  -- XDATA  -- YDATA  -- PENDN
(order)  (order)         (data)   (data)   (order)

```

(continuous plotting facility is not considered at the moment)

Device control goes on executing this sequence from the buffer. Each time (i.e. for a command issue) it checks up the state of the device through a short initial selection sequence.

Whenever it is a WRITE command, the buffer control expects a number of data points (each of two bytes) from the channel so that the buffer gets completely filled. Channel can terminate the data transfer even if the buffer is not full, when the byte count in the CCW is over, which is indicated by the ending sequence. The low order 2 bits of the first byte of the data point contains pen position order and the high order six bits of this bytes contains X-COUNT. Similarly, the low order two bits of the second byte of the data point contains MOVEMENT DIRECTION Order. While remaining six bits indicate Y-count. So the software must cater to this requirement by forming the blocks of data. They must be sent sequentially.

The BUFFER OUTPUT CONTROLLER ckt. loads two presettable down counters which generates (rather allow preparation of the free running clock) the necessary number of impulses for X and Y directions.

5-2.2 BUFFER:

256 bytes size.

5-2.3 COMMAND STRUCTURE:

The different commands which the device controller may expect from the CPU are:

- | | |
|-----------|-----------|
| a) WRITE | 01 (Hexa) |
| b) TIO | 00 (Hexa) |
| c)SENSE | 04 (Hexa) |
| d) REMIND | 07 (Hexa) |
| e) NO OP | 03 (Hexa) |

a) WRITE - Transfers 256 bytes as a burst

b) TIO - Causes the device to send status information to channel. If during the execution of this command there is no stored information - zero status byte is sent.

c)SENSE - Causes device to send SENSE byte.

d) REMIND - Used to remind the operator to put the device in ready state for the execution of command (WRITE).

e)NO OP - Software facility only - no execution (the device sends CHE and DVE).

5.2.4 SENSE BYTE AND STATUS BYTE:**STATUS BYTE**

- Bit 3 - Busy
- Bit 4 - CH.END
- Bit 5 - DEV. END
- Bit 6 - UNIT CHECK
- Bit 7 - UNIT EXCEPTION

SENSE BYTE

- Bit 0 - ILLEGAL COMMAND
- Bit 1 - NOT READY
- Bit 2 - BUS-OUT PARITY ERROR
- Bit 3 - EQUIPMENT CHECK

5-2.5 IN-TAG SIGNAL:

Signals to be generated from interface side to channel are: ADR-IN, OPL-IN, STA-IN, SRV-IN.

5-2.6 DEVICE ADDRESS AND PRIORITY:

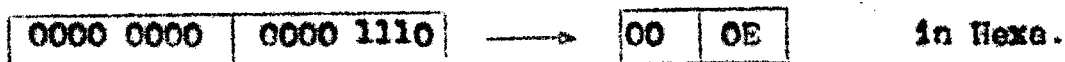
Before a command can be sent to the control unit, the unit must be addressed. Each I/O unit is designated by an I/O address. The I/O address consists of two parts; a Channel address (high order byte) and a Unit address (low order byte).



The channel address specifies the channel to which the instruction applies. In system / EG 1020, maxm. 3 channels are there:

<u>Binary Value</u>	<u>Hexadecimal Value</u>	
0000 0000	00	- Multiplexor channel
0000 0001	01	- 1 Selector channel
0000 0002	02	- 2 Selector channel

The unit address identifies the particular I/O unit in the channel. Any number in the range 0-255 can be used as a device address providing facility for addressing 256 units per channel. Unit addresses from 00¹⁶ to 7F¹⁶ pertain to non-shared sub-channel. Unit address from 80¹⁶ to FF¹⁶ pertains to shared subchannel. In this design, the unit has been placed on multiplexor channel and non-shared subchannel. Thus its address has been fixed as :



The I/O instruction contains the full address of the device i.e. both bytes. The first byte is removed from the address when it is presented to the device controller on Bus-out.

This device has been given the lowest priority on the Mux. Channel.

5-2.7 MAIN BLOCKS IN INTERFACE CONTROL:

- a) Section Block: As this device is connected to the mux-channel, provision should be made to give electrical path to the SEL-OUT signal even if this device is not operational (i.e. switched off). When this device is powered on, the SEL-OUT signal should pass through the Device selection circuitry. The general scheme is shown below:

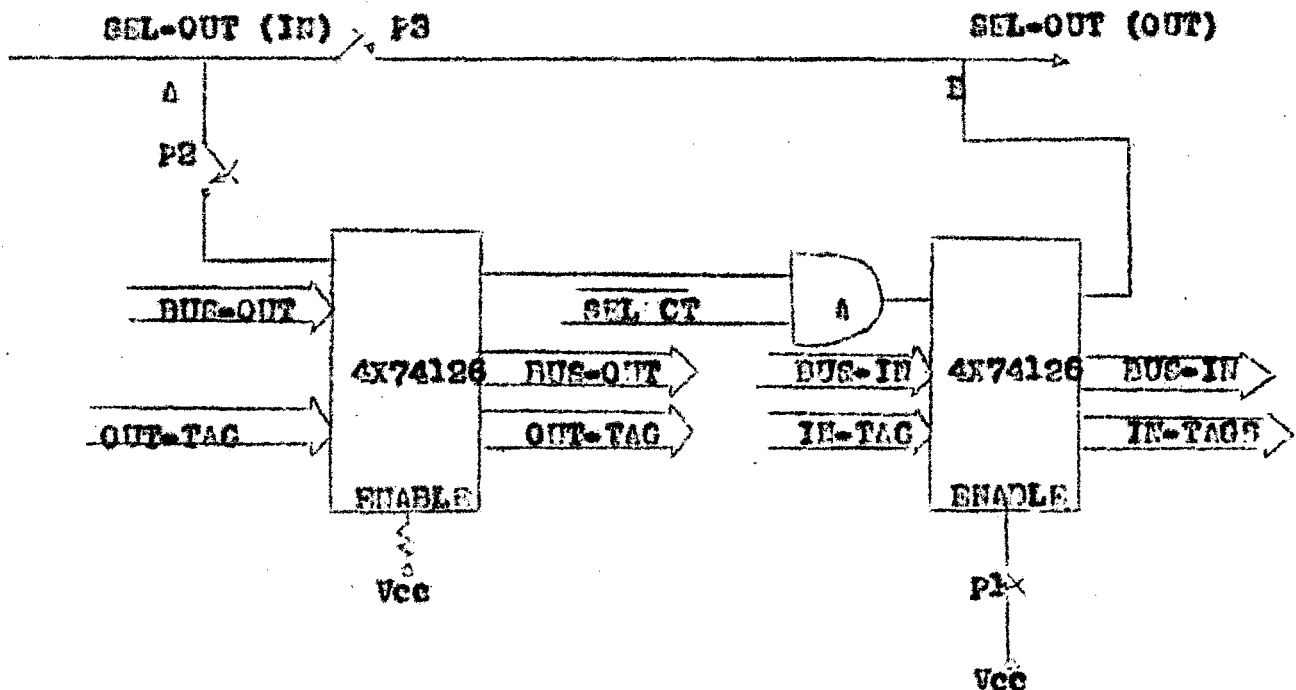
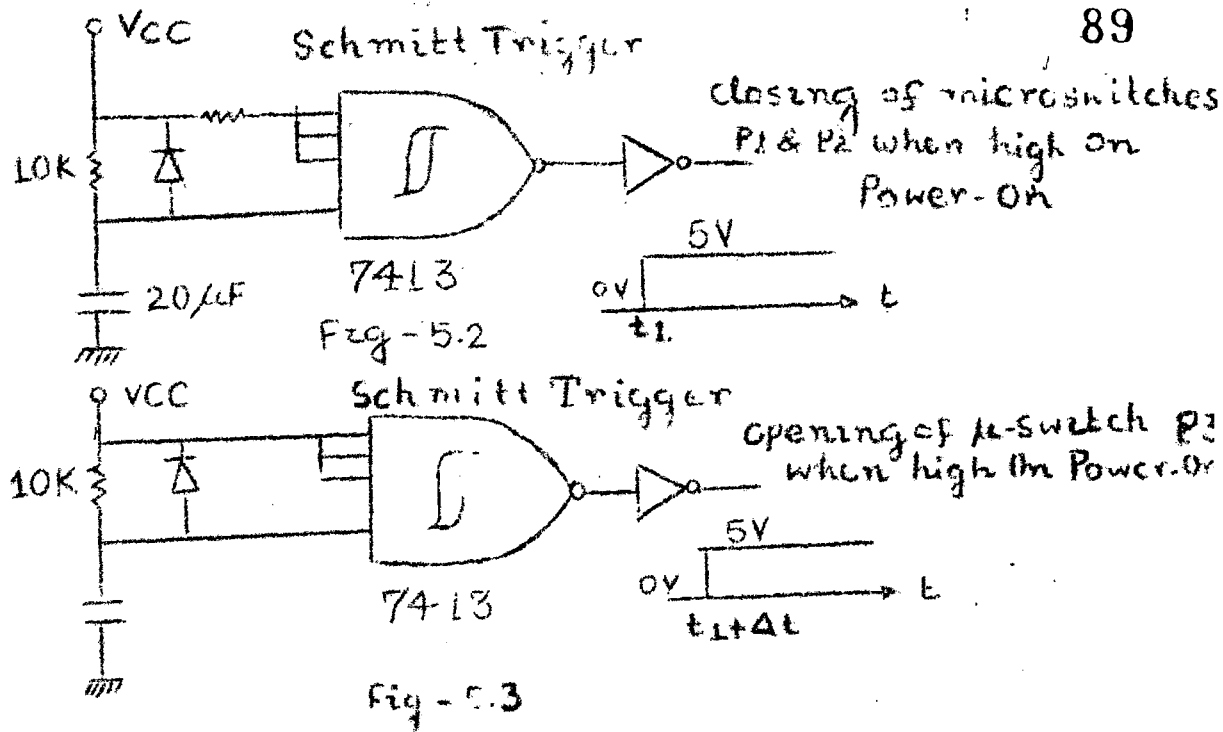


Fig-5.3



Between nodes A & D, the SEL-OUT signal can travel by two paths:

- 1) Via Microswitch P3 - this switch remains closed when the power to this device is switched off, thus enabling the SEL-OUT signal in reaching to the next device on this bus. On power on condition, this switch is opened with the help of schmitt trigger as shown in Figure - 5.3
- ii) Via microswitch P3 and P1 and device selection ckt. - At the power off condition, the switches P1 and P2 remain open, thus preventing the SEL-OUT signal from going to the controller ckt. At power on condition, this two switches are closed with the help of a schmitt trigger as shown in figure - 5.2. In this case, SEL-OUT signal, BUS-OUT, OUT-TAG signals are allowed to reach the interface logic. Thus the SEL-OUT signal reaches the AND gate 'A' where it is

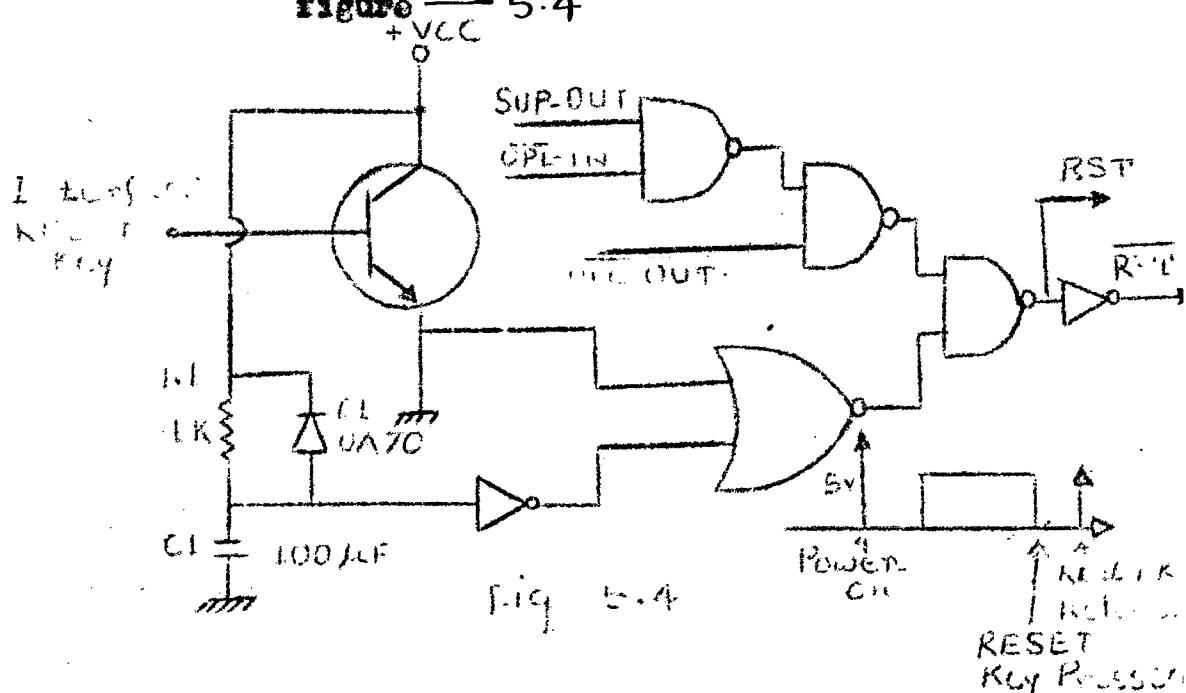
ANDed with another signal SELECT which is low when address matching takes place and High when address matching does not take place. Thus the outcome of this gate is low (i.e. SEL-Out Blocked) when address match takes place and high (i.e. SEL-OUT propagated) when address match fails. When SEL-OUT is propagated, it is passed to the node B through the bus driver chip 74126.

(b) General Reset signal: There are three situations in which a general reset signal should be generated which will reset all the signals in the interface as well as the status byte. These situations are:

(1) System reset - is indicated whenever OPL-OUT and SUP-OUT are down concurrently, and the I/O unit is in the on-line mode. This condition causes OPL-IN to fall and all control units alongwith their status, to be reset. This is performed when the system reset key is pressed, when the power for the system is turned on, when the channel is off line to the I/O interface and as a part of the initial program loading procedure. To ensure a proper reset, OPL-OUT and SUP-OUT must be down concurrently for at least 6 μ Sec.

(ii) Selective reset - is indicated whenever SUP-OUT is up and OPL-OUT drops. This condition causes OPL-IN to drop, and the particular I/O unit in operation and its status to be reset. The I/O device operating on the channel is the only device that is reset. This is incorporated in the figure-5-4

(iii) Power on reset - When ever the Interface is powered on a reset signal should be generated so that all the signals and the status bits are reset. This is incorporated in the figure 5.4



(C) ADDRESS MATCH AND PARITY ERROR: Whenever information is transmitted over interface lines, it carries one extra line called parity line. In this system, odd parity is followed. So whenever any data command or address is transmitted over interface lines, it is

checked for the correct parity. When there is correct parity, the information is accepted otherwise proper status bit is set to indicate error in transmission. When it is correct parity, the device address is latched into a comparator and the internally stored address of the device is compared with the address transmitted over the interface. If these two are same, SELECT signal is generated. This signal high indicates that the SEL-OUT signal should be blocked and this device gets selected. This is realized in cards C1, C2 and C3.

- (d) Command Acceptance: When the device blocks the SEL-OUT it raises its ADR-IN and places, the internal addresses ON BUS-IN for the confirmation of the selection. This has been realized in cards C7, B12 and B13. Then channel sends the command code which is accepted by the card C5 and decoded by the card C4. Depending on the command the control logic does the proper function. It presents its status to the channel through BUS-IN. If the command happens to be the WRITE command, the control logic asks for the data byte through SRV-IN, SRV-OUT sequence. These data bytes are stored in the buffer. This is achieved in the cards C7, B01, B02, M1 and E2.

(e) IN-TAG SIGNALS GENERATION:

The information transfer from channel to device is done through interlocked sequences. The channel sends bus certain condition information on the OUT-TAG signals and BUS-OUT. The device controller in turn are expected to reply back with the help of IN-TAG signals and BUS-IN.

These IN-TAG signals are : OPL-IN, ADR-IN, SRV-IN, STA-IN. These have been realized in cards C6 and C7.

- (f) Buffering the data: The data points (two bytes) are received and stored in the buffer (cards M1 and M2). When writing into the buffer, its addresses are incremented by the WRITEP pulse, which is generated by the SRV-IN, SRV-OUT sequence. The data transfer is stopped in two conditions - when buffer becomes full or the channel transferred all the bytes. In that case, the Buffer writing mode is switched to Buffer reading mode and the buffer address is reinitialized to the first byte of the buffer. This has been realized in card DC2.

- (g) Plot signal generator:

When the buffer comes to reading mode, the two consecutive bytes of data are fetched from the buffer to load the presettable X and Y down-counters which allow the propagation of free running clock pulses to the appropriate X and Y lines of the plotter. This is realized in cards DC3 and PSG. When the counter becomes zero, the next two bytes are fetched and same procedure is repeated till all the data points are plotted. PENUP AND PENDING lines of the plotter are also driven by this card. During the buffer reading mode, the device presents busy condition to the channel for each TIO command. When all the data points are plotted, the interface control logic

presents ready status to the channel when TIO command is presented. Thus channel is ready to send another buffer if needed.

8-3 LOGIC DIAGRAM NOTATIONS:

In the Appendix A, the logic diagrams of the different components of the Interface are shown. These diagrams show the logical connection of each components as well as the physical location of the IC chips on the different PCB's. For the easy reference of these logic diagrams and to locate their physical position on the PCB's, the following conventions are followed:

(i) (a) ⁿ encircled capital letter of the English alphabet inside a rectangular block represents a complete chip on the PCB. Example: (A) represents chip 'A' on the PCB.

(b) A capital letter of the English alphabet in a rectangle inside a gate shows a complete chip on the PCB. Example [A] shows that this is the only gate in the chip and this is chip 'A' on the PCB.

(ii)(a) (A1), (A2), (A3) ----- (An) represent nth gate of the chip A on the PCB.

(b) [A1], [A2], [A3], ----- [An] represent nth gate of the chip A on the PCB.

(iii) (A.1), (A.2), (A.3) ----- (A.n) represents the nth component of the cascading in a cascaded ckt. Alphabet represents the chip on the PCB; the dot represents the

cascading of this chip with the other chip; and the numeric portion represents the logical position of this chip in the cascading. All the chips are put together.

(iv) One page may contain one PCB or many PCB's.

There is one heading for each PCB and the PCB name is represented by an alphanumeric name put in the parentheses in the heading. If many PCB's are shown on one page, their boundaries are separated by thick dotted lines.

(v) Input: In coming signals to a PCB are represented by (a) a thick black dot (•) generated at a PCB on different page.

(b) an ordinary line generated at a different PCB on the same page.

(c) small circle (○) '0' indicates that this signal has appeared at another place on this PCB.

(vi) Output: Out going signals from a PCB (i.e.

signals generated on this PCB) are represented by a dark arrow head (→).

(vii) Signal naming conventions: The signal names are shown along with the dots or arrowheads.

Alongwith the names are also shown the card ^{page} numbers on which the signal was generated in parentheses. The ^{page} ~~card~~ numbers are shown only with the darkened dots.

(viii) The continuity of a signal from one place in the card to another is shown by a combination (\rightarrow)

Example: Refer to the Appendix A, card C1.

In this card following are the incoming signals, STR from card \square , $\overline{\text{RET}}$ from page \square , BUS-OUT signals D0, D1, ----- B9.

Following are the signals generated in this card, B0, D1, B2, B3, B4, B5, D6, D7 and BP.

The reference (A-3) represents that this block is a D-type Flip Flop. This is the 3rd component (D-FF) of the chip 'A' which is type 74174 on this PCB. For the pin connections of this chip, reference should be made to Appendix C, card C1.

5-4 FUNCTIONAL DESCRIPTION OF THE PCB CARDS:

The interface logic has been fabricated in the laboratory in ten PCB cards. Each card is a single side metallic coated 82 pin experimental printed ckt.-board. Each IC is mounted on a socket put on the PCB. During the writing time, only the socket was mounted on the PCB in order to avoid heating of the chips. When whole the wiring was done, the chips were mounted on the sockets. In the following sections of this chapter, each card has been described separately at the functional level.

6-4.1 BUS-OUT BUFFER REGISTER (C1):

Refer to the card C1 in the Appendix A. The input to this card are signals B00, B01, B02,--- B07, B0P. These are the BUS-OUT bits coming from the channel output Bus. The content of this bus is stored in the latches realized by the chips A and B, upon the STR signal going high. These bits are cleared in the beginning by the general reset signal \overline{RST} . The outgoing signals from this PCB are B0, B1, B2,---B7. This bits (address, command or data) are stored in this latch for subsequent use in the interface logic.

6-4.2 ADDRESS MATCH(C2):

This card is meant for the generation of a \overline{SELECT} signal which indicates whether the propagator of the SEL-OUT signal should be blocked or not. When the favourable condition occurs, in this card, i.e. the device address sent by the channel matches with the internal address of this device stored in this card, the corresponding bits of the two addresses are fed in the comparator chips A and B. The output of the two chips are ANDed. When the output of this gate is high and there is no address parity error, then the D.F.F E.1 contains high data at its D input. With the rise of signal BLD-OUT, this data is presented to the gates D4, which is a low signal, the \overline{SELECT} goes low, this indicating the blocking of the signal SEL-OUT. When the \overline{SELECT} goes low, the SEL-IN signal is raised high, thus informing the channel about

its selection. This is done during initial selection sequence or I/O unit busy sequence. The incoming Σ signals to this card are from card C1 and C3. The signal generated in this card is SELECT.

6-4.3 PARITY CHECK (C3):

This card is meant for the parity checking on the BUS-out lines. The parity followed in this system is odd parity. The chip used for the parity checking is 74180 which is the chip 'A' on the PCB. When the parity is correct, the Σ ODD output from this chip will be low. In the case of wrong parity on the Bus-out the Σ ODD output will be high. This parity result is fed to the three latches E1, D2 and D1. These latches are clocked by the signals COM-OUT, T DRV-OUT and ADR-OUT to indicate command parity error, Data parity error, or address parity error respectively. These parity error flags set different bits of the status register and sense register. When there is address parity error, the device does not get selected.

6-4.4 ACCEPTING COMMAND CODE (C4)

The input to this card are the Bus contents latched in card C1. The signal COMW generated in card C5, indicating the command state, clocks the bus content to the decoder logic. The high order bits of the command should be zero in order to be a legal command for this device. So these bits are NANDed together produce illegal command whenever any high order bit

is high. The low order bits D₀, D₁, D₂ & D₄ are directly fed to the 4 line to 16 line decoder chip (E) 74154. The FF₀, A₈ & B₈ are cleared when the STA-IN signal is generated i.e. the status of the device is sent to the channel. The signals generated in this card are:

TIOS, WRITES, REMINDS, SENSEEC, and ILLCOL.

The WRITES signal is cleared by the STOP signal generated at card BC6. Other signals are cleared by the fall of OPL-IN signal generated in card C6.

5-4.5 ADDRESS, COMMAND AND DATA STATES (C5):

The channel will be communicating with the controller in one of the three states, Address, command, or data i.e. the content of the BUS-OUT at any time will be one of these informations. None of these information can appear together. The important signals generated in this card are COME and STR. COME indicates command state and STR signal is the combined effect of all these states. When in the interface any one of these states are present, the STR signal goes high and the data from BUS-OUT is latched, in card C1. This card also serves as a spare card in which left over piece of ckt. from other cards are accommodated.

5-4.6 OPERATIONAL-IN AND STATUS-IN SIGNALS(C6):

5-4.6.1 OPL-IN

OPL-IN is a very important in-tag signal. This signal is cleared by the general reset signal RST. OPL-IN signal is raised high whenever the status of the device

is zero (ZEROS signal low), SEL-OUT signal is high and the address match OCCURS in the initial selection sequence (indicated by SELECT high) This signal is cleared when any one of the following conditions occurs:

- (i) STACKEND - When the device presents its status to the channel, the channel either accepts (raised SRV-OUT) it or asks it to stack it. This situation indicates that STA-IN should be dropped. The drop of STA-IN makes STACKEND to drop, thus finally dropping the OPL-IN.
- (ii) When the command sent to the CU be any one of these : PIO, REMID, NOP or ILLEGAL command, and the STA-IN has been replied by the SRV-OUT i.e. the status has been accepted.
- (iii) When the channel has sent SENSE Command, the CU responds with the STA-IN which is replied by back by the channel with SRV-OUT if status accepted. Then CU raises its SRV-IN and places SENSE byte on BUS-IN. The acceptance of this byte is indicated by the rise of SRV-OUT. This SRV-OUT forces the OPL-IN to fall.
- (iv) In the general reset case, this signal is cleared.

5-4.6.2 STA-IN

The STA-IN signal is raised in the following cases:

- (1) in the Initial selection sequence. In this case, the COMM is high, OPL-IN is high and the CMD-OUT falls. At the parity edge of CMD-OUT, a pulse is generated in monoshot C, which makes the output

of gate F1 low, thus making the output of gate F3 go high. With the rise of the signal, the \bar{Q} of monoshot goes low and then high, which clocks the D-FF J1. With this STA-IN goes high. Similarly, when the channel acknowledges the end of data transfer by raising CMD-OUT in response to SRV-IN the gate F2 goes low, thus raising the signal STA-IN. In the case of short sequence for busy devices, the ZEROS signal remains high and the SEL-OUT signal being high, the SELECT signal (showing address match) makes the output of gate A 1 low, thus generating the STA-IN signal. This signal is cleared in the following cases:

- (i) General reset signal.
- (ii) SRV-OUT in response to STA-IN signal.
- (iii) In the case of ^{sc}short-sequence for busy device when ADR-OUT remains high and SEL-OUT drops. With the fall of SEL-OUT the output of H4 gate goes low, thus clearing the STA-IN signal.
- (iv) CMD-OUT in response to STA-IN indicating STACK status.

6-4.7 SERVICE-IN, ADDRESS-IN, BUFFER-WRITE-PULSE and BUS-IN PRIORITY BITS (C.7).

6-4.7.1 ADR-IN

When OPL-IN is low, the \bar{Q} of FF N1 remains high. When OPL-IN goes high in the initial selection sequence the two inputs of gate D1 remains high. Only the third input coming from monoshot A remains low. When ADR-OUT

goes low, this input also rises, thus clocking the D-FF H2. This makes ADR-IN to rise.

This signal is cleared, when CMD-OUT rises in response to ADR-IN. The output of gate E1 goes high when CMD-OUT rises in response to ADR-IN, thus clocking E1. The \bar{Q} of E1 goes low and the H2 is cleared which makes ADR-IN to go low. This is also cleared by general reset signal.

6-4.7.2 SRV-IN

This signal is generated in two cases:

- (i) When asking for a data byte when the interface is in WRITE state. In this case, the WRITES signal is high, \bar{VE} is high and OPL-IN is high. The trailing edge of SRV-OUT a pulse in monoshort B, which makes the output of gate G4 go low. In this situation, all the inputs of gate F3 goes high and the D-FF C2 is cleared, which means rise of SRV-IN.
- (ii) When sending a SENSE byte in SENSE state. In this case SENSES signal is high. In this case again the fall of signal SRV-OUT makes the output of gate E4 to go low, thus raising the signal SRV-IN.

This signal is cleared in the following cases:

- (i) OPL-IN drops.
- (ii) CMD-OUT rises.
- (iii) General reset signal
- (iv) when D-FF C2 is clocked. The output \bar{Q} of monoshot L remains high till it gets a clock

at input B. When the signal STA-IN is not present and SRV-OUT rises in response to SRV-IN, the output \bar{Q} of L goes low for a short duration and again goes high. At this high going edge, the D-FF C2 is clocked and SRV-IN goes low.

6-4.7.3 BUFFER-WRITE-PULSER:

This pulse is generated whenever SRV-OUT rises in response to SRV-IN, thus indicating a data byte transferred to the CU. SRV-IN, SRV-OUT combination gives rise to signal WRITE P which stores the data byte from the BUS-OUT to the buffer. The address of the buffer is incremented by one with each WRITE P pulse.

6-4.7.4 BUS-IN PRIORITY BITS:

The bits PEB1 and PEB2 are generated for the routing of the address, status, and sense registers to BUS-IN. When the ADR-IN rises with the rise of OPL-IN, the output \bar{Q} of D-FF E2 goes low while the outputs of E1 and E2 remains high. This makes PEB1 and PEB2 both low. Thus indicating that the address register should be routed to the BUS-IN. When the STA-IN rises, the STATUS signal goes high, and the output \bar{Q} of D-FF E1 goes low while the outputs \bar{Q} of E2 and E3 remains high. This makes PEB1 as high and PEB2 as low. This indicates that the status register should be routed to BUS-IN.

In the third case, when $\overline{\text{STARTEN}}_3$ goes low, the signal PEB1 goes low and PEB2 goes high. This indicates that sense register should be routed to BUS-IN.

6-4.8 DIRECTING THE REGISTERS TO BUS-IN (BI1, BI2 and BI3)

The combination of these three cards directs the three registers address, status and sense to BUS-IN with proper parity generation.

6-4.8.1 LOW ORDER BITS (BI1)

This card handles the low order 4bits of the registers. The combination PEB1 -PEB2 as '00' selects the lines 1Co and 2Co from chip A and 1Co and 2 Co from chip B for outputting to 1Y, 2Y of chip A and 1Y, 2Y of chip B. Similarly the combination PEB1-PEB2 '10' selects the status register bits ST0, ST1, ST2 and ST3 for outputting. So is the case with sense register.

6-4.8.2 HIGH ORDER BITS (BI2)

This card handles the high order 4 bits of the registers in the similar fashion as the card BI1.

6-4.8.3 PARITY GENERATION AND BUS-DRIVER(BI3)

This card is meant for the generation of parity bit through the use of chip 74180. The Bus drivers are enabled only when any of the signals OPL-IN, STA-IN or SRV-IN is present.

6-4.9 BUFFER WRITE CONTROL CIRCUIT(BCL)

The main purpose of this card is to generate addresses for the buffer. The addresses are generated through the counter 74197 p.1, L.2. The counter is initialized to zero with the general reset signal. In this case,

The Buffer address is 0. The data transfer starts with the SE signal low meaning that buffer is write enabled. When the first byte of data is transferred to the CU, the SRV-IN, SRV-OUT combination gives rise to one WRITEP pulse. This pulse is used to generate a clock pulse CK# in card BC2. This clock is used to increment the counter P.1 by one. Thus with the each data byte transferred across the interface, CK# signal is generated and addresses of the buffer goes on incrementally till all the bytes from the CCW are transferred. At this moment, the buffer becomes full and it generates under flow signal $\overline{\text{UNDFLOW}}$. This signal changes the signal WE from low to high and data transfer is stopped. At this stage the interface goes into buffer reading mode and the address of the buffer is reinitialized to 0 (gates A1, A2, A3, B1, B3, A4, and C1).

In the case, when the channel has lesser number of byte than the buffer size, the channel indicates STOP signal by raising CND-OUT in response to SRV-IN. At this moment, the last address is stored in the FFCs J1, J2, I1, I2, F1, F2, E1 and E2. While reading the buffer, the read addresses are compared with the stored addresses to know when to stop reading. At this moment disable signal goes high.

6-4.10 BUFFER READ/WRITE CONTROL CIRCUIT(BC2)

The primary function of this card is to generate WE signal (indicator of read/write mode of the buffer). When WE is low, the buffer is write enabled, otherwise

read enabled. With the general reset signal, the \overline{VE} goes to low state. Then the full buffer data is transferred, the STOP signal rises and the output of A6 goes high, thus making output of C2 high. Thus \overline{VE} goes high and buffer goes to read mode. With this the extended \overline{VE} pulse ($X-\overline{VE}$) goes high for a short duration. The $\overline{X-\overline{VE}}$ goes low for a short duration. When it is low the CKD is cleared and \overline{Q} of F1 remains high. Then $\overline{X-\overline{VE}}$ goes high D-FF I2 is clocked and the CK# signal is generated which is used in card DC3 to fetch first byte of data from the Buffer. Rising of CK# generates a clock pulse CK which increments the address of the buffer in card DC1. Now when the first byte of data has been fetched and address pointing to 2nd byte, the Q of D-FF J1 goes high which makes the \overline{Q} of monoshot H to go low for small duration. This gives one clock to the toggle F1 and the Q of this toggle goes to high level, thus raising the signal CKD which fetches the 2nd byte of the data from the buffer in card DC3. These two byte are loaded into a counter with the clock pulse $\overline{X-clock2}$. This is decoded in the cards DC3 and P60 to generate plot signals in X and Y directions with the proper pen positioning. Then both the plot counts become zero, the XY signal goes low and this generates one clock pulse CK#. This CK# increments the Buffer address by 1, thus pointing to 3rd byte of the buffer. This XY also gives one clock to toggle F1 and the \overline{Q} of this

toggle goes high. Thus CKA goes high and the previous procedure is repeated. This way all the bytes from the buffer are plotted. When either $\overline{\text{UNDFLOW}}$ or $\overline{\text{DISADL}}$ occurs in card DC1, the state of VE signal is changed. This presents the channel ready state to receive another set of data bytes.

6-4.11 256 BYTE BUFFER (M1 and M2)

These two boards combined makes 256 bytes of buffer. The low order 4 bits are stored in card M1 and high order 4 bits in the card M2. BA0, BA1, ---B7, are the address bits and B0, B1, B2, v...B7 are the eight bits of the data byte. The chips used are 74200.

6-4.12 BUFFER OUTPUT CONTROL CIRCUIT (BC3)

The two consecutive bytes of the buffer are loaded in D-FFs A1, A2, A3, A4, D1, D2, D3, D4 by CRA pulse and in F1, F2, F3, F4, H1, H2, H3, H4 by CKB pulse. The high order six bits from the first byte is loaded in X-counter B.1 - E.1. The high order six bits of 2nd byte are loaded in X-counter G.1 - I.2. The P1BT1 and P1BT2 from byte 1 are directly fed to the pen position lines of the plotter. The direction bits DRBT1 and DRBT2 are used in card PSG to decode the direction of movement in X & Y direction (+or-). All the counter outputs are NANDed to generate X-PASS signal which passes the free running clock pulses to the plotter till all the input lines become high indicating that count is over. Similar is the case with Y-PASS and Y-counter. The X-counter is decremented by the X-DELAY signal and Y counter by Y-DELAY signal generated in

card PEG.

5-4.13 PLOT-SIGNAL GENERATION (PEG)

This card is meant for generating the four plot signals $+X, -X, +Y, -Y$. Only one X signal and one Y signal will be present at a time. These signals directly goes to the plotter input lines.

The DRDT1, DRDT2 combination selects the X and Y lines. The frequency generator (yet to be implemented) is generating the free running pulses. This is toggling the J-K FF J1. Alternatively \bar{Q} and Q of J1 is going high. As long as the X-PASS is high in Buffer read code, the Q signal from J1 is passed to the data selectors C1 and C2. Depending on the DRDT1 and DRDT3 signals, either C1 will select the signal or C2. C1 will send the signals to $+X$ while C2 to $-X$. Each pulse from Q of J1 produces an X-DELAY pulse after a delay of 7μ Sec. (plotting speed-150 steps/Sec.). The frequency generator can be accordingly designed for the purpose. Similar is the situation with Y signals.

CHAPTER 6CONCLUSION

In this chapter, the topics covered are-design discussion, future extension, software considerations and improvement in the existing design. The present design is based on certain assumed parameters, which the IOCS software (Device driver) is envisaged to create before the data transfer job is passed to the device controller.

6-1 GENERAL CONSIDERATION:

The design philosophy adopted in this work is based on the following points:

- (a) Smallest increment = 0.2 mm at a maximum rate of 150 steps/Sec.
- (b) Software will send data as a difference between two consecutive points (X₀, Y₀) and (X₁, Y₁).
Example = A(10,5), B(15,10).
- (c) X-count or Y-count supplied by the software should not be more than 64 steps in each block of command and data.
- (d) Software will supply Data-Order chain (each of two bytes) as follows:

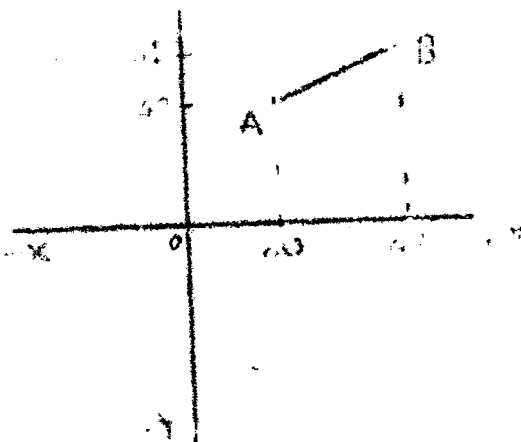
X-COUNT [Data] + PENUP/PENDN [order]

Y-COUNT [Data] + WRITE(±X, ±Y) [order]

This constitutes one block of command and data. One block will be followed by other blocks in plot-program output.

- (e) There will be a continuous plotting. Pen will go up only when it is indicated by the programmer in the order to introduce discontinuity in the plotting.
- (f) Plotting area is divided into four quadrants (+X,+Y), (+X,-Y), (-X,+Y) and (-X,-Y).
- (g) The manual placement of the pen on the paper determines the origin. All movements are relative to this position.
- (h) Scaling and windowing is done by the software. It knows the maximum plotting area. Accordingly it chooses the scale factor based on the information - X-minimum and Y-minimum.
- (i) Example -- Plotting a straight line between points A(20,20) to B(45,25). In the buffer data will be stored in the following manner.

<u>BYTE NO</u>	<u>CONTENT</u>
00	[<X> =64] + [PENUP]
01	[<Y> =84] + [+X,+Y]
02	[<X> =36] + [PENUP]
03	[<Y> =36] + [+X,+Y]
04	[<X> =64] + [PENDN]
05	[<Y> =13] + [+X,+Y]
06	[<X> =61] + [PENDN]
07	[<Y> =12] + [+X,+Y]

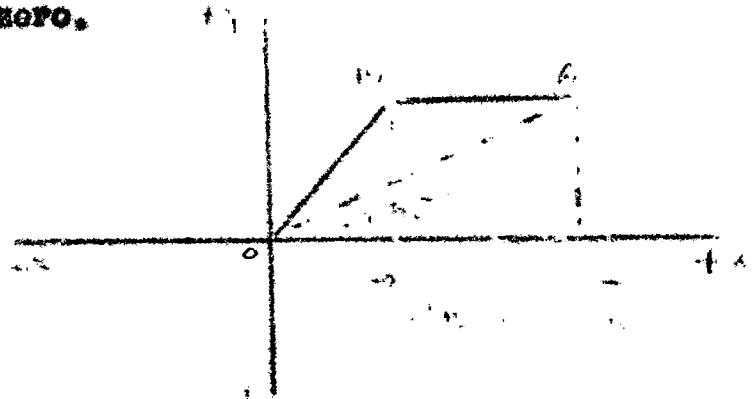


Explanation: Execution of these data points (two consecutive bytes) will start with the pen going up and moving the pen and paper alternatively by 64

steps in $+X$ and $+Y$ respectively. Then again keeping pen up, record data point is plotted by moving pen and paper alternatively 36 times in $+X$ and $+Y$ direction respectively. The effect of this will be - pen moving to point A without plotting. In the similar fashion it plots the other two data points keeping pen low, so that continuous line is drawn between A & B .

RESOLUTION OF PLOTTING:

When supplying data to the pen carriage and drum motor, pulses are given alternately till one of the counts decrements to zero.



When one of the counts becomes zero, straight movement occurs in the other direction whose count is still non-zero.

Let us examine the discrepancy that occurs when the lines are plotted using the above mentioned method. The amount of discrepancy depends on the angle the desired line makes with the X-axis. There is no discrepancy in the cases when the desired lines are at an angle of K where $K=45^\circ$ and K any integer. Thus in the case of line falling on any of the axes or at an angle of 45° with the axes, there will be no discri-

pancy between the fn plotted line and the desired line.

The maximum discrepancy occurs, when the line falls at an angle of $K\theta/2$ where $\theta = 45^\circ$ and K is an odd integer. Thus maximum discrepancies will be at the angles, $22.5^\circ, 67.5^\circ, \dots, 337.5^\circ$ from X-axis.

With 1 byte (8 bits) the counter value will be 255 to send 256 pulses in one direction. So the maximum discrepancy in this situation is

$$\Delta = FQ = OQ - OP \quad (OA \text{ is the desired line \& } OMA \text{ is the plotted line})$$

$$= OQ - MP$$

$$= OQ - AQ$$

$$= OQ - OQ \tan 22.5^\circ$$

$$\Delta = 256 - 256 \tan 22.5^\circ \quad (\text{Assuming that } OQ \text{ can be plotted with 256 steps each of } 0.2 \text{ mm})$$

$$\Delta = 256 - 0.6 \times 256 = 100 \text{ Steps } \text{-----(A)}$$

But 100 Steps (= 20mm) is quite a high error.

By assuming that an error of $\frac{1}{4}$ th of this error i.e.

25 steps = 5mm, will be a tolerable one, equation

(A) becomes,

$$\Delta \cdot \frac{1}{4} = 256/4 = 0.6 \times 256/4.$$

But $256/4 = 64$ can be represented in only six bits. Thus in one block of data, maximum allowable data count is 64.

For the two counters (X counter and Y counter) two data bytes are required. The first byte for X-counter and second byte for Y-counter. Only the six bits of first byte is used for data count.

The remaining two low order bits are utilized to pass-on information about the pen positioning associated with this block of data. Similarly, the six high order bits of the 2nd byte contain Y- count and the remaining two low order bits are used to pass-on information regarding direction of movements $(\pm X, \pm Y)$.

6-2 FUTURE EXTENSIONS AND IMPROVEMENTS:

The existing design is based on the parameters described before. These parameters have been chosen in such a way that the project could be finished in the limited time frame. The present design is a simplified one, so that it could be implemented in the laboratory within the existing constraints. Further improvements could be made in order to increase the total throughput of the system and the resolutions of plotting. The following are the areas in which improvements can be made.

6-2.1 RESOLUTION OF PLOTTING:

Since the plot signal generator does not do any processing on the X- and Y- counts, the X and Y counts of the two bytes are directly used to send those number of pulses in X and Y directions. Thus the software is burdened to generate many data blocks having smaller counts in order to improve the resolution of plotting. This means more number of data bytes to be created and increased data transfer duration. To illuminate this

problem, one Arithmetic and Logical Unit (ALU) chip could be used in the Plot signal generator board which will make necessary calculations on the X & Y counts (larger values) before loading the X and Y driver counters. The ALU and the X,Y registers load control ckt may be controlled by the signs involved in the order ($\pm X, \pm Y$) and the magnitude of X and Y counts. The arithmetic operation may be done on $(X,Y)_n$ and $(X,Y)_{n-1}$ data blocks to calculate the magnitude to be loaded to X&Y registers of the plot signal generator board. The X and Y registers, then may load two presetable down counters which will allow propagation of the free running clock to send necessary number of impulses for X and Y direction.

6-2.2 HARDWARE TECHNOLOGY:

The implementation of this design has been done with the help of TTL, MSI integrated ckt. chips. The individual IC chips are connected with other IC chips with external wires. This makes the ckt. more cumbersome to implement and maintain. This design could have been implemented with the help of Microprocessors and RAM, ROM memory chips. The Microprocessor based design are easy to implement and maintain and it is more efficient than the MSI IC based ckt.

6-2.3 PLOTTER ckt. IMPROVEMENTS:

The present plotter does not provide any line indicating the plotting status to the controller. The Ready and Not-Ready lines can be generated at the plotter and sent to the controller to form the overall status of the plotting system.

When the plotter is in Manual mode then it should make Not-ready line high to tell the controller about its unwillingness to accept computer orders. When the operator puts it in automate the Ready line should be made high indicating its willingness to accept the computer orders.

6-2.4 CARD-LAYOUT:

The hardware implemented for this interface has been done with the help of standard PCB cards available ready made in the market. These cards contain provisions for IC mounting and external wiring. Due to the restrictions of 28 pins percard and limited space on the card, too many cards were needed for the purpose. Also, due to the external wiring, the cards look clumsy and can give trouble at high frequency operation. If printed ckt boards are manufactured in stead of using ready made cards, the need for external wiring does not arise. By making use of the PPLS software package supported by SPERRY UNIVAC 1100 systems, the detailed PCB layout can be generated. PPLS package accepts the pin-to-pin connections table as input (as given in the Appendix B) and gives detailed PCB layout on the printer or plotter.

6-2.6 BUFFER SIZE:

Presently the Buffer size has been chosen as 256 bytes. The CCT generated are required to contain 256 bytes of data to be transferred in one shot. The last CCT may contain less number of bytes. The larger buffers should be

incorporated if the graphics system needs it.

6-2.6 INTERRUPT HANDLING FACILITY:

In the present case the software has to keep constant watch on the status of the device controller. When ever the controller goes in buffer read mode, the channel is presented with the device busy status. Thus this status is passed to the software indicating that next CCW should not be executed until the device becomes ready. Without the REQ-IN protocol signal, the device controller does not send its ready status automatically. In turn, the software has to regularly check the status of the device by issuing TIO instruction. This slashes down the CPU and channel efficiency. Thus by incorporating REQ-IN signal the software can be informed of the ready status of the device. This relieves the software from issuing regular TIO instructions.

6-3 SOFTWARE CONSIDERATIONS:

The present design is based on the trade-off between hardware and software considerations in the implementation of the interface. This section deals with two components of the software development. General strategy and detailed specifications. In the general strategy case user facilities are taken into consideration while in the detailed specification case the IOCS software requirements are considered.

6-3.1 GRAPHICS SOFTWARE-GENERAL STRATEGY:

The general graphics software can be divided into three categories:

- * Application Programs

- * Functional Software
- * Basic Software.

6.3.1.1 APPLICATION SOFTWARE:

The highest level of Graphic Software is the Application Program. An application program is a complete problem solver. The user need only supply data and select among program options to obtain the desired graphical Output -no programming is required on the user's part. Application programs can be written in higher level languages - FORTRAN IV etc. The Application Program normally uses a Basic software. Some of the typical application programs can be:

FLOWGEN - to automatically produce and plot flowchart of any ANS FORTRAN IV PROGRAM directly from source deck.

DATAGRAPH - to produce management information charts and graphs directly from user data on cards, tape or disc

6-3.1.2 FUNCTIONAL SOFTWARE:

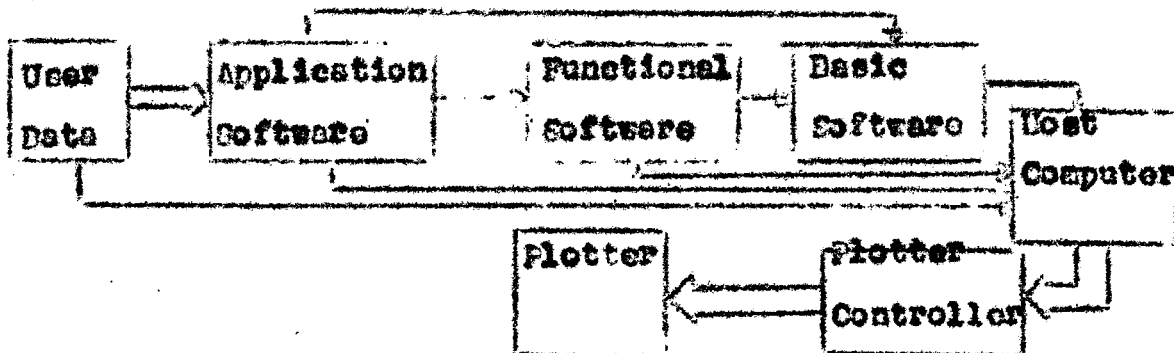
Functional software is an intermediate level of software which relieves the user of programming many commonly used graphic functions. Functional software is further subdivided into host computer resident and controller resident type software. Host computer resident functional software is usually written in FORTRAN. It consists of a set of subroutines to perform particular operations. The subroutines can be CIRCLE (to draw a circle or spiral), FLINE (to draw a smooth curve

through a set of data points), BAR (to draw bars, for bar graph plotting), ARROW (to draw a line terminated with an arrow head through a set of data points) and many more. These subroutines should cater to the needs of different disciplines likely drafting, business, scientific and general. A controller resident functional software is relevant only in the cases of programmable - controller.

6-3.1.3 BASIC SOFTWARE:

The lowest level of software used in producing a plot is termed basic software. This consists a set of subroutines which allow the programmer to perform elementary plotting operations such as drawing lines or annotation, selecting a pen, scaling the plot, etc. The basic software generates the commands necessary to perform the specified operation and transmits them to the plotter if it is on-line.

The plot commands generated by the basic software may be the actual codes necessary to move the plotter, but usually they are codes which represent the input data in a highly compact and efficient format. These codes then interpreted by hardware which then produces the actual codes necessary to drive the plotter. The figure given below shows the relationship between the types of software.



The Basic software should contain a set of subroutines written in FORTRAN and/or Assembly language which can control elementary operations of the plotter and provide some aid in plotting graphs. These subroutines may be called by the user programs written in FORTRAN or Assembly. The subroutines to be written are:

PLOT - to plot straight line between two points and to establish plot origin.

SYMBOL - to Plot annotation and special symbols.

NUMBER - to plot decimal equivalent of an internal floating point number.

SCALE - to determine starting value and scale for an array of data to be plotted on a graph.

AXIS - to draw annotated axis line for a graph.

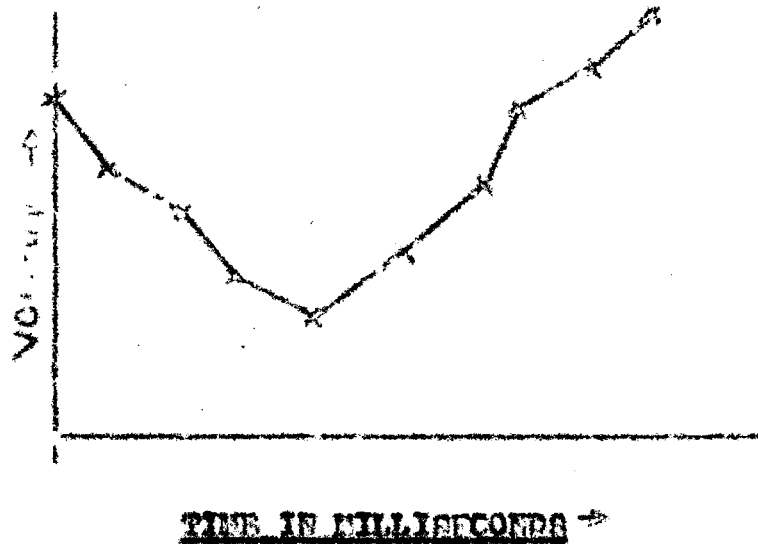
LINE - to scale and plot a set of data points

WHERE - to return current location.

6-3.1.4 SAMPLE PLOTTING PROGRAM:

	DIMENSION XARR(10), YARR (10)	1
10	CALL PLOT (0,0,8)	2
20	READ 25, (XARR(I), YARR(I), I =1,10)	3
25	FORMAT (2F 6.2)	4
30	CALL PLOT (0.0, 0.5,-3)	5
40	CALL SCALE (XARR, 5.0, 10,1)	6
50	CALL SCALE (YARR,6.0,10,1)	7
60	CALL AXIS (0.0,0.0,25H TIME IN MILLISECONS)	8
70	CALL AXIS (0.0,0.0,7H VOLTAGE)	9
80	CALL LINE (XARR, Y ARR,10,1,2,4)	10
110	CALL PLOT (12.0, 00,999) ████████	11
120	STOP	12
	END	13

The plotted graph will look like:



Line 1 reserves space for 10 data points. Line 2 initializes the PLOT subroutine with the logical number of the output device as 8. Line 3 reads 10 pairs of TIME and VOLTAGE from an input file into two arrays XARR and YARR. Line 5 establishes a new origin because of negative third argument. Line 6 computes scale factors for use in plotting TIME values within a 5 inch plotting area. Line 7 computes scale factor for VOLTAGE data values within a 6 inch plotting area. Line 8 draws the TIME axis. Line 9 draws the VOLTAGE axis. Line 10 plots the VOLTAGE vs TIME drawing a line between each of the 20 scaled points and a symbol X at every other point.

6-3.2 IOCS SOFTWARE SPECIFICATIONS:

Instructions required to accomplish any I/O operation are generally quite involved and lengthy. Yet for different devices, instructions needed to accomplish the I/O operations are identical. To simplify a

programmer's work, standardised routines are provided by the manufacturers. These routines are collectively called IOCS software and perform such functions as:

1. Opening and closing of data files.
2. Reading and Writing of records.
3. Blocking and deblocking of records.
4. Error condition checking etc.

The IOCS consists of two parts: Physical IOCS and Logical IOCS. Physical IOCS controls the actual transfer of record between the external medium and the main storage i.e., the actual reading and writing of the records.

Physical IOCS consists of the following routines:

1. Start I/O routine - initiate the I/O operation
by issuing the SIO instruction.
2. Interrupt routine - gets control when the operation
is over.
3. Channel Scheduler.
4. Device error routines.

These routines are part of the supervisor, which is permanently located in lower main storage while problem programs are being executed.

A new IOCS routine for this particular plotter has to be developed and integrated with the existing IOCS routines. Logical IOCS perform those functions that are needed to locate and access a logical record for processing. IOCS Modules ; There are many IOCS routines available with the DC 1020 system. These modules provide necessary

instructions and support to accomplish the input/output functions of the problem program. An installation could include every one of these many IOCS modules in the relocatable library to support every possible I/O functions. But to obtain better storage utilization, each installation, at system generation time may include a set of these IOCS modules which are based on the equipment being used. For the plotter use, the IOCS module for this device, should be provided in the relocatable library.

Records can be transferred to or from an Input/Output device by issuing physical IOCS macro instructions. These macro instructions communicate directly with the physical IOCS. When using physical IOCS to perform I/O operations, the programmer must provide the functions such as setting up channel command words (CCWs).

Three macro instructions are available to a programmer for direct communications with physical IOCS. These are:

1. CCB - Command control block
2. EXEC - Execute channel program
3. WAIT - Wait till the ICC operation is over.

Whenever physical IOCS macro instructions are used a programmer must also construct CCWs for the I/O operations.

Plotter IOCS module should include following specifications :

1. It should send one SIO instruction for a CCW containing 256 bytes of data out of the data area provided by the problem program.
2. Whenever, the problem program sends data area more than 256 bytes, the IOCS module should

execute the SIO instruction in such a way that 256 bytes of data are sent per CCW.

3. After issuing the SIO instruction, it should check for the device status through TIO instruction and PSW bits. Device shows busy condition when it goes to the data plotting mode. When its plotting for the whole buffer is over, the IOCS should check for the readiness to accept another buffer of data bytes through regular issue of TIO instruction. When the device shows not busy condition, the another set of data bytes (256) should be sent to the device.
4. It should check for the status bits returned in the PSW and take appropriate action.

REFERENCES

1. System/ES Principles of Input/Output - Eina Ciniagina
ES-1020 system Manual.
2. Input/Output channels of system ES/1020 - Ivan
Dalbokov & Maxim Levi.
ES1020 System Manual.
3. System/ES Input/Output Units : ES-7074-01, ES-8012,
EC -7033/DW 3.
Nikola Kirov and Vladimir Todorov.
ES/1020 system manual.
4. IBM/360 system architecture
5. Digital computer design - Raymond H. Kline - 1977
Prentice-Hall, Inc., Englewood Cliffs, New Jersey.
6. Digital computer system principles - Herbert Hellerman
1974.
Tata McGraw Hill Publishing Company Ltd., New Delhi
7. An Introduction to standard Channel Interface.
SPERRY UNIVAC computer systems.
European Education and Training Centre, Switzerland.
8. The TEL DATA Book for design Engineers.
Texas Instruments Incorporated - U.S.A.
9. KELTRON Technical Manual -Graphic Terminal.
Kerala State Electronics Development Corporation Ltd.
10. Programming Calcomp Electronic chemical Plotters, 1977.
California Computer Products, Inc. California.
11. Operating systems - Kadnick and Donovan 1974.
Magrahill Kagakusha, Ltd.
12. Digital systems: Hardware Organisation and Design
P.J. Hill and G.R. Peterson 1973.

APPENDIX -A

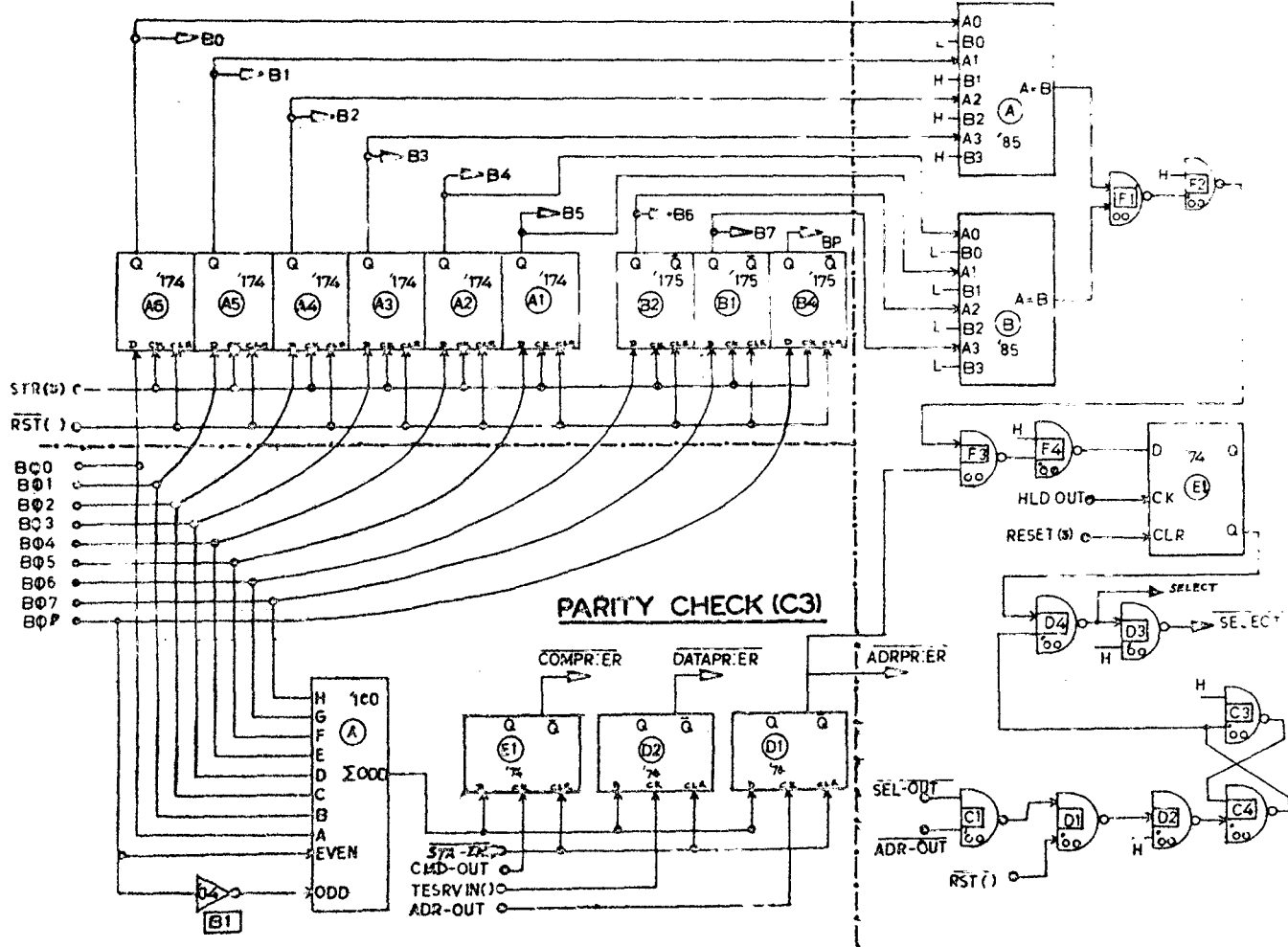
LOGIC DIAGRAMS IMPLEMENTATION:

In this appendix, all the circuit diagrams realized in thirteen PCB are given. How to refer to these PCBs has been described in chapter 5. The chip labels indicated in this diagram refer to their positions in the PCB, whose physical layout is given in Appendix B and chip pin description in Appendix C.

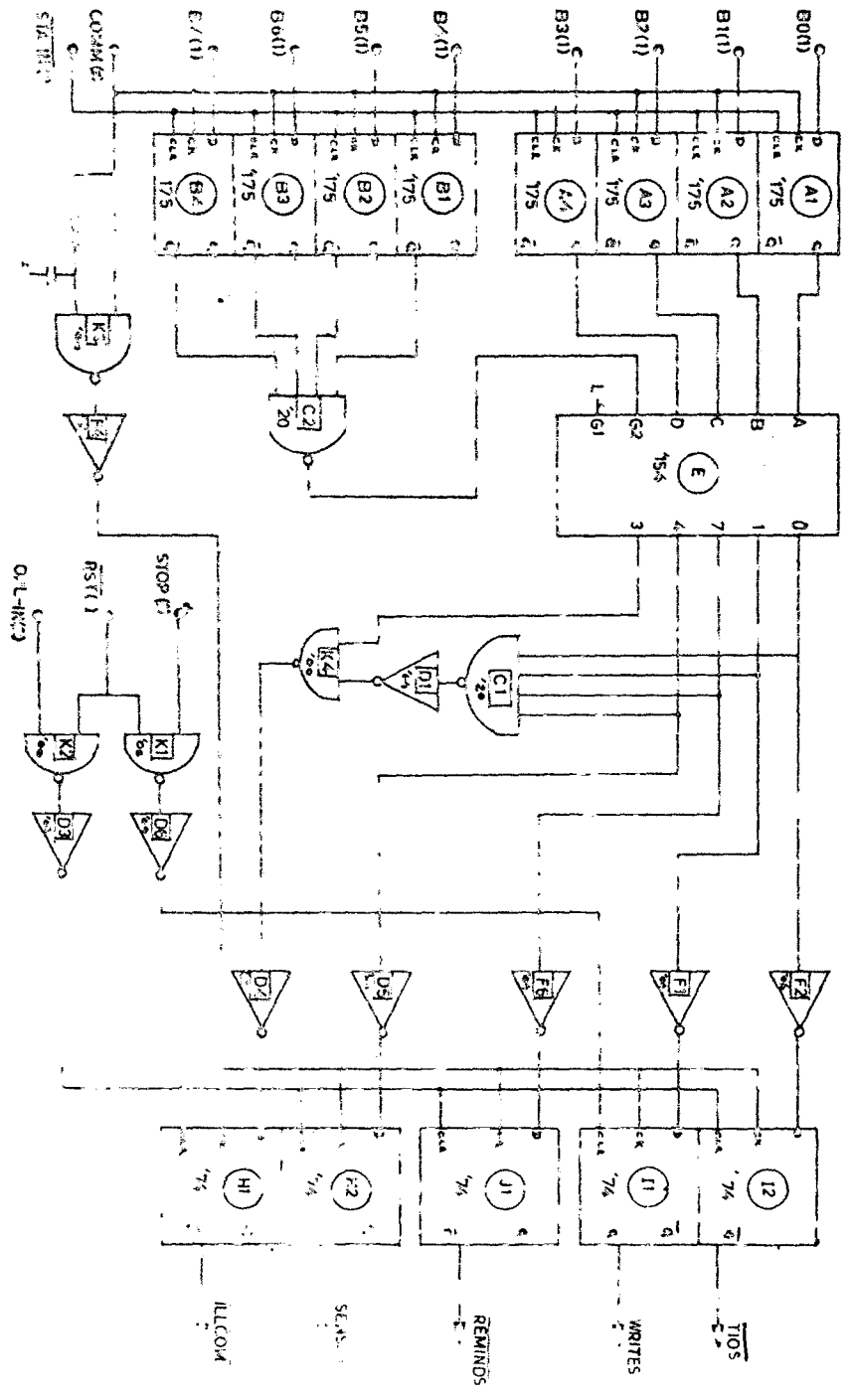
All the page numbers referred in the signal names are relative to this Appendix i.e. the first card in this PCBs is on page 1.

BUS-OUT BUFFER REGISTER (C1)

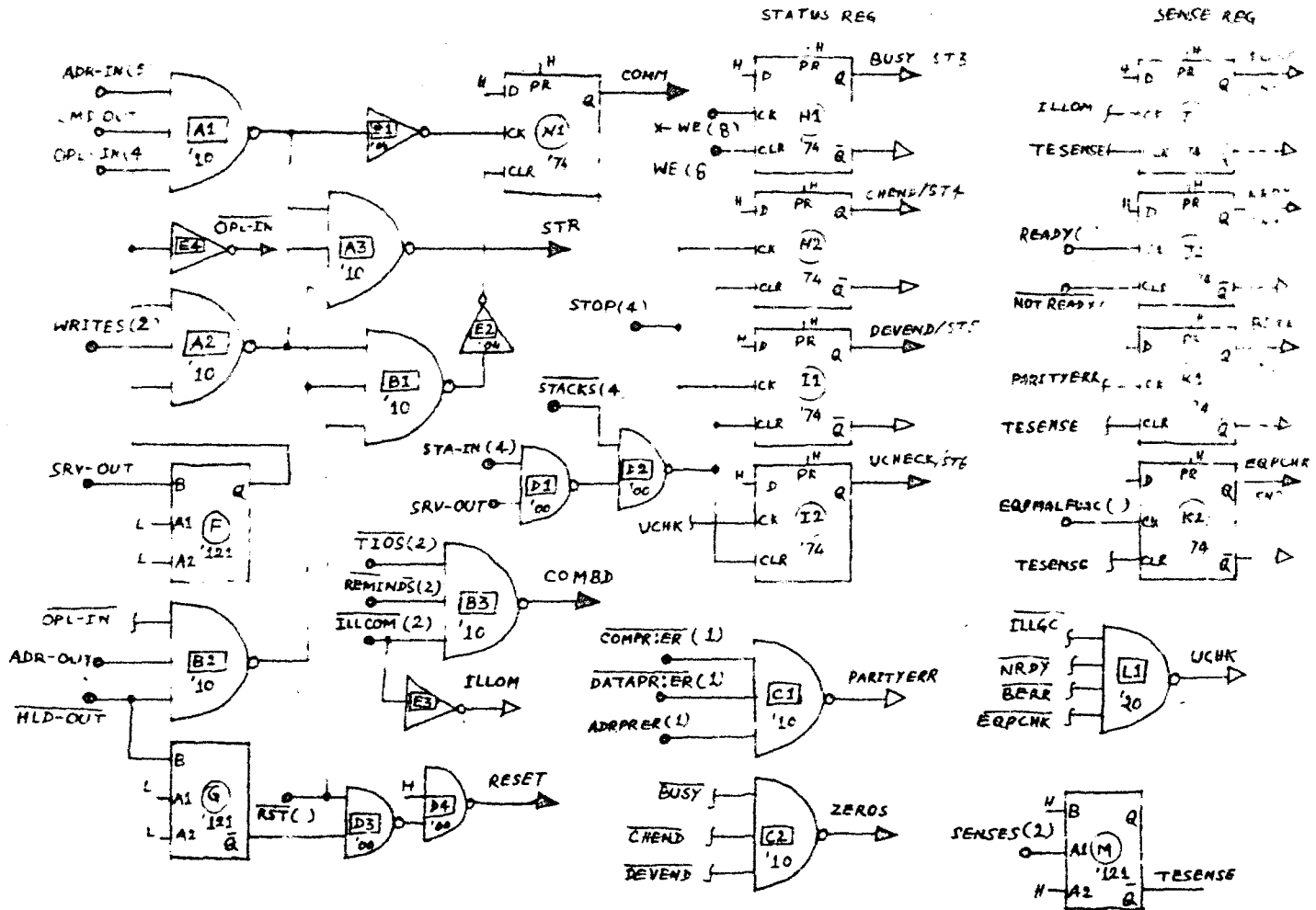
ADDRESS MATCH (C2)



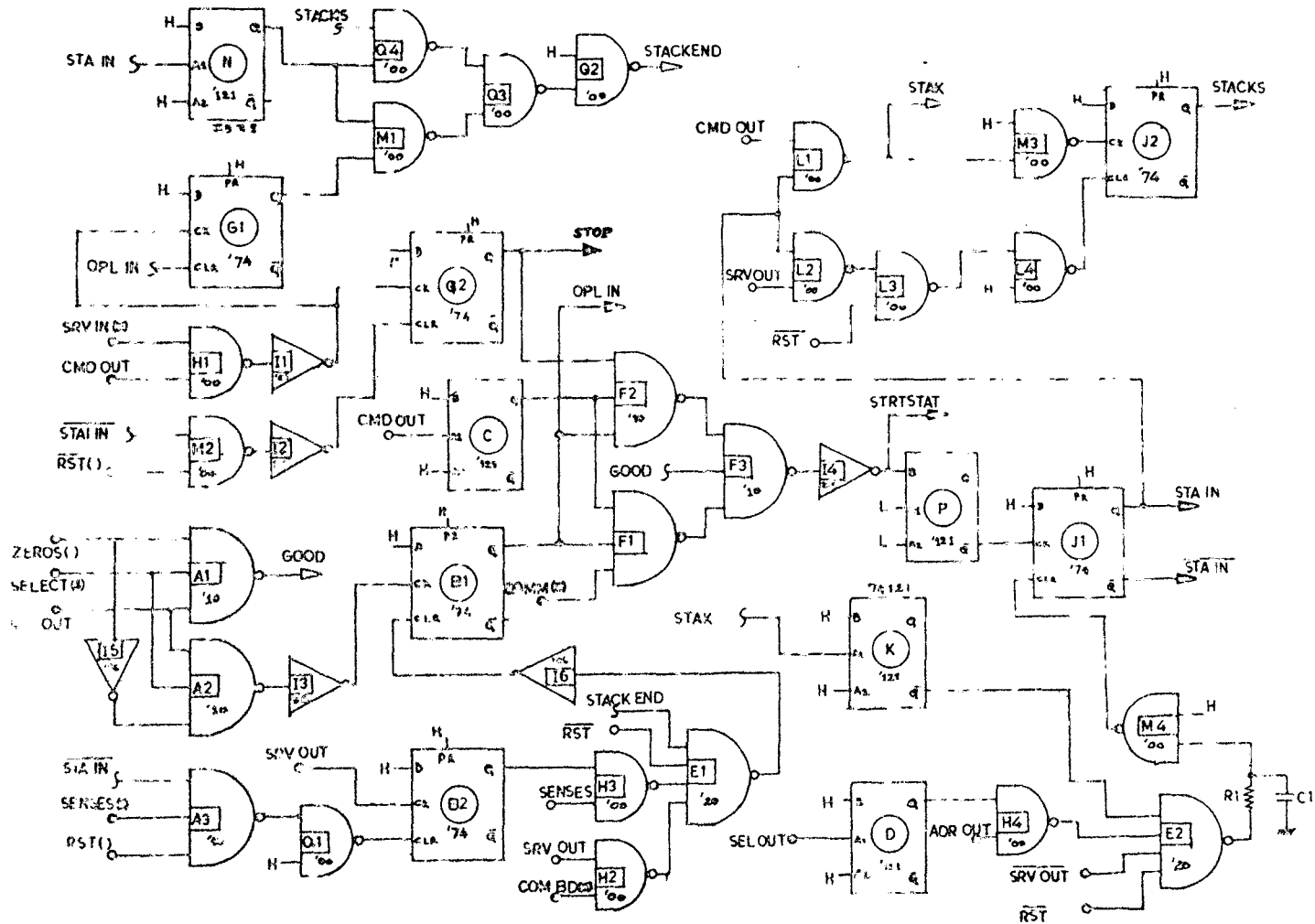
ACCEPTING COMMAND CODE (C4)



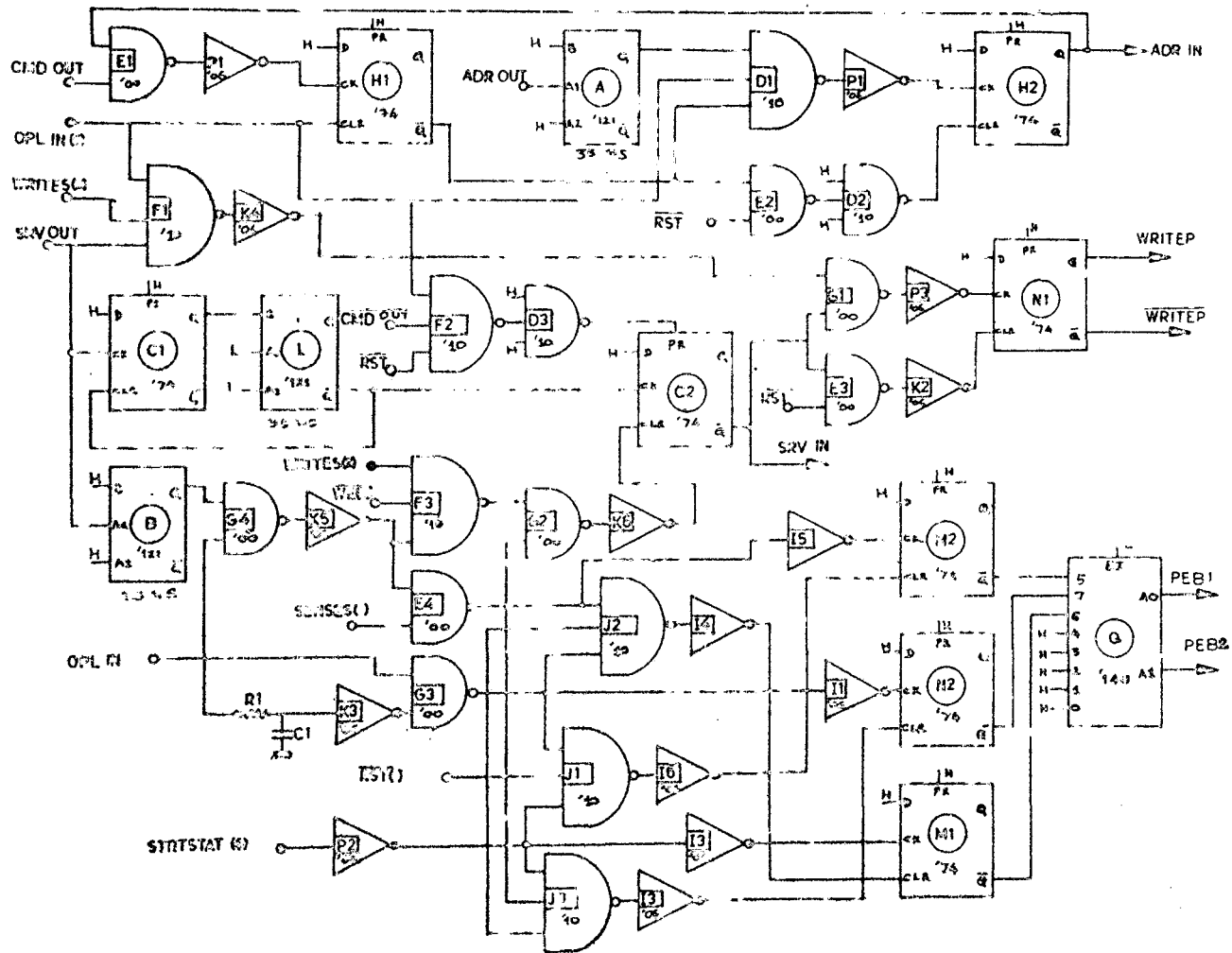
DATA, ADDRESS & COMMAND STATES (IC5)



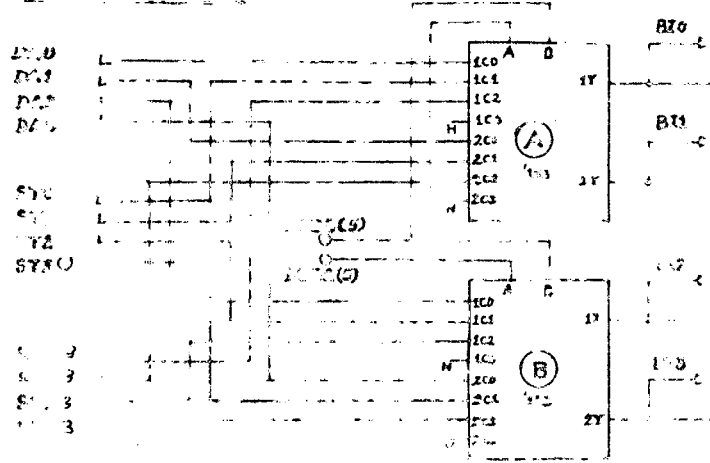
OPERATIONAL IN & STATUS IN SIGNAL (C6)



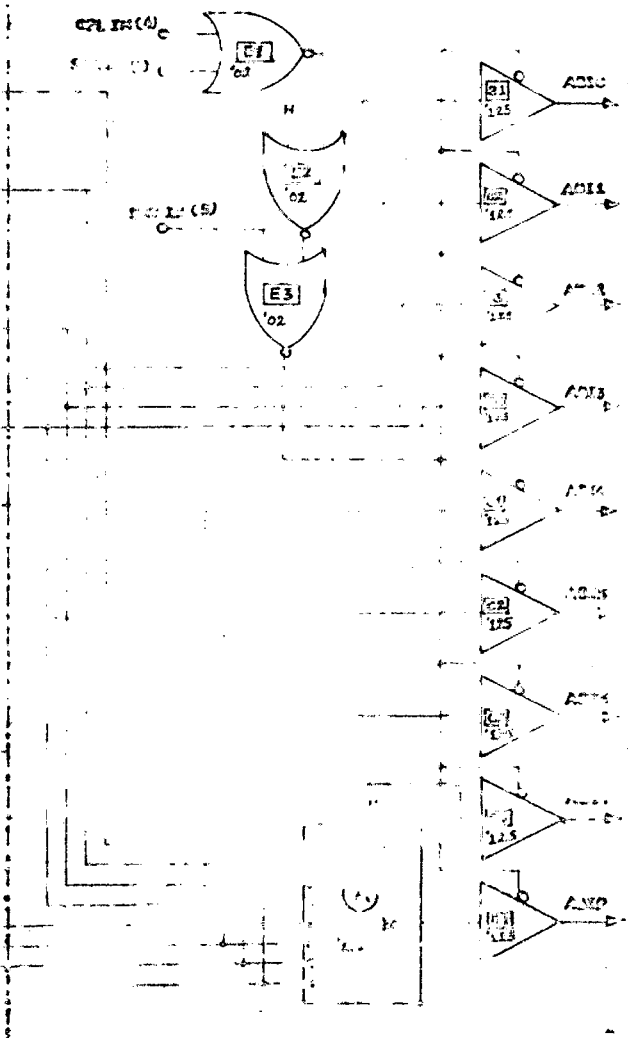
SERVICE IN, ADDRESS IN, BUFFER-WRITE-PULSE & BUS IN PRIORITY BITS (C7)



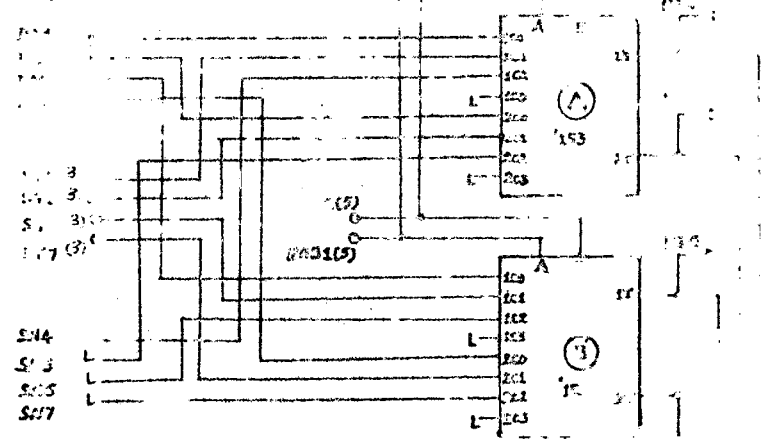
DIRECTING THE REGISTERS TO BUS IN (B11)
 Low Order Bits



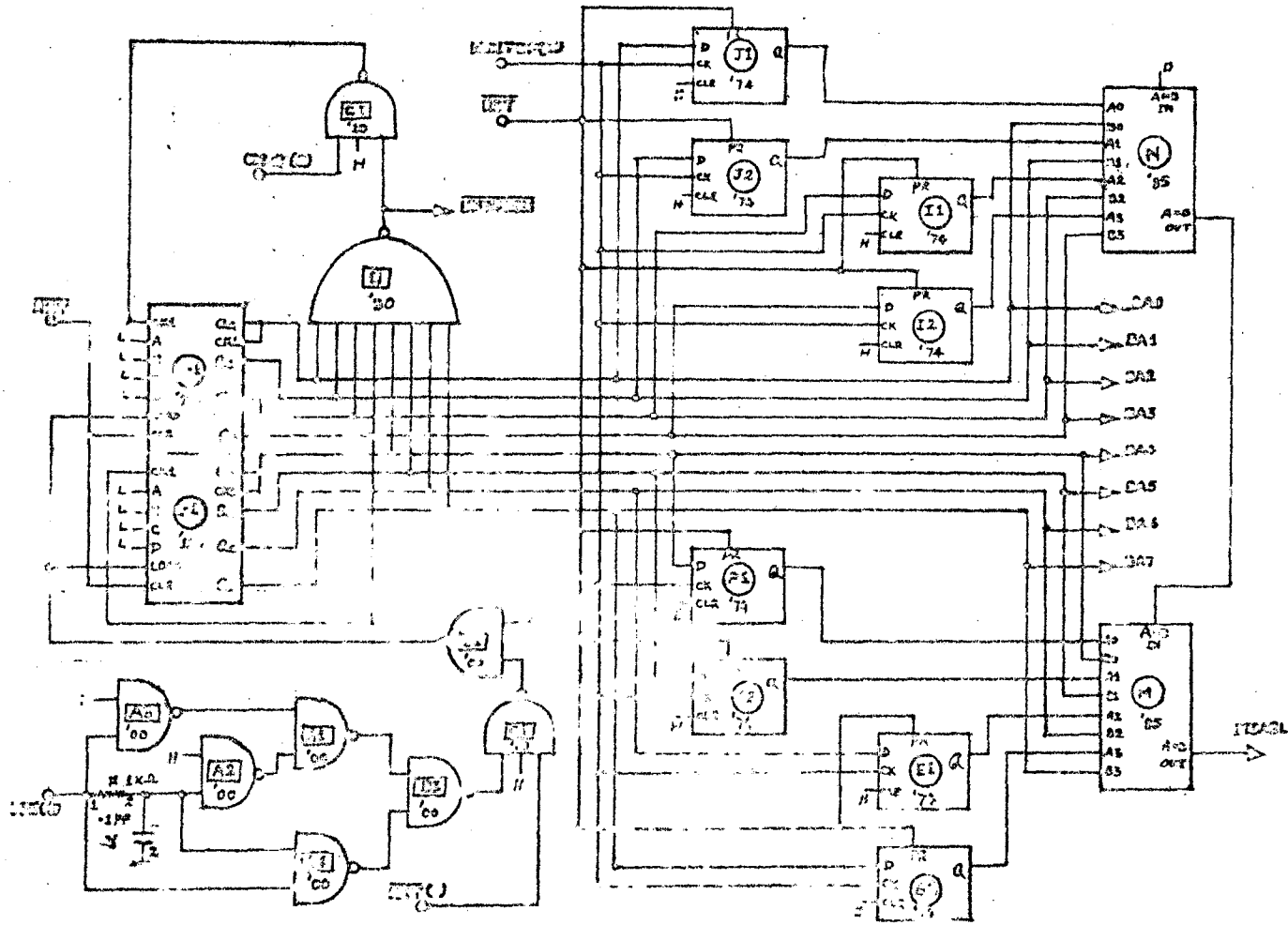
PARITY GENERATION & BUS DRIVER (B13)



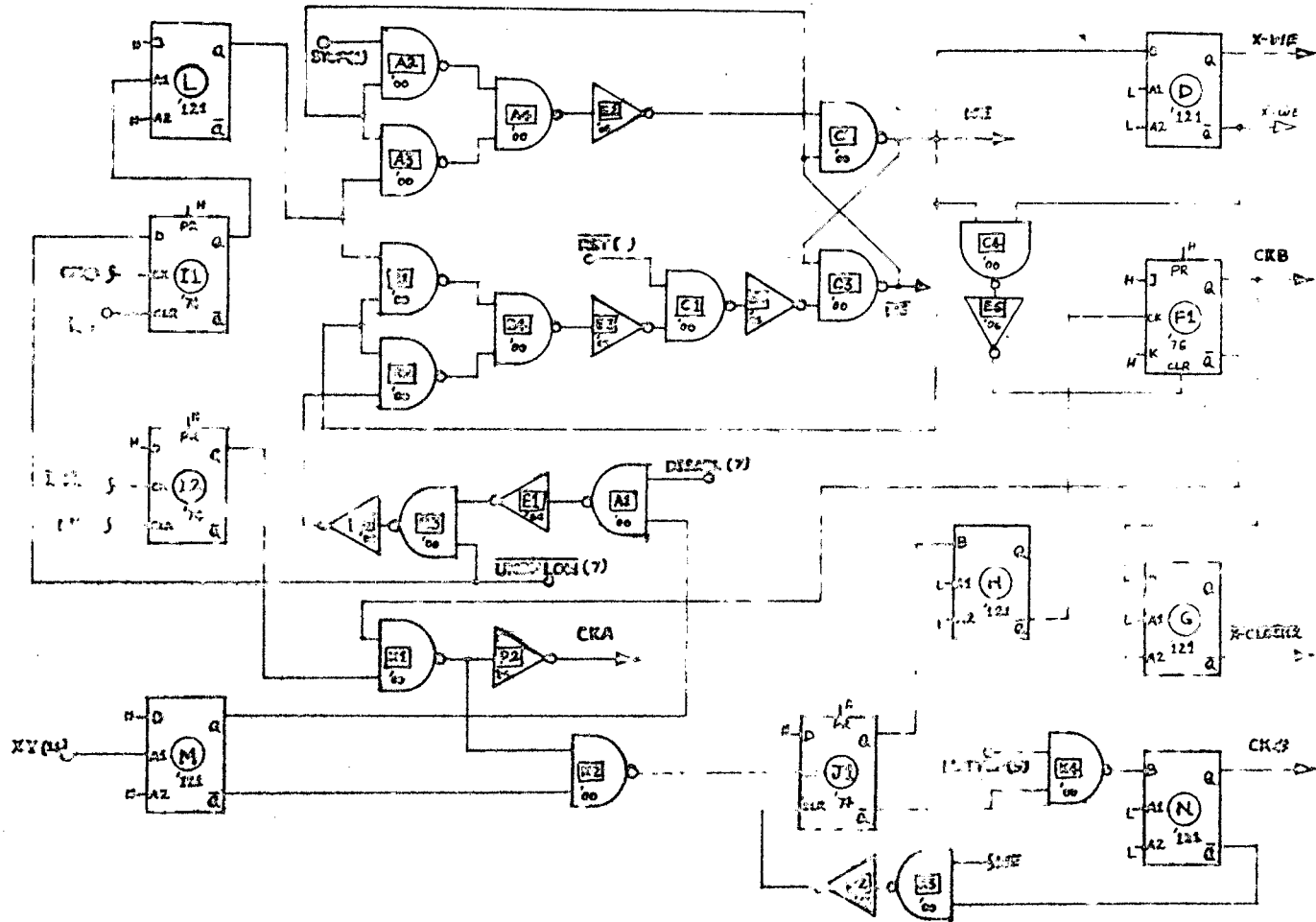
DIRECTING THE REGISTERS TO BUS IN (B11)
 High Order Bits



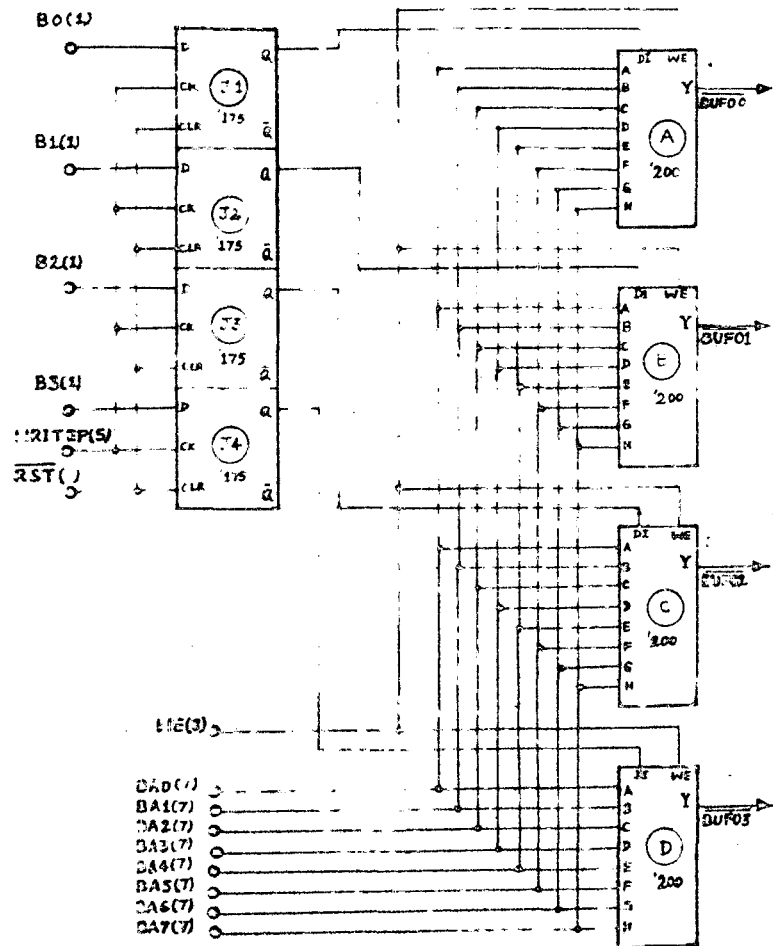
BUFFER WRITE CONTROL CKT. (BCI)



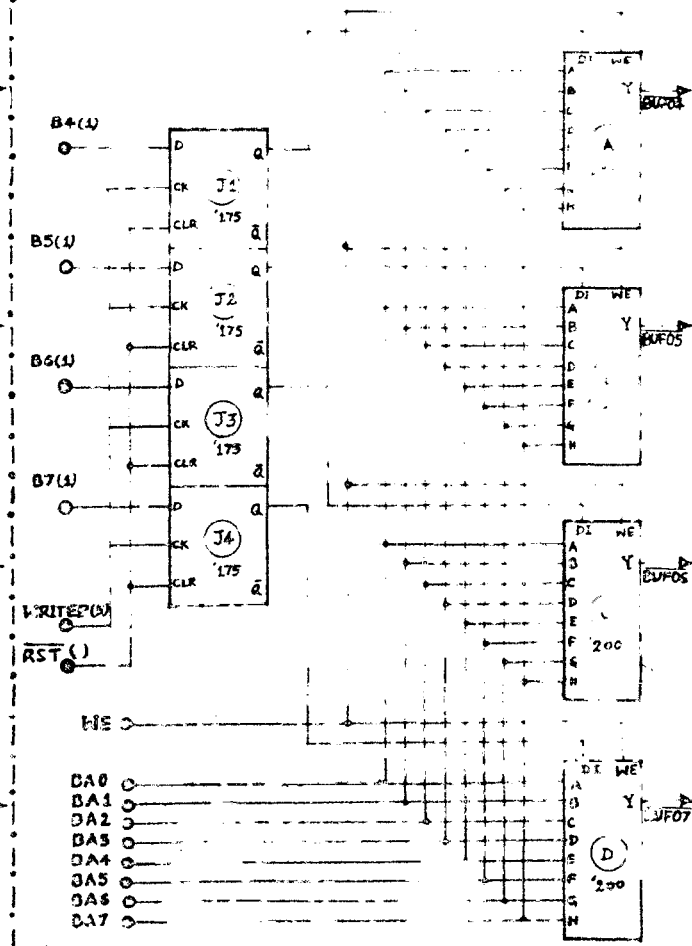
BUFFER READ/WRITE CONTROL CKT (CC2)



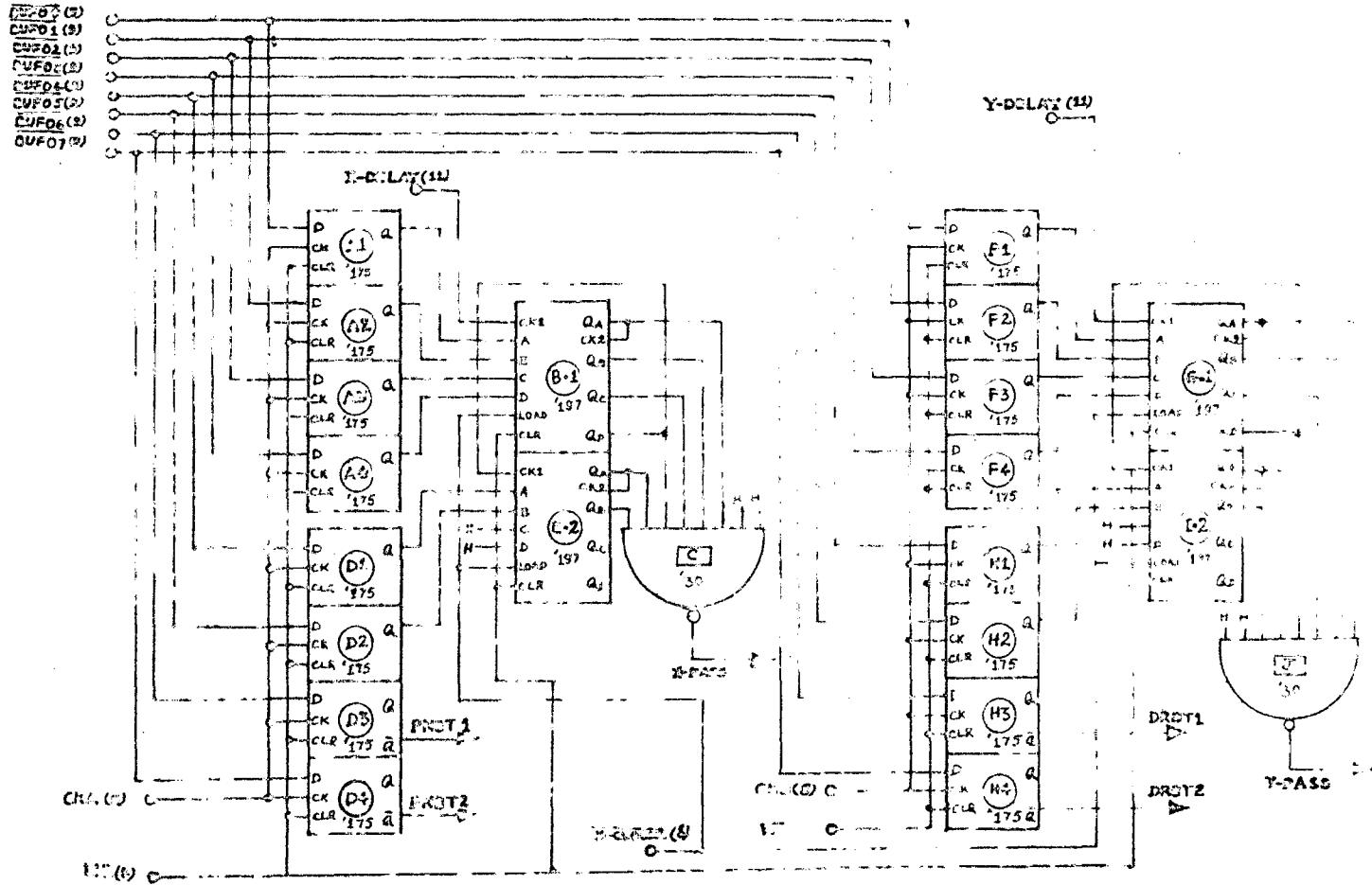
256 BYTES BUFFER-LOW ORDER BITS (M1)



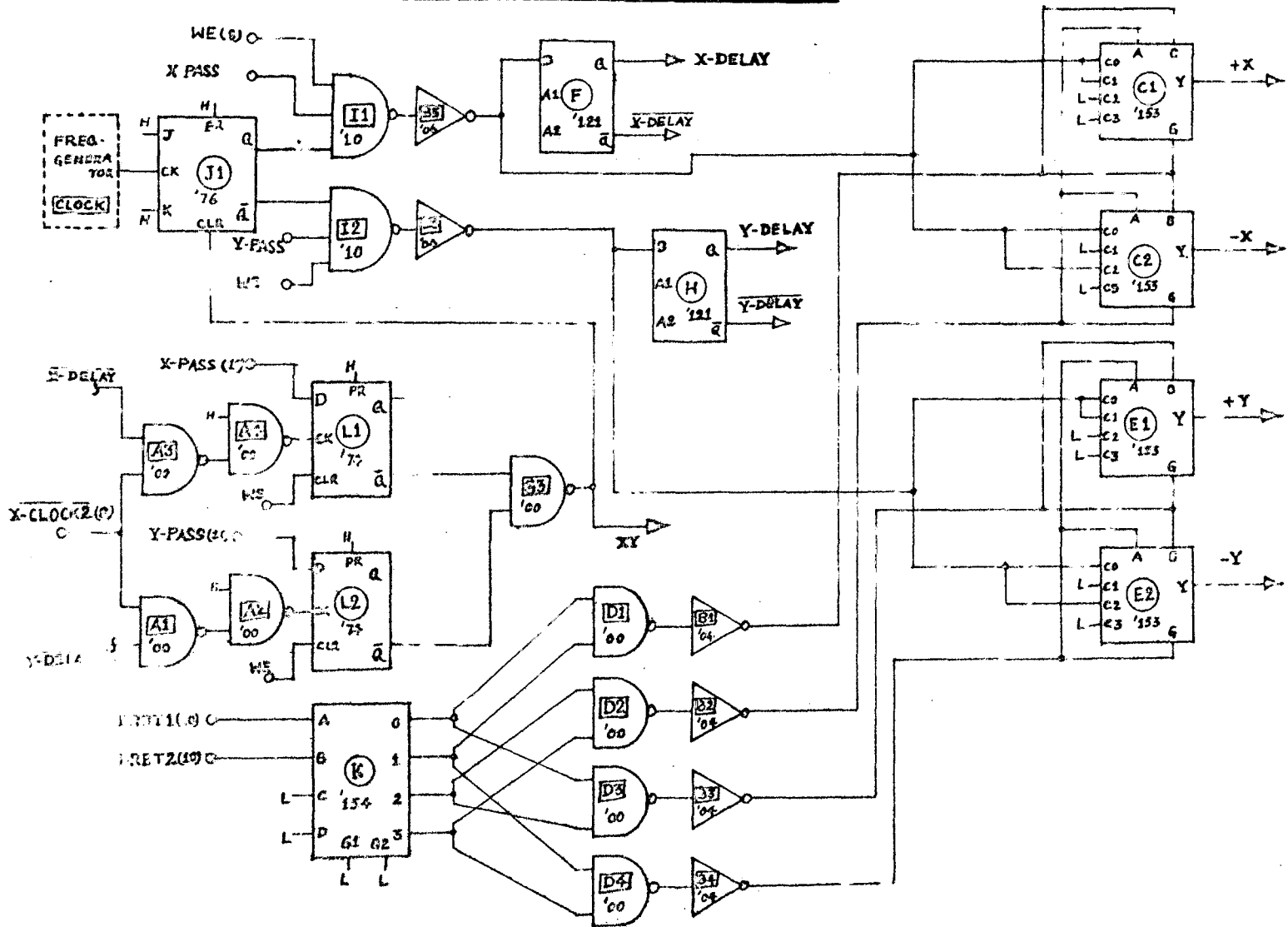
256 BYTES BUEFER HIGH-ORDER BITS (M2)



BUFFER OUTPUT CONTROL CKT (BC3)



PLOT SIGNAL GENERATION (PSG)



APPENDIX-B

BUS-OUT BUFFER REGISTER (C1)

<u>A</u>	<u>B</u>
74174	74178

ADDRESS MATCH (C2)

<u>A</u>	<u>B</u>	<u>C</u>
7486	7486	7400
D	E	F
7400	7474	7400

PARITY CHECK (C3)

<u>A</u>	<u>B</u>	<u>C</u>
74180	7408	74121
<u>D</u>	<u>E</u>	
7474	7474	

ACCEPTING COMMAND CODE (C4)

<u>A</u> 74175	<u>B</u> 74175	<u>C</u> 7420	<u>D</u> 7404
<u>F</u> 74154	<u>F</u> 7404		
	<u>G</u> 7400		
<u>H</u> 7474	<u>I</u> 7474	<u>J</u> 7474	<u>K</u> 7800

COMMAND, DATA, and ADDRESS STATES

<u>A</u> 7410	<u>B</u> 7410	<u>C</u> 7410	<u>D</u> 7400
<u>E</u> 7404	<u>F</u> 74121	<u>G</u> 74121	<u>H</u> 7414
<u>I</u> 7414	<u>J</u> 7414	<u>K</u> 7414	<u>L</u> 7420
<u>M</u> 74 121	<u>N</u> 7414		

OPERATIONAL-IN & STATUS-IN SIGNAL (C6)

A	B	C	D
7410	7474	74174	74121
E	F	G	
7420	7410	7474	
H	I	J	K
7400	7406	7474	74121
L	M	N	P
7400	7400	74121	74121

SERVICE-IN, ADDRESS-IN AND BUFFER-WRITE-PULSE GENERATION(C7)

A

74121

B

74121

C

7474

D

7410

E

7400

F

7410

G

7400

H

7474

I

7406

J

7410

K

7406

L

74121

M

7474

N

7474

P

7406

Q

74148

DIRECTING THE REGISTERS TO BUS-IN (BI 1)

A

74163

B

74163

BI 2

A

74163

B

74163

BI 3

A

74180

C

74126

D

74126

B

74126

E

7402

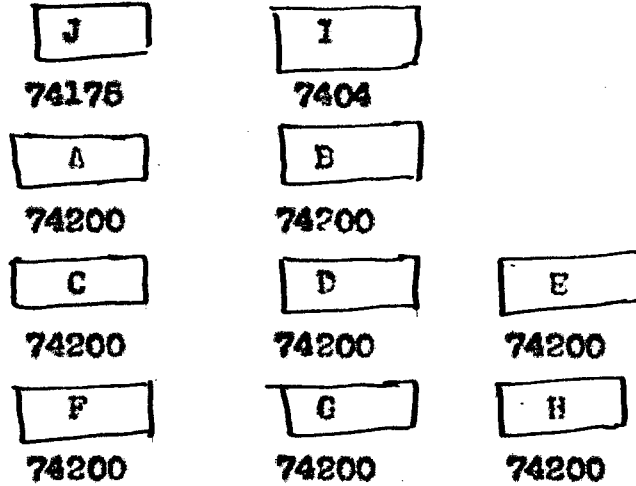
DUPPER ADDRESS GENERATION (DC 1)

A	B	C	D
7400	7400	7410	
E	F		
7474	7474		
I	J		
7474	7474		
M	N	O	P
7486	7486	7430	74187

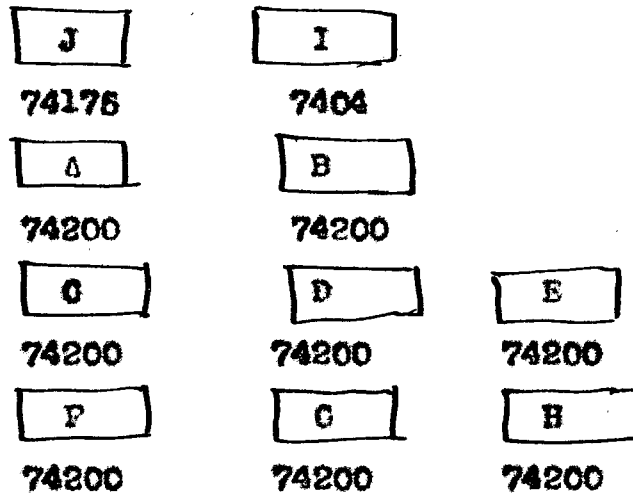
DUPPER READ/WRITE CONTROL (DC 2)

A	B	C	D
7400	7400	7400	74121
E	F	G	H
7406	7476	74121	74121
I	J	K	
7474	7474	7400	
L	M	N	P
74121	74121	74121	7406

256 BYTES BUFFER(L1)



256 BYTES BUFFER(L2)



BUFFER OUTPUT CONTROLLER (BC3)

A	B	C
74176	74197	7430
D	E	
74176	74197	
F	G	
74176	74197	
H	I	J
74176	74197	7430

PLOT SIGNAL GENERATION (PSG)

A	B	C
7400	7406	74163
D	E	K
7400	74163	74164
F	G	L
74121	7400	7474
H	I	J
74121	7410	7476

APPENDIX C

IC CHIP PIN ASSIGNMENTS

In this appendix, the logic and pin assignments for some of the important chips used in this design are described. In the case of logic gate, the pin numbers are shown in parentheses. The numbers before dash shows input and after dash shows output:

(1) 7400 - quadruple 2 input positive NAND gates
Gate 1(1,2-3), Gate 2(4,5-6), Gate 3(9,10-8)
Gate 4(12,13-11).

(2) 7400 -Hex inverters.

Inverter 1 (1-2), Inverter 2 (3-4), Inverter 3(5-6),
Inverter 4(9-8), Inverter 5(11-10), Inverter 6(13-12)

(3) 7406 - Hex inverters with open collector high voltage output.

Pin Assignment same as 7400.

(4) 7410 - Triple 3 -Input positive NAND gates

Gate 1(1,2,13-12), Gate 2(3,4,5-6), Gate 3(9,10,11-8).

(5) 7420 - Dual 4 -Input Positive NAND gate

Gate 1(1,2,4,5-6), Gate 2 (9,10,12,13-8).

(6) 7430 - 8 - Input positive NAND gate

Gate (1,2,3,4,5,6,11,12-8).

(7) 7474 - Dual D-Type positive-edge-triggered flip-flop.

Pin assignment and function table.

1st D-FF	PR (4)	CLR (1)	CK (3)	D(2)	Q(5)	$\bar{Q}(6)$
2nd D-FF	PR(10)	CLR(13)	CK(11)	D(12)	q(9)	$\bar{q}(8)$
	L	H	X	X	H	L
	H	L	X	X	L	H
	L	L	X	X	H*	H*
	H	H	↑	H	H	L
	H	H	↑	L	L	H
	H	H	L	X	Q	\bar{Q}

VCC (14)
GND (7)











(8) 7476 - Dual J-K Flip-Flops

Pin Assignments and function table

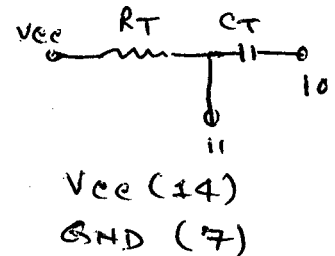
1st FF	PR(2)	CLR(3)	CK(1)	J(4)	K(16)	Q(15)	$\bar{Q}(14)$	VCC(5)
2nd FF	PR(7)	CLR(8)	CK(6)	J(9)	K(12)	q(11)	$\bar{q}(10)$	GND(13)
	L	H	X	X	X	H	L	
	H	L	X	X	X	L	H	
	L	L	X	X	X	L	H	
	L	L	X	X	X	H*	H*	
	H	H	⌋	L	L	Q	\bar{Q}	
	H	H	⌋	H	L	H	L	
	H	H	⌋	L	H	L	H	
	H	H	⌋	H	H	H	Toggle	

(9) 74121 - Monostable multivibrator

Pin assignment and function table

Δ -1(3)	A2(4)	B(5)	Q(6)	\bar{Q} (1)
L	X	H	L	H
X	L	H	L	H
X	X	L	L	H
H	H	X	L	H
H	↓	H		
↓	H	H		
↓	↓	H		
L	X	↑		
X	L	↑		

Δ is the pulse duration which can be adjusted a particular value with the help of Rext-Cext combination between pins 10 and 11.



(10) 74125 - quadruple Bus Buffer gates with three state outputs.

Gate 1(1,2-3), Gate 2(4,5-6), Gate 3(10,9-8),
 Gate 4(13,12-11).

Positive logic - $Y=A$, out put is off (disabled) when C is high.

(11) 74183 - Dual 4-line-to-1-line Data Selectors.

Pin assignment and function table:

	Select	Inputs	Data	Inputs		Strob	Output	
Selector 1	B(2)	A(14)	Co(6)	C1(5)	C2(4)	C3(3)	G(1)	Y(7)
Selector 2	B(2)	A(14)	Co(10)	C1(11)	G(12)	C3(13)	G(13)	Y(9)
	H	H	H	X	X	X	H	L
	L	L	L	X	X	X	H	L
	L	L	H	X	X	X	L	H
	L	H	X	L	X	X	L	L
	L	H	X	H	X	X	L	H
	H	L	X	X	L	X	L	L
	H	L	X	X	H	X	L	H
	H	H	X	X	X	L	L	L
	H	H	X	X	X	H	L	H

VCC(16)

GND(8)

(12) 74154 - 4 Line-to-16-Line decoder.

Pin assignment and Function table (partial)

INPUTS						OUTPUTS			
A(18)	B(19)	C(20)	D(21)	E(22)	F(21)	O(1)	1(2)	2(3)	3(4)
L	L	L	L	L	L	L	H	H	H
L	L	L	L	L	H	H	L	H	H
L	L	L	L	H	L	H	H	L	H
L	L	L	L	H	H	H	H	H	L
L	H	X	X	X	X	H	H	H	H
H	L	X	X	X	X	H	H	H	H
H	H	X	X	X	X	H	H	H	H

VCC(24)

GND(12)

(13) 74180 - 9-Bit ODD/EVEN parity Generators/Checkers.

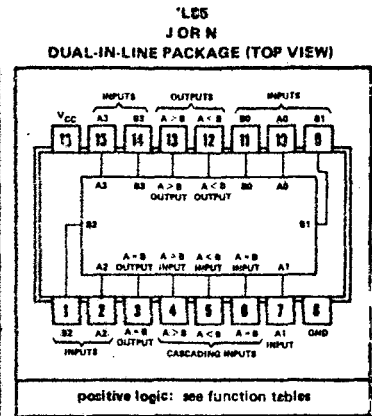
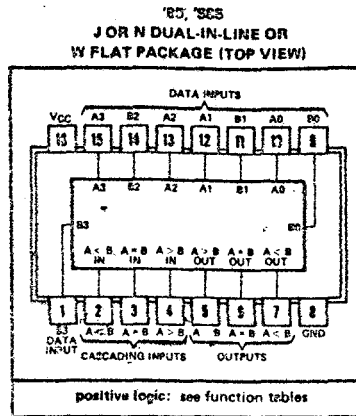
Inputs						Outputs		Output			
A(8)	B(9)	C(10)	D(11)	E(12)	F(13)	G(1)	H(2)	EVEN (3)	ODD (4)	EVEN (5)	ODD (6)
H's at A thru D				H	Even	H	L	H	L		
H's at A thru D				H	Odd	H	L	L	H		
H's at A thru D				H	Even	L	H	L	H		
H's at A thru D				H	Odd	L	H	H	L		
				X		H	H	L	L		
				X		L	L	H	H		

VCC(14) GND(7)

TTL
MSI

TYPES SN5485, SN54L05, SN54S05,
SN7485, SN74L05, SN74S05
4-BIT MAGNITUDE COMPARATORS
BULLETIN NO. DLS 7211810, DECEMBER 1972

TYPE	TYPICAL POWER DISSIPATION	TYPICAL DELAY (4-BIT WORDS)
'85	275 mW	23 ns
'L85	20 mW	90 ns
'S85	385 mW	11 ns



description

These four-bit magnitude comparators perform comparison of straight binary and straight BCD (8-4-2-1) codes. Three decoded decisions about two 4-bit words (A, B) are made and are externally available at three outputs. These decisions fully explain the relative number of bits without external gates. Words of greater length may be compared by cascading comparators in series. The A > B, A < B, and A = B outputs of a stage handling less-significant bits are connected to the corresponding A > B, A < B, and A = B inputs of the next stage handling more-significant bits. The stage handling the least-significant bits must have a high-level voltage applied to the A = B input and additionally for the 'L85, low-level voltages applied to the A > B and A < B inputs. The cascading paths of the '85 and 'S85 are implemented with only a two-gate-level delay to reduce overall comparison times for long words. An alternate method of cascading which further reduces the comparison time is shown in the typical application data.

FUNCTION TABLES

COMPARING INPUTS				CASCADE INPUTS			OUTPUTS		
A3, B3	A2, B2	A1, B1	A0, B0	A > B	A < B	A = B	A > B	A < B	A = B
A3 > B3	X	X	X	X	X	X	H	L	L
A3 < B3	X	X	X	X	X	X	L	H	L
A3 = B3	A2 > B2	X	X	X	X	X	H	L	L
A3 = B3	A2 < B2	X	X	X	X	X	L	H	L
A3 = B3	A2 = B2	A1 > B1	X	X	X	X	H	L	L
A3 = B3	A2 = B2	A1 < B1	X	X	X	X	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 > B0	X	X	X	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 < B0	X	X	X	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	L	L	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	H	L	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	H	L	L	H

'85, 'S85									
A3 = B3	A2 = B2	A1 = B1	A0 = B0	A > B	A < B	A = B	A > B	A < B	A = B
A3 = B3	A2 = B2	A1 = B1	A0 = B0	X	X	H	L	L	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	H	L	L	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	L	H	H	L

'L85									
A3 = B3	A2 = B2	A1 = B1	A0 = B0	A > B	A < B	A = B	A > B	A < B	A = B
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	H	H	L	H	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	L	H	H	L	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	H	H	H	H	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	H	L	H	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	L	L	L	L

H = high level, L = low level, X = irrelevant

TYPED 74S173, 94S173

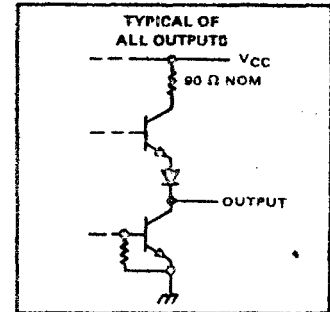
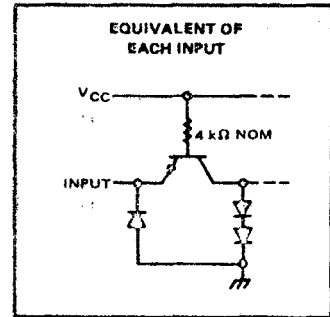
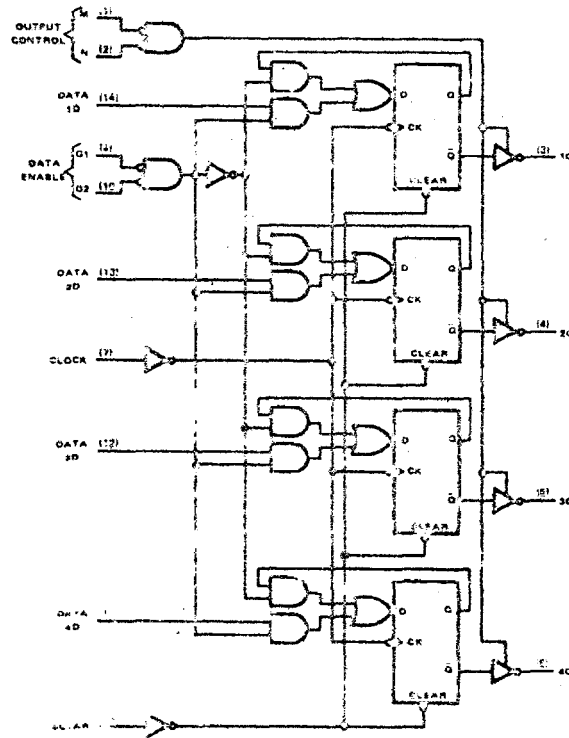
4-BIT D-TYPE REGISTERS WITH 3-STATE OUTPUTS

Switching characteristics, $V_{CC} = 5\text{ V}$, $T_A = 25^\circ\text{C}$, $R_L = 400\ \Omega$

PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
f_{max} Maximum clock frequency	$C_L = 50\text{ pF}$, See Note 4	25	35		MHz
t_{PHL} Propagation delay time, high-to-low-level output from clear input		18	27		ns
t_{PLH} Propagation delay time, low-to-high-level output from clock input		28	43		ns
t_{PHL} Propagation delay time, high-to-low-level output from clock input		19	31		ns
t_{ZH} Output enable time to high level		7	16	30	ns
t_{ZL} Output enable time to low level	$C_L = 5\text{ pF}$, See Note 4	7	21	30	ns
t_{H7} Output disable time from high level		3	5	14	ns
t_{LZ} Output disable time from low level		3	11	20	ns

NOTE 4: Load circuits and voltage waveforms are shown on page 140.

functional block diagram and schematics of inputs and outputs



Clear and input active-low by a transition from a high level to a low level.

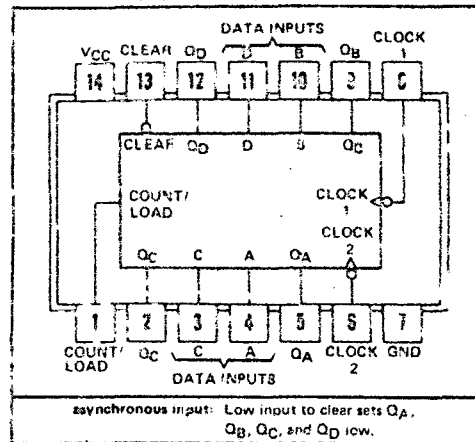
**TYPES SN54196, SN54197, SN54LS196, SN54LS197,
SN74196, SN74197, SN74LS196, SN74LS197**
50/30-MHz PRESETTABLE DECADE OR BINARY COUNTERS/LATCHES

BULLETIN NO. DLS 7211806, DECEMBER 1972

FOR NON DUAL IN-LINE OR
W FLAT PACKAGE (TOP VIEW)

- Performs BCD, Bi-Quinary, or Binary Counting
- Fully Programmable
- Fully Independent Clear Input
- Input Clamping Diodes Simplify System Design
- Output Q_A Maintains Full Fan-out Capability In Addition to Driving Clock-2 Input

TYPES	GUARANTEED COUNT FREQUENCY		TYPICAL POWER DISSIPATION
	CLOCK 1	CLOCK 2	
'196, '197	0-50 MHz	0-25 MHz	240 mW
'LS196, 'LS197	0-30 MHz	0-15 MHz	100 mW



description

These high-speed monolithic counters consist of four d-c coupled, master-slave flip-flops which are internally interconnected to provide either a divide-by-two and a divide-by-five counter ('196, 'LS196) or a divide-by-two and a divide-by-eight counter ('197, 'LS197). These four counters are fully programmable; that is, the outputs may be preset to any state by placing a low on the count/load input and entering the desired data at the data inputs. The outputs will change to agree with the data inputs independent of the state of the clocks.

During the count operation, transfer of information to the outputs occurs on the negative-going edge of the clock pulse. These counters feature a direct clear which when taken low sets all outputs low regardless of the states of the clocks.

These counters may also be used as 4-bit latches by using the count/load input as the strobe and entering data at the data inputs. The outputs will directly follow the data inputs when the count/load is low, but will remain unchanged when the count/load is high and the clock inputs are inactive.

All inputs are diode-clamped to minimize transmission-line effects and simplify system design. These circuits are compatible with most TTL and DTL logic families. Typical power dissipation is 240 milliwatts. Series 54 and 54LS circuits are characterized for operation over the full military temperature range of -55°C to 125°C ; Series 74 and 74LS circuits are characterized for operation from 0°C to 70°C .

typical count configurations

'196 and 'LS196 typical count configurations and function tables are the same as those for '176. See page 370.
'197 and 'LS197 typical count configurations and function tables are the same as those for '177. See page 370.

functional block diagram

'196 and 'LS196 functional block diagram is the same as that for '176. See page 371.
'197 and 'LS197 functional block diagram is the same as that for '177. See page 371.

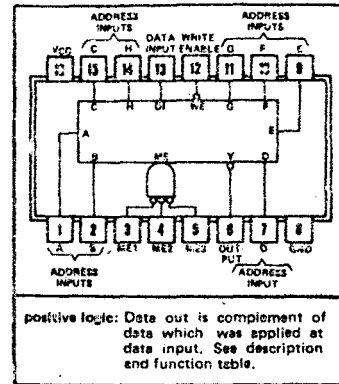
TTL
LSI

TYPE SN74200 256-BIT READ/WRITE MEMORY WITH 3-STATE OUTPUT

BULLETIN NO. DLS 7211735, MAY 1972 · REVISED DECEMBER 1972

- For High-Speed Memory Systems
Access from Memory-Enable Inputs . . . 17 ns Typical
Access from Address Inputs . . . 42 ns Typical
Power Dissipation . . . 1.8 mW/Bit Typical
- For New Designs and/or Military Environments,
SN54S200 and SN74S200 Are Recommended
- Fully Decoded, Organized as 256 Words of One Bit Each
- Compatible with Most TTL and DTL Logic Circuits
- Multiple Memory-Enable Inputs to Minimize
External Decoding

JORN
DUAL-IN-LINE PACKAGE (TOP VIEW)



description

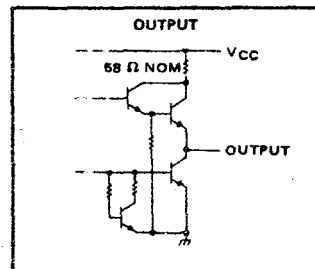
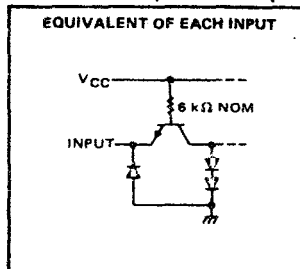
This 256-bit active-element memory is a monolithic transistor-transistor logic (TTL) array organized as 256 words of one bit each. It is fully decoded and has three gated memory-enable inputs to simplify decoding required to achieve the desired system organization. The SN74200 features a three-state output and is functionally equivalent to the SN74S200.

The drive capability of the three-state bus connectable output permits expansion up to 66,304 words of N-bits without additional buffering. See the table below.

WORD CAPACITY
vs
SERIES 54/74 TTL LOADS

SERIES 54/74 TTL LOADS	MAXIMUM NUMBER OF COMMON OUTPUTS	MAXIMUM NUMBER OF WORDS
1	259	66,304
2	220	56,320
3	180	46,080
4	140	35,840
5	100	25,600
6	60	15,360
7	20	5,120

schematics of inputs and outputs



write cycle

The complement of the information at the data input is written into the selected location when all memory-enable inputs and write-enable input are low. While the write-enable input is low, the output is in the high-impedance state. When a number of outputs are bus-connected, this high-impedance output state will neither load nor drive the bus line, but it will allow the bus line to be driven by another active output or a passive pull-up if desired.

read cycle

The stored information (complement of information applied at the data input during the write cycle) is available at the output when the write-enable input is high and the three memory-enable inputs are low. When any one of the memory enable inputs is high, the output will be in the high-impedance state.

FUNCTION TABLE

FUNCTION	INPUTS		OUTPUT
	MEMORY ENABLE [†]	WRITE ENABLE	
Write (Store Complement of Data)	L	L	High Impedance
Read	L	H	Stored Data
Inhibit	H	X	High Impedance

H = high level, L = low level, X = irrelevant
[†]For memory enable: L = all ME inputs low,
 H = one or more ME input high.