DESIGN AND DEVELOPMENT OF A DATA TRANSMISSION LINK BETWEEN MAINFRAME COMPUTER AND A MICROCOMPUTER

By

Mallick Shameem Ahsan

A Thesis

Submitted to the Department of Computer Engineering, Bangladesh University of Engineering and Technology, Dhaka, in partial fulfilment of the requirements for

the degree of

MASTER OF SCIENCE IN COMPUTER ENGINEERING

JULY, 1986

TO

MY PARENTS

DECLARATION

I do hereby declare that neither this thesis nor any part thereof has been submitted or is being concurrently submitted in candidature for any degree at any other University.

Shameem Ahsan

(Mallick Shameem Ahsan)
Candidate

CERTIFICATE OF RESEARCH

Certified that the work presented in this Thesis is the result of the investigation carried out by the candidate under the supervision of Dr. Syed Mahbubur Rahman at the Department of Computer Engineering, BUET,Dhaka.

Shameem Ahsan

(Mallick Shameem Ahsan)
Candidate

SM Rahman 27.7.86

(Dr. Syed Mahbubur Rahman)
Supervisor

ABSTRACT

Transmission of data between mainframe and personal computer plays a very important role in modern data processing. In an attempt to establish a data transmission link from mainframe computer to personal computer the present work studied the 3270 Information Display System, its communication protocol specially from the downloading point of view, the information content of the signal transmitted from Control Unit and display device and designed a data acquisition hardware and a software. The experimental set up developed was used at the Computer Centre of Bangladesh University of Engineering and Technology to download data from IBM System/370 to personal computer. The hardware was designed to work with the 3270 Information Display System and since the 3270 System is attachable to System/370, 30XX and 4300 processors of IBM, the hardware and software will be able to download data from all these mainframe computers operating in any environment to personal computers.

Accepted as satisfactory for partial fulfilment of the requirements for the degree of M.Sc. Engineering in Computer Engineering

BOARD OF EXAMINERS

27.7.86

(Dr. Syed Mahbubur Rahman) — Chairman and Supervisor
Associate Professor
Department of Computer Engineering
BUET, Dhaka.

27.7.86

(Dr.A.K.M.Mahfuzur Rahman Khan) — Member
Professor and Head
Department of Computer Engineering
BUET, Dhaka.

---Ahmed 27.7.86

(Dr. Shamsuddin Ahmed) — Member
Professor & Dean
Faculty of Electrical & Electronic
Engineering, BUET, Dhaka.

---27.7.86

(Professor A. M. Patwari) — Member (External)
Vice-Chancellor
Bangladesh University of Engineering
and Technology, Dhaka.

ACKNOWLEDGMENT

The author expresses his deepest gratitude to Dr. Syed Mahbubur Rahman, Associate Professor, Department of Computer Engineering for his constant guidance and supervision to prepare the work.

He expresses his sincere gratitude to Dr. Abdul Matin Patwari, Vice-Chancellor, for his keen interest and encouragement for the development of research works in the field. He also expresses his gratitude to Dr. Shamsuddin Ahmed, Dean, Faculty of EEE, Dr. A.K.M. Mahfuzur Rahman Khan, Prof. and Head, Dept of Computer Engineering and Dr. J. R. Choudhury, Professor of Civil Engineering and Director, Computer Centre, BUET for their special interest, constant encouragement and advices to complete this work.

The author thankfully acknowledges the valuable suggestions and advices received from Mr. Dulal Chandra Kar, Assistant Professor, Dept. of Computer Engineering and Mr. Sanjoy Kumar Poddar of Computer Centre, BUET.

The author is also thankful to the personnels of BUET Computer Centre for their cooperation and Mr. Abu Taher for excellent typing.

CONTENTS

Chapter 1

INTRODUCTION

1.1 GENERAL

During the first decade of its development, Computer technology was concerned with single isolated computers. People brought their problems to the machines and carried away their results. Then experiments were tried in which two computers interacted and computers were accessed from a distance. These were the beginnings of computer links, the potential is still being explored.

Modern information technology has been paced by the improvement of storage medium and all the big developments of information systems depend on a storage hierarchy stretching from small stores resembling logic to big stores, which can cost less than writing data on paper. Instead of being restricted to calculation, or the storing of valuable data, information systems can now be used for the general run of information storage and retrieval. The methods by which data are captured for the computer and disseminated to the end users can determine the cost of the operation. For this reason the transfer of data from one system to another is an important trend and modern computers are increasingly being connected to communication

networks. Another trend which can now be seen is the connection of two computers by a data network so that they can collaborate in a task. Systems are now in operation in which one computer depends on another for its backing store, its fast processing or a range of output devices.

1.2 SYSTEMS AND NETWORKS - LEVELS OF COMMUNICATIONS

A multicomputer complex may be classified according to how communication is handled, both inside and outside the complex[11]. Within such a complex there is a wide spectrum of possible means of communication. If the complex is geographically centralized, control may still be distributed or else it too may be centralized. If the complex is strongly coupled by its intercommunications it is here called a system; if it is loosely coupled it is called a network[11].

The intercommunications within a computer structure generally involve sharing of common memory, or sharing of control signals and communication paths or both. There is obviously no fine line between systems and networks. For example, an online interactive computer complex with multiple processors and remote communication links may be termed as a system from the view of the computer structure and as a network from the view point of communication[4].

In discussing computer communication networks, it is desirable to specify the relative roles of computers and communication among different computer systems, which link these systems into a computing network more powerful than any of the systems. It is also possible to consider

computers used as a part of communication network, as for example in store and forward communication.

In general, a single network may take on both views simultaneously, serving as computing and communication network. For present purpose, therefore neither of these two views is adopted as the dominant view.

1.3 THE DEVELOPMENT OF DATA COMMUNICATION LINK

The earliest data communication links were designed simply to connect a number of terminals to a single computer. As the use of terminal access networks increased it began to be appreciated that there would be advantages in allowing computers to communicate with other computers as well as groups of terminals. Connecting equipment and software sometimes referred to as interface elements, are used to bridge the different physical and operating environments that exists between input/output (I/O) devices and central processors. And a variety of data transmission channels are available to carry data from one location to another. The advent of microcomputers popularly known as personal computers has opened new horizons of computerization. As people are becoming more and more inclined to personal computers the data communication link between personal computers and between personal computer and mainframe have become a common urge of the computer users. Because of the popularity of personal computers different manufacturers from all over the world have come forward to present various types of I/O devices for personal computers. On the contrary, personal computers were not designed to handle complex problems that require

millions of operations of various natures. An ideal solution to this problem is using mainframe computers and monsters to do complex calculations and then down loading data to personal computers for further analysis, graphical representation and so on. The operating environment of mainframes in the developed countries is different from that of developing countries like Bangladesh. The type and characteristic of interface hardware and software required to establish data communication link between mainframe and personal computer solely depend on the operating environment. Therefore, the interface kit required to establish a data communication link between personal computers and mainframes operating in an environment like that of the installation of our country is not available in the market at all.

1.4 PRESENT STATUS OF DATA COMMUNICATION LINK BETWEEN MAINFRAME COMPUTERS AND PERSONAL COMPUTERS IN BANGLADESH

At present there are seven mainframe installations in the country and all these mainframes are IBM. But the number of personal computers installed in the country is quite large and this number is increasing day by day. Out of the seven main frame installations, five have time sharing systems. But there is only one installation which have a data communication link between mainframe and personal computers but their operating environment is similar to that of developed countries.

1.5 BACKGROUND AND OBJECTIVES OF THE RESEARCH

Transmission of data between mainframe and microcomputers plays a very important role in modern data processing. Researchers are still working to make the transmission easier but all these works are done for an operating environment called virtual machine environment which requires expensive operating systems and large storage (real as well as virtual). But almost all the computer installations of Bangladesh, including BUET, are using DOS/VS (Disk Operating System/Virtual Storage) or DOS/VSE (Disk Operating System/Virtual Storage Extended). Unfortunately no research work has been carried out to develop and design a data transmission link between mainframe computers running under DOS/VS or DOS/VSE and microcomputers in the past.

The objectives of this study are to develop:

i) a data transmission link between IBM 370/115 running under DOS/VS and IBM personal computer

ii) a data transmission link between IBM 4331 running under DOS/VSE and IBM personal computer.

Both of these mainframes are installed at the Computer Centre of BUET.

Chapter 2

PRESENT SYSTEM SURVEY

2.1 SYSTEM/370 ARCHITECTURE

2.1.1 Basic System Structure

The IBM family of computers is an extension and logical outgrowth of System/360 computers of the same manufacturer. The design philosophy with both families of computers is to have a computer architecture that appears essentially same to the user. From the user's point of view, the System/370 can be regarded as a single machine with the differences between models being a function of engineering implementation[8]. The modular design of System/370 makes the computer appropriate for a wide range of applications: scientific, business, real time to name only a few. Functional units such as input/output channels and devices can be added to the system to meet the needs of a particular application.

2.1.1.1 System Organization

A System/370 model is composed of five different types of functional units: storage, a central processing unit, input/output (I/O) channels, control units and input/output (I/O) devices. As shown in figure 2.1 the input/output units are attached to input/output channels through control

units. Although the operation of the entire system is controlled by the central processing unit, it shares access to storage with the input/output channels, allowing input/output and computing to overlap in time to some degree. As with most modern computers, instructions and data are held in storage for use by the central processing unit. Information can be stored on a permanent or temporary basis with a complete range of input/output and mass storage devices, including tape, cards and direct-access storage units.

A simplified data flow of System/370 model 115 (processor 3115-2) is shown in figure 2.2.

2.1.1.2 Data Representation and Addressing

The basic unit of addressable main storage in System/370 is the 8-bit byte, which can represent one character, two decimal digits or eight bit binary data. Bytes may be addressed separately or be grouped to form words or variable-length fields. Different sized words are permitted. A half word is formed by two consecutive bytes and is the basic building block of instructions. Fixed point arithmetic is also permitted on half words. A word is composed of four consecutive bytes i.e. 32 bits and can be used for fixed-point and floating-point arithmetic. Floating point is also available with double words which is composed of eight consecutive bytes and is extended on some models of System/370 to extended-precision words (16 bytes).

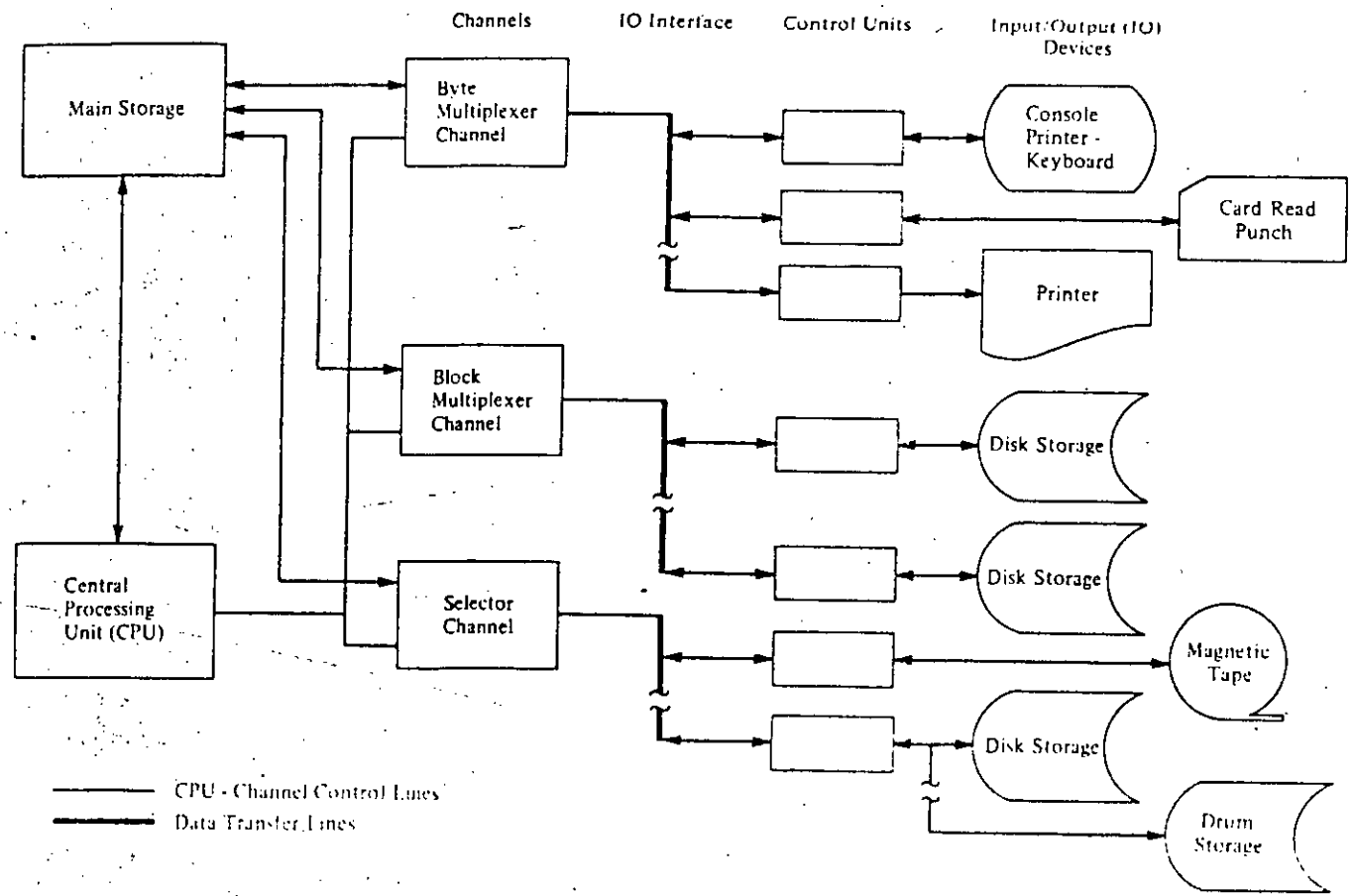


Figure 2.1 : System/370 Organization

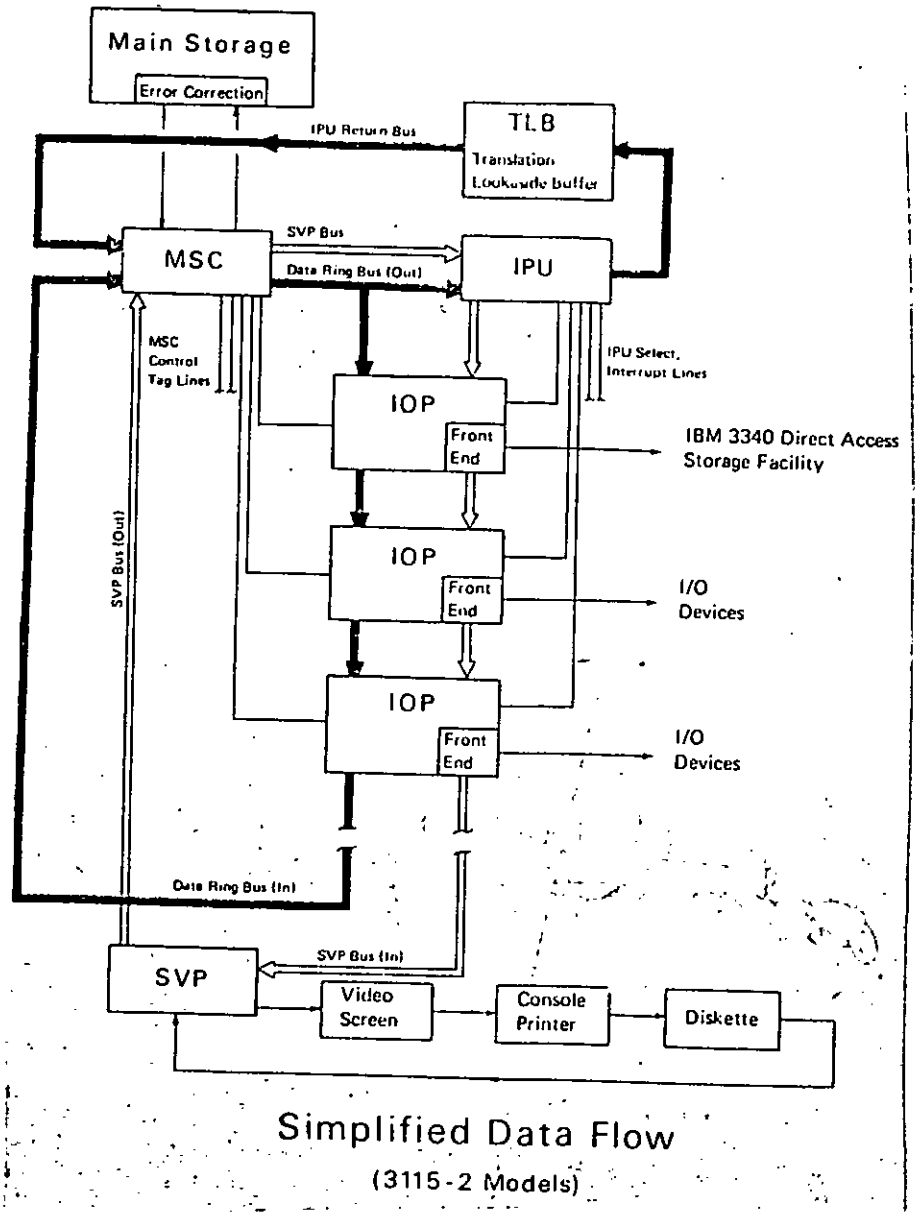


Figure 2.2: System/370-115 Data Flow

Storage addresses within the computer are represented as binary integers starting at zero and the location in main storage of any field or group of bytes is specified by the leftmost byte.

Character information is conveniently represented as 8 bit units of information allowing 2^8 (256) possible characters. System/370 uses the Extended Binary Coded Decimal Interchange Code (EBCDIC) shown in appendix - I. The value of the contents of a byte location is also conveniently denoted by two hexadecimal digits - each requiring 4 bits. For example, the letter A can be denoted by the hexadecimal number C1 and a dump of main storage display the contents of each byte location as a pair of hexadecimal digits.

2.1.1.3 Data Formats

Numeric data may be stored in four different formats: Zone decimal, packed decimal, fixed point binary and floating point binary. Computer instructions are available for conversion between zone decimal and packed decimal and between packed decimal and fixed point binary.

Zone decimal number format is available for use with EBCDIC input and output and permits a sign in the low order position of the field. Zoned decimal is depicted in figure 2.3.

```
High Order      Low Order
   Byte            Byte

-------------------------- - - - - --------------
|Zone |Digit |Zone |Digit          |Sign |Digit |
-------------------------- - - - - --------------

Representation of +1234

-------------------------------------------------------
|1111 | 0001 | 1111 | 0010  | 1111 | 0011 | 1100 |0100|
-------------------------------------------------------
```

Figure 2.3 : Zoned decimal number format

```
High Order      Low Order
   Byte            Byte

-------------------------- - - - - --------------
|Digit|Digit |Digit|Digit          |Digit|Sign  |
-------------------------- - - - - --------------

Representation of +1234

-----------------------------------------
|0000 | 0001 | 0010 | 0011  | 0100 | 1100|
-----------------------------------------
```

Figure 2.4 : Packed decimal number format

Packed decimal format allows efficient use of computer storage for commercial applications and permits arithmetic computation without conversion to a fixed point or floating point representation. The packed decimal format is shown in figure 2.4. A variable field length is permitted up to 16 bytes which is equivalent to 31 digits and a sign. The numbers are treated as signed integers and a negative integer is carried in true form.

Fixed-point binary numbers can be represented as half words and full words as shown in figure 2.5. The format allows 15 and 31 bits of precision respectively. If the sign bit is 0, the number is positive. If the sign bit is 1 the number is treated as a negative integer stored in 2's compliment form.

```
          Half Word
 ------------------------
 | S |   Integer        |
 ------------------------
   0   1              15

          Full Word
 ------------------------------------------------
 | S |                Integer                   |
 ------------------------------------------------
   0   1                                       31
```

Figure 2.5 Fixed point binary format

Short Form

S	Exponent	Fraction
0 1	7 8	31

Long Floating-point

S	Exponent	Fraction
0 1	7 8	63

Extended Precision Form

S	Exponent	Fraction			Fraction
0 1	7 8	63	0 1	7 8	63

Figure 2.6 Floating-point number formats

The architecture of System/370 includes three formats for floating point binary data: short, long and extended. All formats are shown in figure 2.6. The short form uses a full word and is equivalent to seven decimal places of precision. The long form uses a double word and provides 17 decimal places of precision. The fraction of a floating point number is regarded as hexadecimal digits with the

radix point immediately to the left of the high-order digit of the fraction. The exponent portion of the number represents a power of 16, in excess 64 forms, that should be considered as a multiplier of the fraction. The exponent provides a range of -64 through +63 which permits decimal numbers to be represented in the range of 10^{-78} to 10^{75}. A negative number is carried in true form where a 0 sign represents plus and 1 represents minus. The practice of regarding the exponent as a power of 16 allows relatively large range of values to be represented with a relatively short (only 7 bits) exponent. An extended-precision floating point number consists of two long precision floating point numbers, called high order and low order parts. The fraction of the low-order part is considered an extension to the fraction of the high-order part. The high order part carries the true exponent and the true sign. The exponent of the low order part is always 14 less than the exponent of the high order part. An extended precision floating number can be used to carry approximately 34 decimal places of accuracy.

2.1.1.4 Local Storage

Local storage in System/370 includes 16 general registers and 4 floating point registers (Fig.2.7). Each general register has a capacity of a full word and can be used for arithmetic and logical operations and in address arithmetic and addressing. The general registers which are also termed as general purpose registers are numbered 0 through 15 and are identified by a 4 bit R field in a computer instruction. The floating-point registers are used for floating-point operations are identified by the numbers

R Field	Reg. Number	Control Registers (32 Bits)	General Registers (32 Bits)	Floating-Point Registers (64 Bits)
0000	0			
0001	1			
0010	2			
0011	3			
0100	4			
0101	5			
0110	6			
0111	7			
1000	8			
1001	9			
1010	10			
1011	11			
1100	12			
1101	13			
1110	14			
1111	15			

Note: The braces indicate that the two registers may be coupled as a double-register pair, designated by the R field of the lower-numbered register. For example, the general register pair 0 and 1 is designated by the R field of register 0.

2.7 : Local Storage in System/370

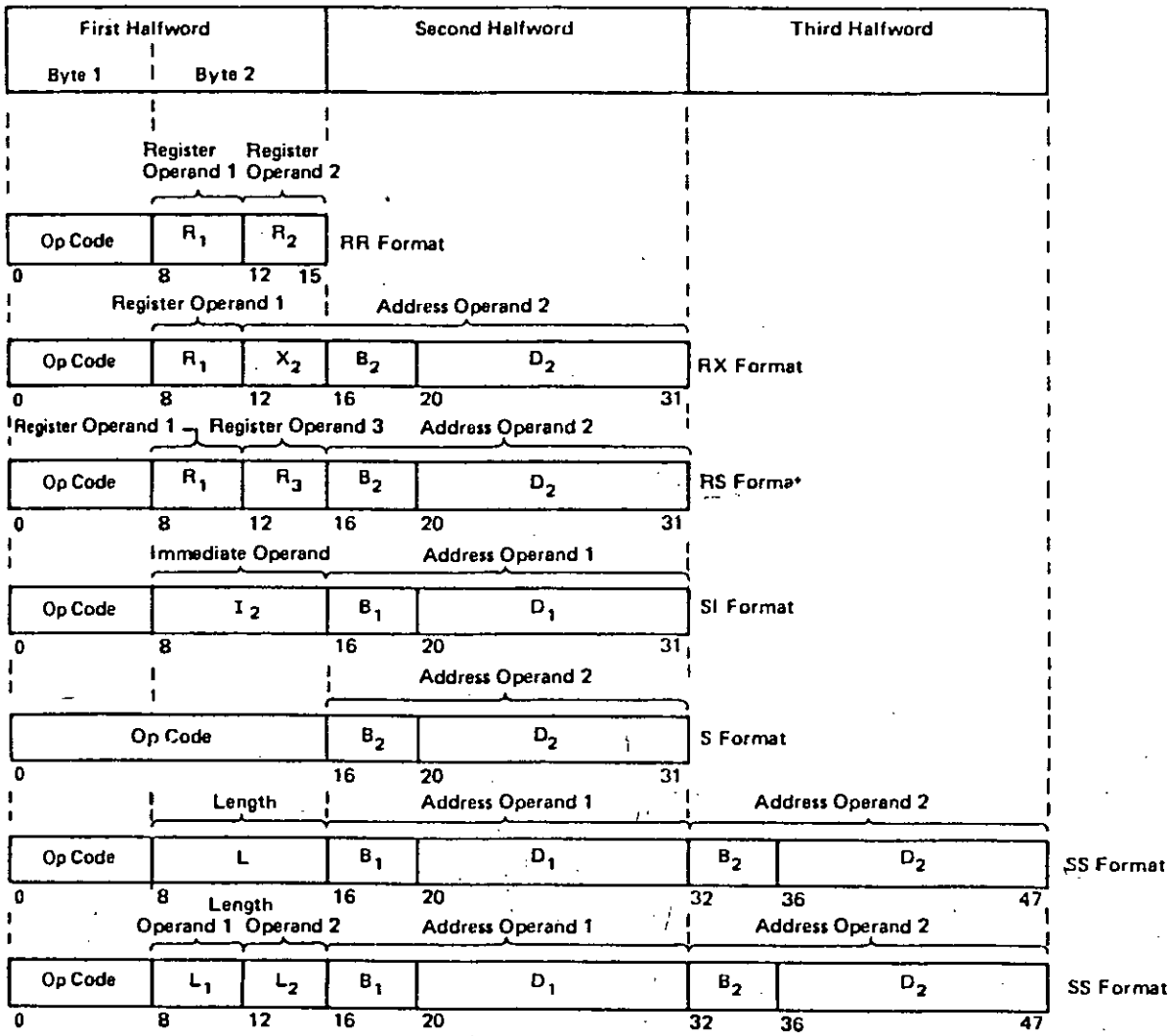
0, 2, 4 and 6. Each floating point is a double word (64 bits) in length and can hold either a short, long or a extended precision floating point number.

2.1.1.5 Instruction Format

In System/370, computer instructions use five different formats that are generally related to the amount of information necessary to specify a given operation. The length of an instruction format can be one, two or three half words as given in figure 2.8. The five formats are denoted by the codes RR, RX, RS, SI and SS. RR denotes a register to register operation; RX denotes a register and indexed storage operation; RS denotes a register and storage operation; SI denotes a storage and immediate operand which refers to a data value that is held in the instruction itself, operation; and SS denotes a storage-to-storage operation. The components of a computer instruction are denoted by letters, such as R for register and B for Base. When instructions are described, the letters are subscripted to denote the manner in which the operands participate in the instruction.

2.1.1.6 System Operation

System/370 operates by fetching instructions from main storage, decoding them, and initiating their execution by an appropriate function unit. A general schematic of Central Processing Unit (CPU) operation is shown in figure 2.9. It is seen from the diagram that floating-point arithmetic operations use the 4 floating-point registers, and the 16 general purpose registers are used for fixed-



Six Basic Instruction Formats

Figure 2.8 : System/370 Instruction Formats

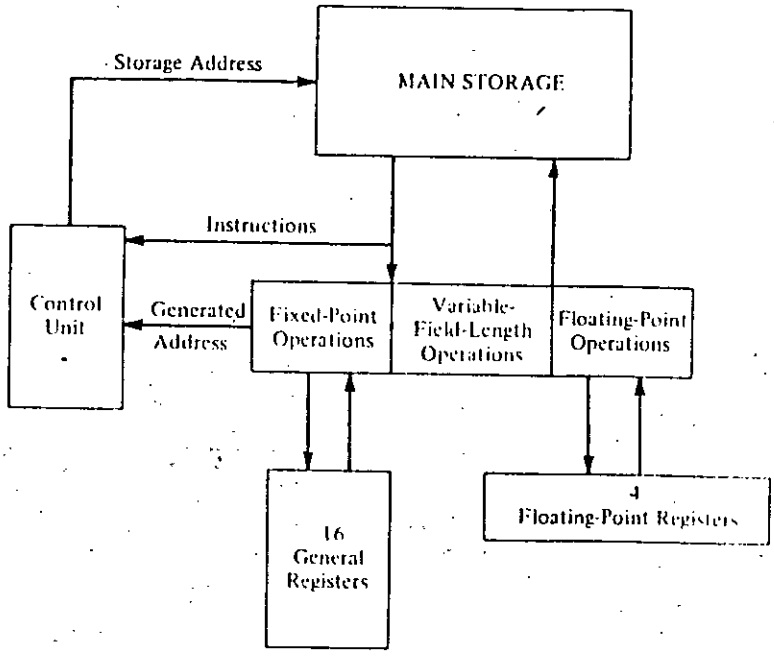


Figure 2.9: Central Processing Unit Operation: System/370

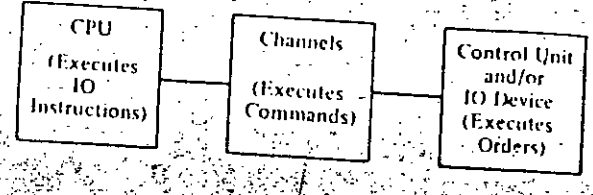


Figure 2.10: Relationship of I/O Instruction System/370

point arithmetic operations and for address arithmetic. The execution of instructions proceeds sequentially until either a branch instruction is executed or an interrupt is received by the CPU; in either case, sequential execution is suspended and execution proceeds by fetching the next instruction from a specified location in main storage.

Although input/output (I/O) operations are initiated by the CPU, I/O transfer and control proceeds a synchronously under control of commands and orders. A command controls the operation of an I/O channel and specifies a main storage address and a byte count for a data transfer operation or control information for a control operation. An order is particular to a particular I/O control unit or device and specifies functions such as rewinding a tape unit skipping a card on a card reader. The relationship of I/O instructions, commands and orders in given in figure 2.10.

The entire computer system runs under the direction of collection of programs called an operating system[10]. The operating system effectively manages computer system resources and schedules programs for execution which are stored in machine-readable form on a mass storage or I/O device[10].

2.1.2 System Control Functions

System/370 satisfies the need for a computer that can supervise itself without manual intervention over a wide range of applications. The design of the computer allows several distinct programs, operating concurrently to share

the resources of the central processing unit and main storage - an operating philosophy classified as multiprogramming or as time sharing. Similarly, the input/output channel concept allows input and output to proceed independently of program execution. All the preceding facilities imply a certain amount of control over system operation.

Operating control of the System/370 computer is inherent in a double word called the program status word (PSW). Whenever a System/370 computer is operating there is a current PSW held in a register internal to the central processing unit . PSW denotes the state of the computer by indicating the following information: interruptions that can occur, program states, condition and mask bits, interruption codes and the current instruction address. Thus, the process of storing the current PSW would preserve the state of the system at the point in time for further inspection[8]. Moreover by loading a new PSW, the state of the system is changed.

The design structure of System/370 provides sixteen 32-bit control registers that augment the PSW in controlling the operation of the computer. The registers are not part of addressable storage but can be loaded and stored by appropriate computer instructions that are executable in the supervisor state. Only five of the control registers are currently used (0, 2, 8, 14 and 15) with the remainder being reserved for further expansion. The information stored in the control registers is listed in appendix - II.

The execution of a program is affected by the states that determine the overall CPU status. Four distinct alternatives are possible: Problem/Supervisor state, Wait/Running state, Masked/Interruptible state and the Stopped/operating state.

In the supervisor state all computer instructions are valid - including I/O instructions, protection instructions, and status changing instructions. I/O, protection and status changing instructions are classed as privileged instructions. When the CPU is in the problem state only non-privileged instructions can be executed. The problem state is normally used for the execution of user programs.

The wait state is usually entered when awaiting an interruption by loading a new PSW. In the wait state,no instructions are processed but the timer continues to run. At the end of the wait state processing of instruction proceeds in the normal manner.

When the CPU can be interrupted for a particular interruption that interruption is said to be enabled and a mask bit in either in the PSW or one of the control registers is set to one when the mask bit is zero the interrupt will not occur and the interruption is said to be masked off or disabled.

The CPU can also be either in the stopped or in the operating state. In the stopped state, instructions are not executed and interruptions cannot take place; in the operating state, the system is either waiting or running,

as determined by the wait state bit in the PSW, and interruptions can take place. The stopped and operating states can be entered only manual intervention at the system control panel or by a machine malfunction.

The interruption system permits the System/370 to respond to external conditions, to requests for system monitoring or service, and to error conditions on a dynamic basis. As a result of an interruption, the CPU can change state and normal execution proceeds from a specified location. By setting an appropriate bit in the PSW or a control register, an interruption can be enabled or disabled. A disabled interruption may be ignored by the CPU or be left pending, depending upon the type of interruption. A pending interruption is taken when that type of interruption is enabled. An interruption takes place after the execution of one instruction is finished and before the execution of a new instruction is started. Moreover, an interruption is accepted even if it is enabled for a single instruction execution.

Five types of interruption are possible: input/output, program, supervisor call, external and machine check. Associated with each type of interruption are two fixed locations in main storage - each of which can hold a double word. In the first double word CPU automatically stores the current PSW at the instant of interruption; that double word is called the old PSW. The second double word is called the new PSW and the CPU automatically makes the double word the current PSW. The process of switching of PSWs is depicted in figure 2.11. The instruction address field of the new PSW would naturally contain the address of

the first instruction of a sequence of instructions designed to process that type of interruption. Clearly, the state of the CPU at the time of interruption is stored in the old PSW location for subsequent inspection.

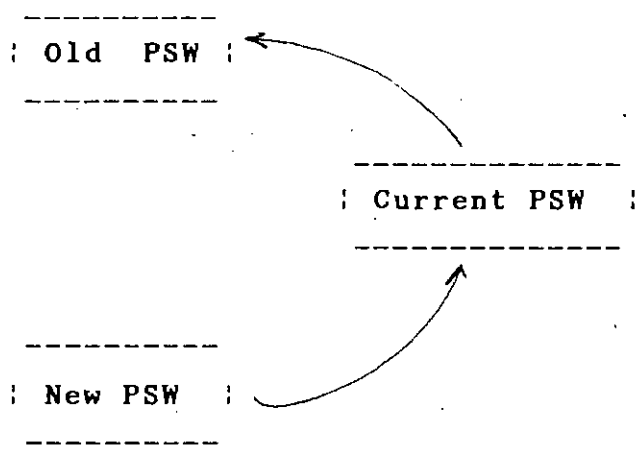


Figure 2.11 : Switching of PSW

An input/output interruption signals to the CPU that an I/O channel is free, that a specific channel or control unit activity has been completed or that a special condition has arisen. The address of the I/O channel and control unit associated with the interruption are stored in bits 16-31 of the old PSW. Additional information on the nature of the interruption is saved in a status word that is stored when the interruption is accepted by the CPU.

A program interruption arises from an improper specifications of computer facilities, an illegal use of an

instruction or data, or a request for system monitoring. Sixteen possible conditions are listed in appendix - III. Of these 16 conditions 4 can be masked off by setting the respective bits in the PSW. The other interruptions cannot be masked off. The system monitoring feature is also implemented through the program interruption system.

The major purpose of the supervisor call interruption is to provide a means of switching from the problem state to the supervisor state. The supervisor call interruption is initiated by a supervisor call instruction (SVC) in a problem program. Thus information may be passed between the calling program and a supervisor program.

The external interruptions allow the CPU to respond to the timer, or an interrupt key on the operator console. The cause of an external interrupt is identified by the bits 24-31 of the old PSW; the interruption code bits are listed in appendix - IV. External interruptions are controlled by bit 7 of the PSW and an appropriate bit in control register zero.

A machine check interruption (MCI) results from a system malfunction and is classified as either hard or soft. A hard MCI is caused by a non-recoverable machine error and results in an immediate termination of CPU activity and a logout of machine status if PSW bit 13 is set to one. A soft MCI is caused by a recoverable error and results in normal interruption processing. Both types of interruptions may be held pending. The circuitry of System/370 provides for error checking and correction (ECC) related to single bit storage error. Other types of errors

are detected but not corrected automatically by the circuitry. Machine check processing is necessarily model-dependent and differs in the amount of permanent storage required.

The PSW contains a two-bit field called condition code (CC) that is set by the majority of arithmetic and logical operations. The two bits provide four settings: 0, 1, 2 and 3. The meaning of the various settings is determined by the operation that sets it. For example, a fixed-point add operation generates the following conditions:

Condition Code	Result
0	Sum is zero
1	Sum is less than zero
2	Sum is greater than zero
3	Overflow

Similarly a floating point compare operation would yield one of these conditions:

Condition Code	Result
0	Operands are equal
1	First operand is low
2	First operand is high
3	Undefined

The condition code is stored as a part of the old PSW and a new condition code is loaded with a new PSW in status

switching or with the set program mask instruction. During normal program execution, the condition code is used by the branch instruction to alter the normal sequential execution of instructions.

The high performance nature of System/370 and its suitability for use in multiprogramming and time-sharing environments have created a need for a feature that provides a means of selectively recording information during the execution of the system. The monitor call instruction is available that designates one of sixteen monitoring classes and initiates a program interruption when executed by the CPU. A program is monitored by inserting monitor call instructions at selected locations in that program. The program interruption generated by the monitor call is usually serviced by a monitoring routine that records a monitor code supplied with instruction.

The monitor call instruction is of the storage immediate type (SI) and provides a monitor class and a monitor code. The instruction operates in conjunction with mask bits of control register 8. The mask bit for a monitor class denotes that events of that type are being monitored at that point in time. Several mask bits may be set at one time. When a monitor call instruction is executed by the CPU, the mask bit in control register 8 specified by the monitor class field in the instruction is inspected.

Each model of System/370 family of computers includes an interval timer and a time-of-day clock. The interval timer is maintained in fixed location in main storage and has a resolution of 3.33 milliseconds. If occupies a 32 bit

word; however only 24 bits are used corresponding to a full cycle of 15.5 hours. The interval timer is counted down at the rate of 60 cycles per second and generates an external interruption (if not masked off) when its value goes from positive to negative.

The time-of-day clock is an internal double word binary counter that provides an accurate measure of elapsed time independent of system activity. The double word is essentially a fixed point number of double precision. The resolution of the clock is one microsecond. The cycle of the clock is approximately 142 years.

2.1.3 Storage design and Addressing

The main storage facilities of the System/370 are designed to support high performance computing over a broad spectrum of applications.

The performance of a main storage unit is dependent upon several important factors: capacity, access widths, cycle time, and degree of interleaving. Performance is also affected by parity and error checking, storage protection, address generation and addressing.

Main storage capacity of System/370 ranges from 96K to 3072K bytes depending on model of the CPU. Access width refers to the amount of information that is transferred between main storage and the CPU with each access[8]. The width is particularly important for operations that allow variable-length and double word operands since a given instruction may require more than one access to storage.

The access width is model dependent and ranges from 4 to 16 bytes. Cycle time refers in general to the time that storage is busy after a reference is made to it. Cycle time is related to the performance index of a given model. Storage is also interleaved with some models of System/370 allowing storage access to be overlapped, in time, depending upon the sequence of references required by a given program.

Data are transferred between main storage and CPU and within the CPU in multiple of 8-bits. CPU registers, buffers, and data transfers carry a ninth parity bit along with each byte. Odd parity is maintained and a machine check is generated whenever a parity error cannot be corrected.

The storage protection facility in System/370 protects against unauthorized access to specified storage blocks. A 5 bit storage key is provided with each 2048 bytes of main storage.

The System/370 uses a base/displacement addressing scheme that allows relative addressing within a 4096-byte block of main storage. Three fields in a computer instruction pertain to main storage addressing: a base register specification (B), an index register specification (X), and a displacement (D). Register specification denotes 16 general purpose registers and are 4 bit in length. The displacement is 12 bits long and allows relative addressing of upto 4096 bytes beyond the base address. Many System/370 instruction designates an index register that is also used in address generation effective main storage address is

calculated by adding the contents of the base register, the contents of the index register (if used), and displacement field - as shown in figure 2-12. Address arithmetic uses the low order 24 bits of the specified registers.

```
 Index  Base   Displacement
 :  X  :  B  :     1230     :
 ----------------------------

          plus B:   4000

          plus X:   0020

                :   5250     :
                ---------------
                  Effective address
```

Figure 2.12 : Address generation

2.1.4 Channel Organization and Input/Output

System/370 computers support a variety of input/output devices. As a result the I/O systems are exceedingly complicated.

Multiprogramming and time-shared operating systems require I/O device independence, asynchronous execution of I/O operations, storage protection for nonoperating programs and the capability of attaching a variety of devices with totally different operating characteristics. Concurrent I/O operation must be supported along with

device-dependent operations when required by a particular application. On the other hand, low speed and control operations should not be permitted to monopolize I/O channel facilities. System/370 satisfies all these requirements with an efficient I/O system that function with a minimum amount of supervision by an operating system control program[8].

Input and output operation are executed by I/O channels, control units and I/O devices operating under control of an I/O control program that executes in the CPU. Most functional units are treated as independent entities by other functional units in the system.

Storage devices, external to the computer are referred to as I/O devices - regardless if they are used for input or output or as a mass storage medium. In other words, the term input and output refer to an operational procedure rather than a functional assignment[8]. Frequently used I/O devices are card equipment, tape units, printers and direct-access-storage (DASD) devices, but may also include optical/visual/audio devices and keyboard units.

An I/O device is attached to an I/O channel through an I/O control unit that contains data buffers and the logical circuitry necessary to operate the device and perform control functions. Information is transferred between the I/O control unit and the I/O channel over a standard connection termed the I/O interface. The interface allows one byte to pass between the two units at one time. The standard I/O interface is common to all channels and to all device types.

The connection between an I/O channel and main storage is not standard and varies from one system/370 model to another. In general, the amount of data passing over the channel/storage interface at one time is dependent upon storage design and the access width.

All I/O operations do not involve the transfer of information; some perform a control function such as tape rewind or a disk seek. A control function is performed by an I/O control unit independently of CPU and other channel operations. A control unit may be shared or nonshared. A nonshared control unit controls only one device. A shared control unit controls several devices - each with same or similar characteristics.

The CPU initiates an I/O operation by specifying the channel and device to be used and by indicating to the channel the main storage location of beginning of a channel program. The channel program specifies the I/O operation that is to take place and exists a series of channel command words (CCWs). A CCW specifies an operation, flags, a count and a storage location. Six I/O operations are defined: read, write, read backward, control, sense and transfer-in-channel.

An I/O channel directs and controls the flow of information between main storage and I/O devices and permits CPU processing to proceed concurrently with I/O operations[8]. An I/O channel is designed to perform a variety of functions. The manner in which these facilities are implemented varies between the different types of

channels of System/370. The channel may be functionally independent unit or may time-share certain facilities with the CPU. Similarly, some types of channels operate differently from others and the method of operation determines the kinds of devices that can be attached to it.

System/370 has three types of I/O channels: byte multiplexer, selector and block multiplexer. Each type of channel is characterized by its mode of operation viz: byte, burst or both and the manner in which it responds to CPU requests and I/O device activity. The byte mode is used with low speed devices, such as card reader, and effectively allows information from different devices to be interleaved as it passes through the channel - thus sharing channel facilities. The burst mode is used with higher-speed devices, such as magnetic tape or disk storage and utilizes all channel facilities during the transfer of a block of data. All three types of channels can operate in burst mode.

2.1.5 3270 Information Display System

The IBM 3270 Information Display System is a family of products that can be tailored to meet the needs of alphameric display applications. The 3270 system offers the user a wide selection of components and configurations. Also available are a large variety of features which improve performance, provide additional operational capability and permit expansion of the display system[15].

Components of the 3270 can be selected from 3270 configuration attachable to System/360, System/370, 4300 processors, 30XX processors as host systems.

The 3270 Information Display System has three basic components: a control unit(CU), a display station and a printer. The control unit provides for the 3270 system's attachment to a data processing system and directs the operation of attached display stations and printers. The display station provides image display of data transmitted from the host system. A display station with an attached keyboard enables the user to enter, modify or delete data on the display and to cause the revised data to be returned to the host system for storage or additional processing. The printer provides printed copy of data displayed at a display station or transmitted from the host system.

A locally attached 3270 Display System using 3272 Control Unit is shown in figure 2.13.

2.1.5.1 Control units (CU)

Each unit in the 3270 system has its own buffer for storing data. Buffers are checked to determine that all characters in the buffers have correct parity. When not executing a command operation, the control units continually perform an internal poll of all attached devices. Internal polling is performed to determine the current device status and whether the device has an I/O pending condition[15].

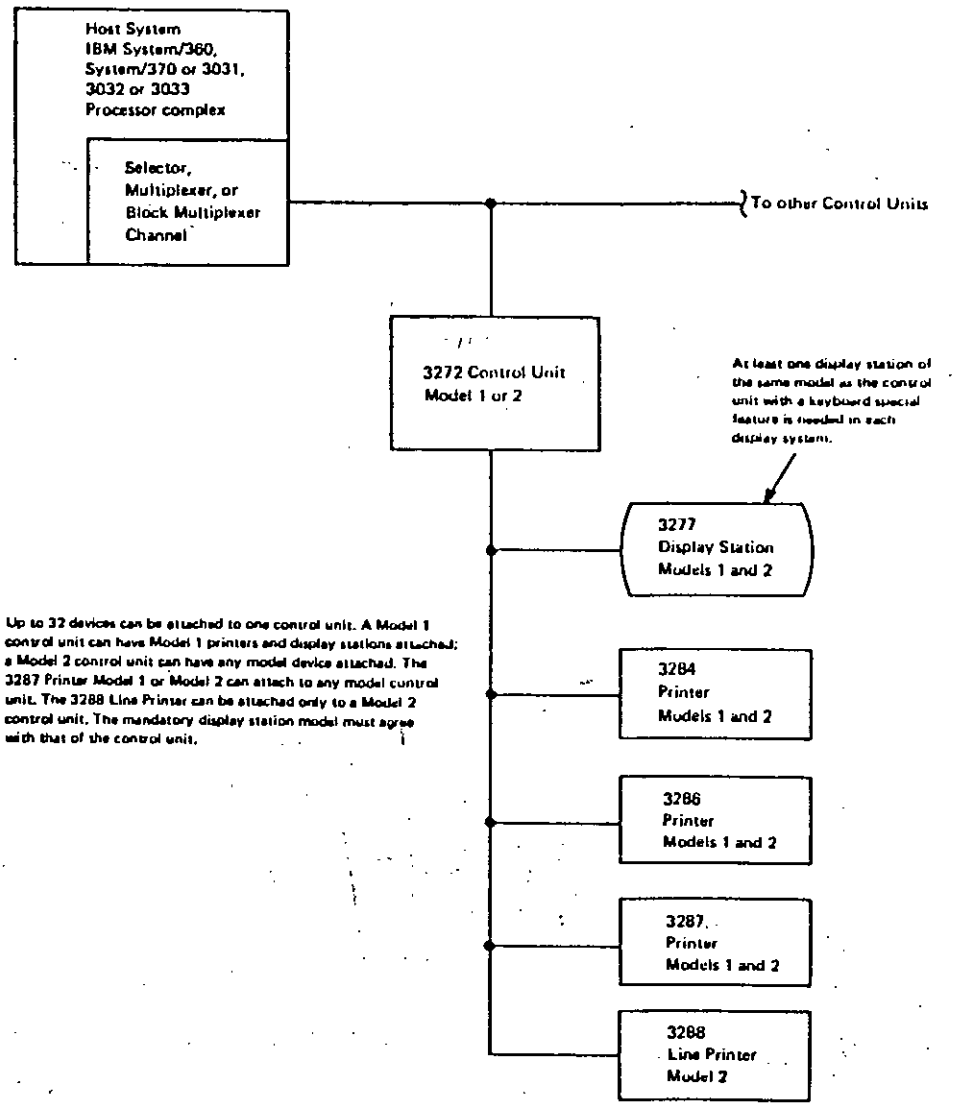


Figure 2.13: Locally attached 3270 Display System

The current status of each device indicates to the control unit whether the device is available, ready, or busy. This information is recorded in the associated device adapter in the control unit.

When an I/O pending condition is detected at a device attached to a 3271 or 3272 controller, polling stops and the control unit communicates solely with that device. When communication is ended, the control unit commences polling at the next sequential device.

Additionally, when the program addresses a specific device, the control unit stops the sequential polling and polls the addressed device to obtain its latest status. If conditions permit, the control unit communicates solely with that device until the operation is completed[15]. At that time sequential polling is resumed.

Figure 2.14 and figure 2.15 show data flow and buffer storage of 3272 control unit.

2.1.5.2 Displays

Display station for the 3270 system are buffered displays. Data displayed on the screen is stored in coded form in a display buffer, the buffer contains as many locations as there are character positions on the screen. The data may be loaded from the system by the application program or from a keyboard attached to the display station.

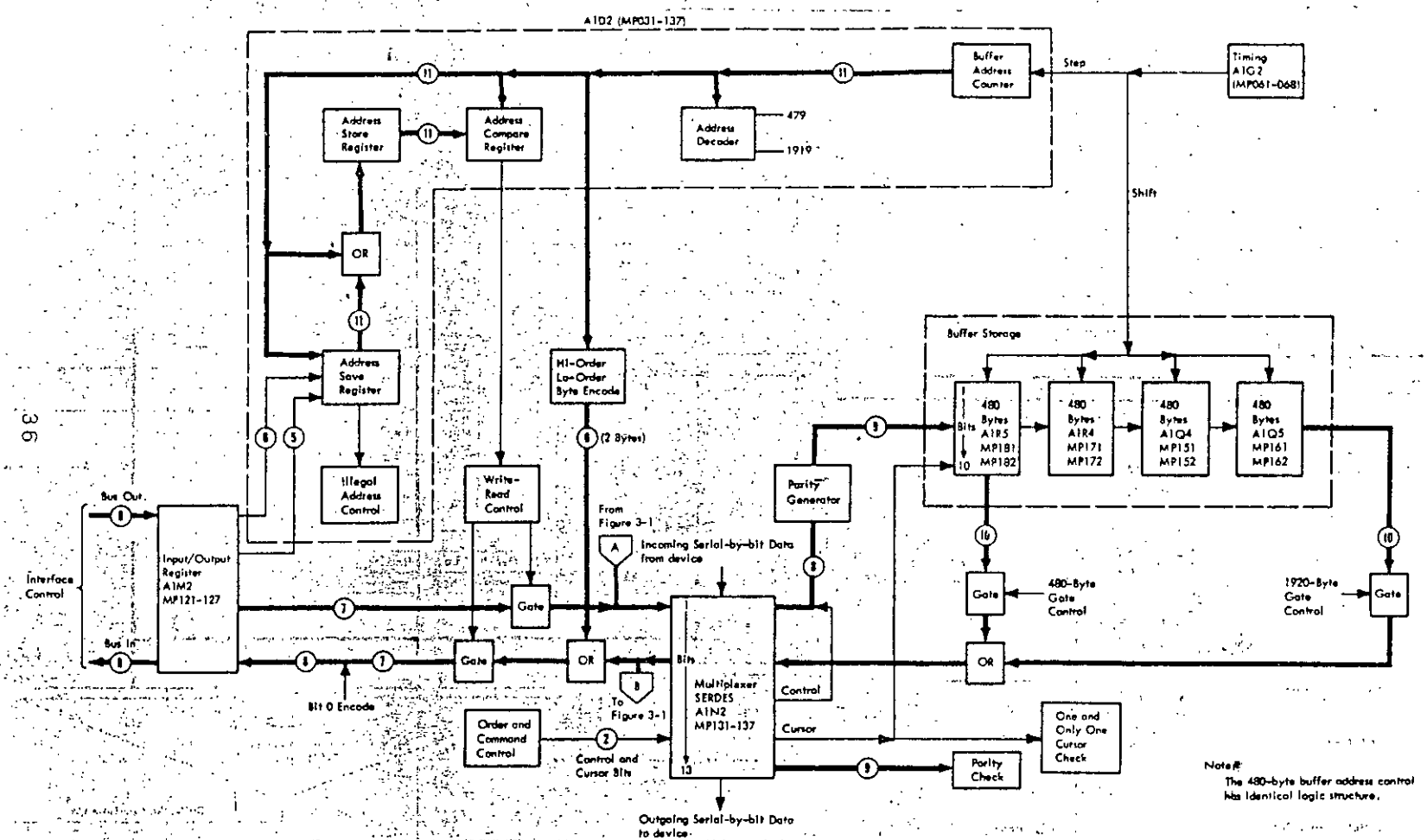


Figure 2.14: 3272 Control Unit Data Flow

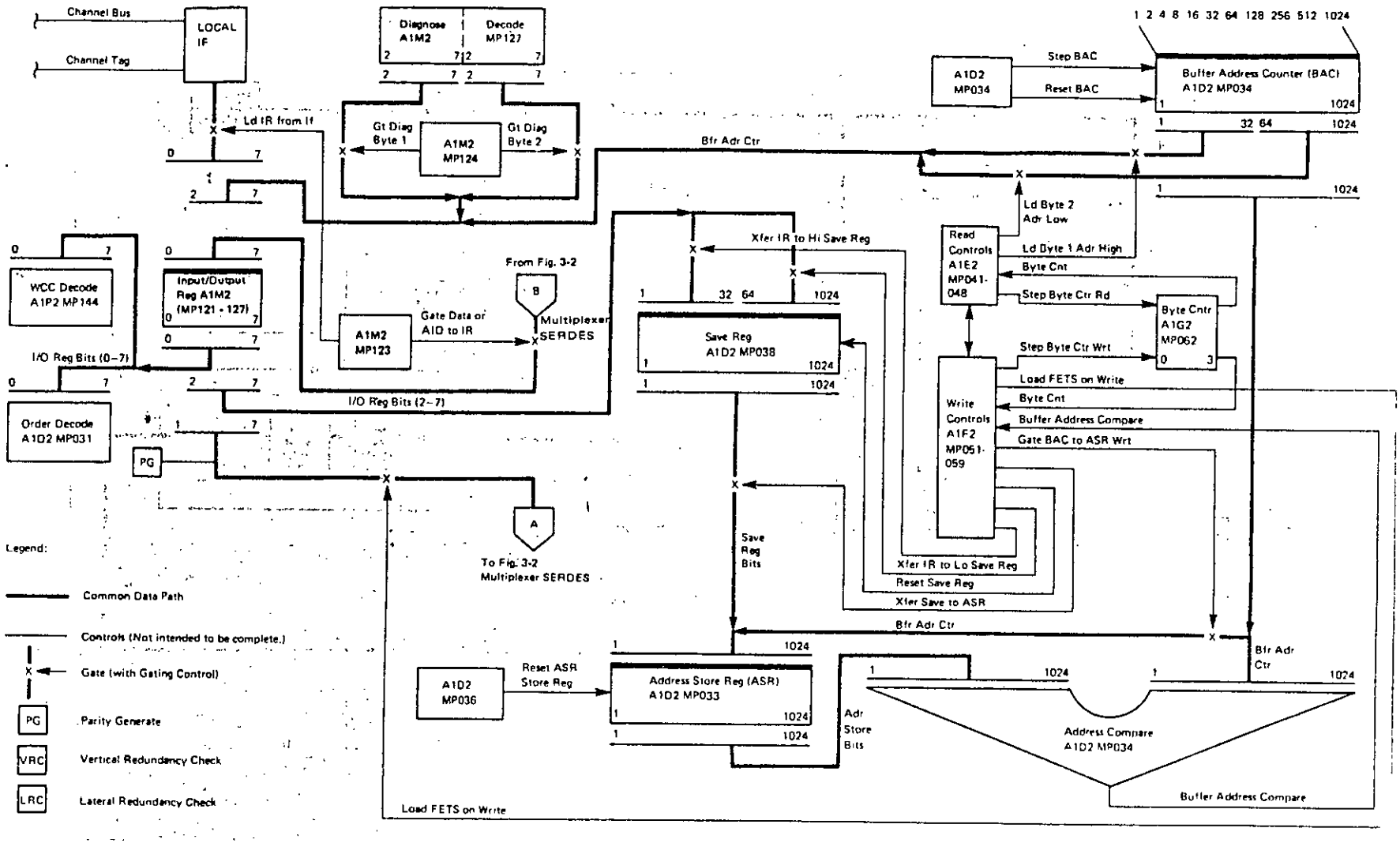


Figure 2.15: 3272 Buffer Storage Address Control

The display image contains a fixed number of horizontal rows with a fixed number of character positions in each row. For example, 3277 displays are 1920 character display which display 24 rows of 80 characters.

Figure 2.16 shows data flow of 3277 display device.

2.1.5.3 Printers

Printers for the 3270 Information Display System provide a printed copy of information that is displayed at a display station or of information written from the program. Printed data appears in the same alphameric characters and symbols that appear on a display and printouts can be formatted in the same manner as a display is formatted. Cursor information is ignored by the printer.

2.2 PC ARCHITECTURE

Like a mainframe a PC is also composed of the five basic parts: arithmetic and logic unit, memory unit, control unit, input unit and output unit. These parts are associated as depicted in figure 2.17. Arithmetic and logic unit is responsible for all the mathematical and logical functions like additions, subtractions, multiplications, divisions, comparisons.

The memory unit is used to store programs, results and other data. There are two types of memories used in PC - RAM (random-access memory) which can be read from and

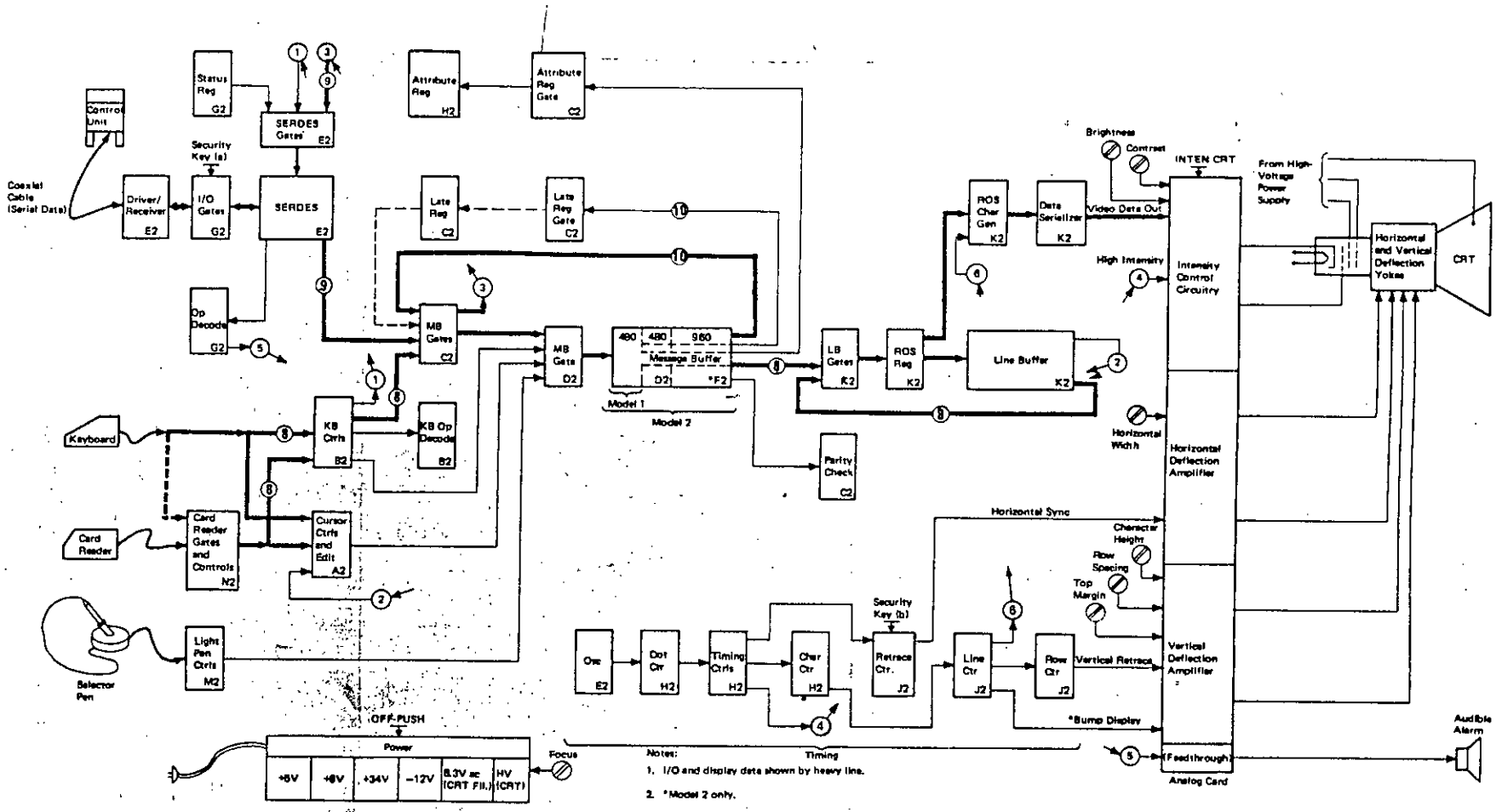


Figure 2.16: 3277 Display Station Data Flow

written to and ROM (read-only memory), which can be read from but not written to.

The input units let user enter information into the computer. Data can be communicated to PC through key-board, light pen, graphic tablet, a mouse or a voice recognition-board and a microphone.

An output unit gets the information from the computer for the user. Typical output devices that are used with PC are printer, plotter, video screen, etc.

The CPU chip used in IBM PC is Intel 8088. With 20 address lines 8088 can address over a million memory locations. It can also operate in maximum mode with a coprocessor, the 8087 numeric data processor, which greatly increases the computational speed.

The 8088 operates at 4.77 MHZ. Since, internally it handles words upto 16 bits long only, the CPU expands its internal address word to 20 bits at its output by segmentation scheme. Memory addresses are logically subdivided into special segments of 64K bytes each. These segments can be allocated to special segment registers within the 8088. Bytes within a segment are addressed using a 16 bit offset address. The 20 bit physical address is constructed inside the 8088 by adding the 16-bit offset address to the 16-bit segment address with the segment address shifted left by one hexadecimal value.

The 8088 CPU on the system board is configured with special support chips. These chips work hand-in-hand with

the 8088 CPU. These are 8284 clock generator, 8259 programmable interrupt controller, 8255 programmable peripheral interface, 8253 programmable interval timer.

Control signals, addresses and data are shared between the CPU and the rest of the PC system over tiny parallel lines or traces on the mother board called buses. A bus is like a roadway over which the 8088 CPU communicates with other components (peripherals such as disk drives) and the outside world. IBM PC has an advanced bus design with all data and address output lines fully buffered for protection. The IBM PC buses include a data bus, an address bus and a control bus.

The data bus and the address bus are the primary buses. Information on the data bus can travel either to or from the CPU. Even though the IBM PC has a CPU that operates on 16-bit instructions it is technically an "8-bit machine" because the data word on the data bus is eight bit wide. It requires a signal to control the direction of data flow.

The largest bus in the PC is the address bus. This bus carries the addresses the CPU accesses for program instructions or data. The address bus is 20 bits wide enabling it to address over 1 megabyte or memory locations.

The complete IBM PC bus structure is described in figure 2.18.

Under the direction of the CPU, special control signals are placed on the control bus and unique addresses

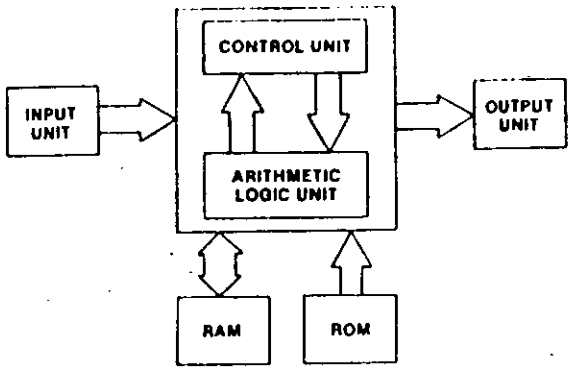


Figure 2.17: Basic Units of IBM PC

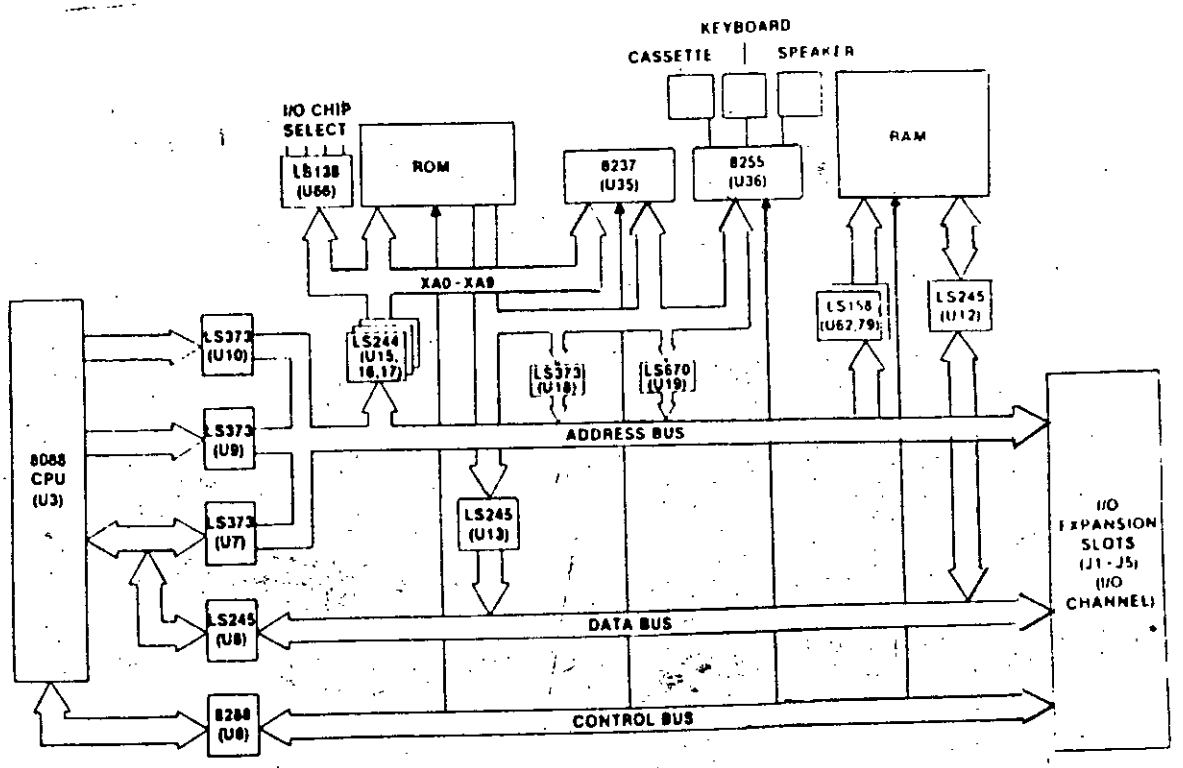


Figure 2.18: Complete IBM PC Bus

are placed on the address bus. The control signals open the address location, letting the information stored in these locations appear on the data bus, which is acted on by the CPU or I/O.

Together, the address bus, the data bus and the control bus are called the system bus. The system bus lies beneath the five expansion slots of IBM PC motherboard. In addition to the address bus, the data bus and the control bus, the 62-pin system bus includes timing and power signals. Another name for this 62-pin arrangement is the PC bus[2].

2.3 DISK OPERATING SYSTEM/VIRTUAL STORAGE (DOS/VS)

Disk Operating System/Virtual Storage (DOS/VS) is a comprehensive collection of program components designed to make full use of the resources of a data processing system. DOS/VS and the hardware system it controls combine to form a complete, effective computing facility[12].

DOS/VS controls the work (input, processing, output) to be performed by the computing system. It supervises the use of system, resources and based on control information from the user, their allocation to the jobs run on the computer system. The component programs that make up DOS/VS may be divided into:

(i) Control programs

(ii) Processing programs

(iii)Data management routines.

For execution, the components of DOS/VS are stored online (that is immediately and directly accessible whenever required) in areas on magnetic disk called libraries. This allows fast loading of any program or routine into storage whenever its function is needed.

DOS/VS control programs comprise the initial program loader, the supervisor and the job control program. The initial program loader is used to start operation with the system. It loads the supervisor into storage. The supervisor controls overall system operation and provides general functions required by the job control program and all processing program[12]. It resides in the lowest area of storage, called the supervisor area, throughout system operation. The job control program is loaded by the supervisor to initiate the execution of each new program are to be invoked while the program is running.

Processing programs are classified as all programs whose execution is initiated by the job control program and controlled by the supervisor. Processing programs can be divided into three categories: language translators, service program and application programs.

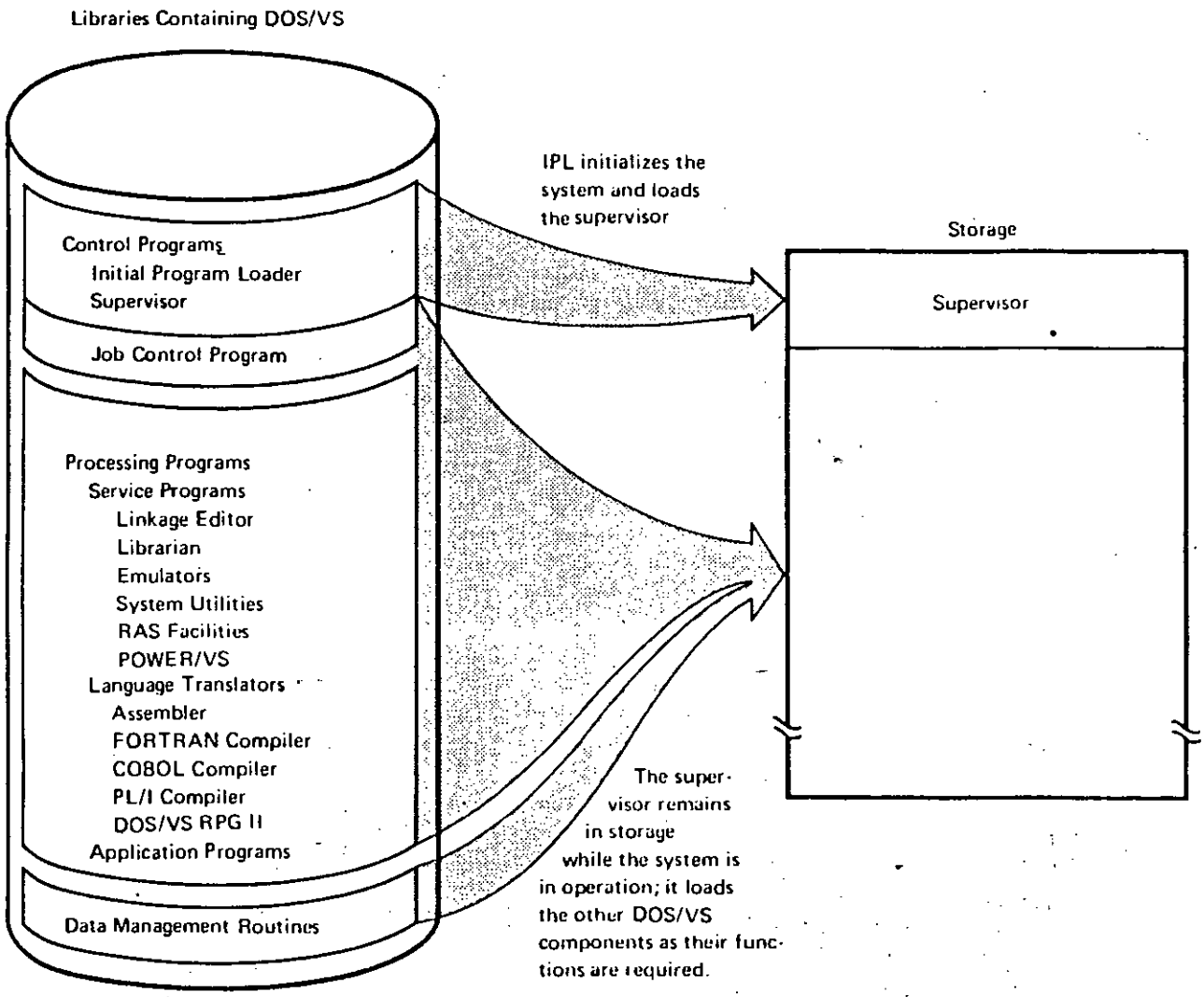


Figure 2.19: Summaries of Concepts of DOS/VS

A third important class of components of DOS/VS are its data management routines. These are available for inclusion in problem programs to relieve the programmer of the detailed programming associated with the transfer of data between auxiliary storage and programs. Figure 2.19 summarizes the concepts of DOS/VS described.

DOS/VS operating systems most significant functions and facilities are system control, libraries, data management, system operator communication, reliability availability and serviceability, emulators and system generation.

The function of system control falls into two main categories: functions that control the processing of jobs and function that controls the allocation and the use of system resources. Functions that controls the processing of jobs includes automatic job-to-job transition, symbolic device addressing, loading of programs from the libraries for processing, handling of program termination. While function that controls the allocation and the use of system resources include multiprogramming with concurrent execution of a number of programs, virtual storage support, input/output queueing to achieve efficient use of local and remote I/O devices and the CPU and job accounting.

DOS/VS allows the user higher flexibility in organizing and utilizing the storage in which to process his programs. The single-partition system presents the simplest type of storage organization. The lower storage area contains the supervisor which remains resident throughout system operation. The remaining area or

background partition is where all other program execute. A standard DOS/VS storage protection feature isolates the supervisor and all processing programs during system operation, so that one program cannot cause damage to another.

DOS/VS allows the user to divide the problem-program area into as many as five areas, called partitions. Each partition can contain a separate program, which allows concurrent execution of multiple programs. This is called multiprogramming. Each program is logically independent, but it takes turns with the other programs in using the CPU facilities, thus reducing the time that the multiprogramming system is in the unproductive wait state. Figure 2.20 and 2.21 show the storage organizations.

```
0K   ------------------------------
     |        Supervisor area      |
     |-----------------------------|
     |                             |
     |        Problem-Program      |
     |        Area or              |
     |        Background           |
     |        Partition            |
     |                             |
Max. K  ------------------------------
```

Figure 2.20 : Storage Organization; Single partition

```
θ K  ------------------------------
     |        Supervisor area       |
     |------------------------------|
     |        Background            |
     |------------------------------|
     |        Foreground-4          |
     |------------------------------|
     |        Foreground-3          |
     |------------------------------|
     |        Foreground-2          |
     |------------------------------|
     |        Foreground-1          |
Max. K  ------------------------------
```

Figure 2.21 : Storage Organization; Multiple partition

The programs and routines that make up DOS/VS are stored in libraries on disk storage. There are four types of libraries in DOS/VS system - source statement, relocatable, core image and procedure - which correspond to the basic formats in which program modules and control information may be maintained.

The transfer of data between auxiliary storage devices and programs as well as organization of such data is usually controlled by the DOS/VS data management routines. The services of data management are invoked by all system

and user-written programs whenever they require the execution of input/output operations.

2.4 DOS/VS ENTRY TIME SHARING SYSTEM (ETSS)

The DOS/VS Entry Time Sharing System is an intermediate level interactive system designed to provide the power of the computer to several individual terminal users concurrently[16].

An interactive system is one which interacts with a terminal user by allowing him to enter information or commands and to receive responses back from the system in the manner of a dialogue. This dialogue between the terminal user and the time sharing system is the "interaction" referred to[16].

The term "time sharing" refers to the fact that several different terminal users appear to be using the computer system simultaneously; thus sharing the resources of the computer. The individual terminal user interacts with a System/370 via display terminals.

ETSS is broken down into the following functional areas:

i) Foreground Terminal Command Processors
ii) Main Control Program
iii) On-line Service Programs
iv) Utility Programs.

The Foreground Terminal Command Processors read and interpret commands or data from the user and perform the requested activity. This could consist, for example, of saving a member in the library, printing a member on the terminal or entering a line of input data.

The Main Control Program supervises the activities of all components of the system. Its responsibilities include: attaching and monitoring all subtasks, scheduling jobs, intercepting SVCs, performing library file I/O and allocating buffers.

The On-line Service Programs perform such functions as job entry statement interpretation, operator communication, job scheduling, interpretation of procedures or command lists and abend dump processing.

The Utility programs provide numerous functions both for the terminal user and the system programmer who maintains the system.

The ETSS library file is a file on DASD device. Access to the data in this file is via a specialized ETSS direct access method. The library file is divided into three logical areas:

Fixed area: Containing system record, user profile records and library header records.

Permanent area: This area is distributed among various ETSS users in the form of user libraries and their corresponding directories. The size of the user library is not fixed, it competes for records with other user libraries.

Temporary area: This area competes with permanent area for records. The records in the temporary area are chained from ETSS internal blocks. Figure 2.22 gives an overview of the ETSS library structure.

Each user of ETSS has access to one or more libraries. Each library has a directory which may contain any number of entries. Each directory entry, contains a name which the user has given to his data. A uniquely identifiable set of data in the library is called a 'member'.

Each user of the system has a unique, four character user identification. Associated with this user identification are two data records referred to as the profile records which contain information like identification code logon, password, identification of library, accounting information, library usage, information, security level, etc.

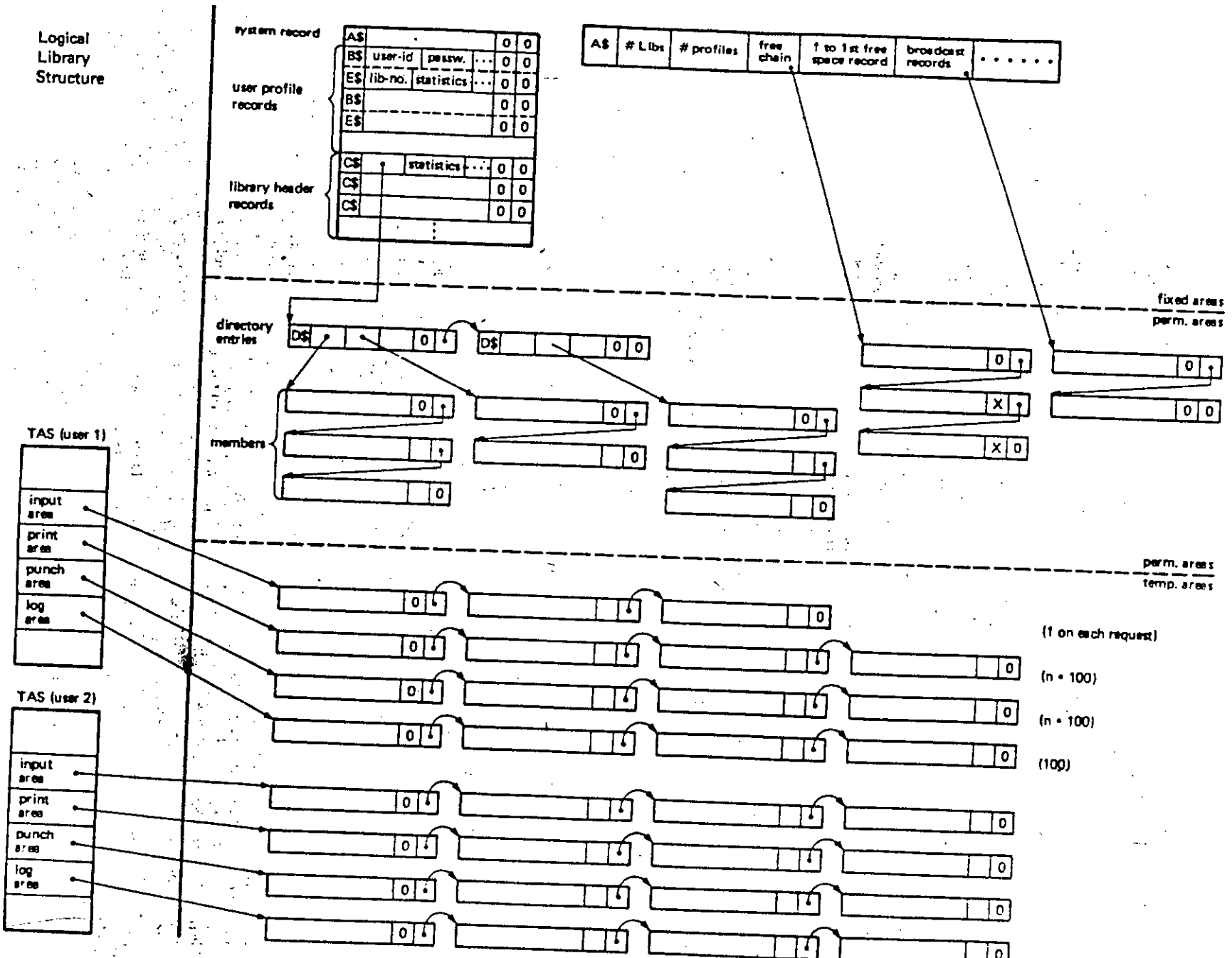


Figure 2.22: ETSS Library Structure

Chapter 3

INTERFACE HARDWARE DESIGN

3.1 BIT TIMINGS

When not executing a command operation, the control units continually perform an internal poll of all attached devices. Internal polling is performed to determine the current status and whether the device has an I/O pending condition.

The 3270 Control unit to device interface is a single wire coax, characteristic impedence 93 ohms, with serialy-bit data transferred in either direction but only one direction at a time[17]. Bits on the coaxial cable appear as negative-going pulses. The centre conductor of the coaxial cable, when measured at the control unit with reference to the outer conductor will be +7.4 nominal (+1.1-2.2) volts with no signal present and power on at each unit. The signal from the control unit on the coaxial cable centre conductor will appear at the device as depicted in figure 3.1.

Bit timings from the device to the control unit will meet the same requirement as from control unit to device except for bit rate. The bit rate from the device will be 630 ns (minimum) to 1.05 microsecond (maximum) per bit. The

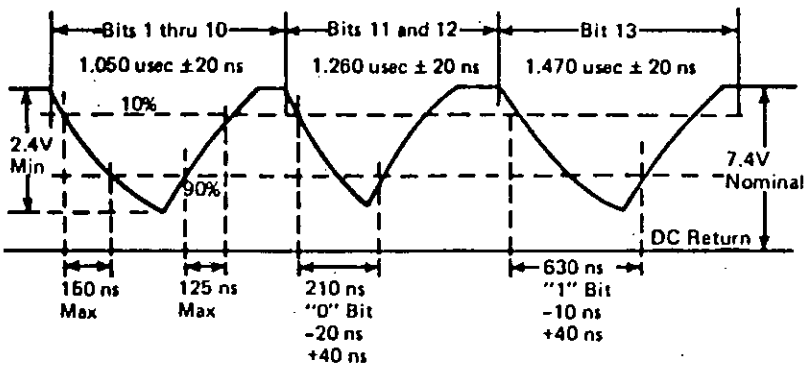


Figure 3.1: Signal from the Control Unit on Coax Cable

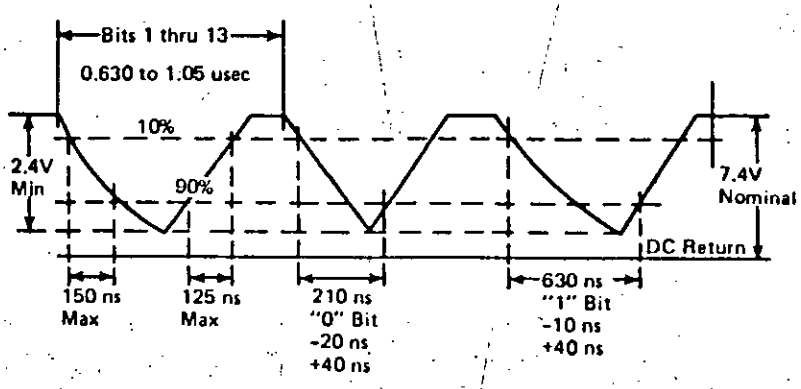


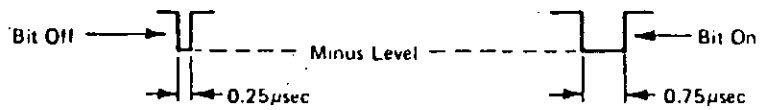
Figure 3.2: Signal from the Display on Coax Cable

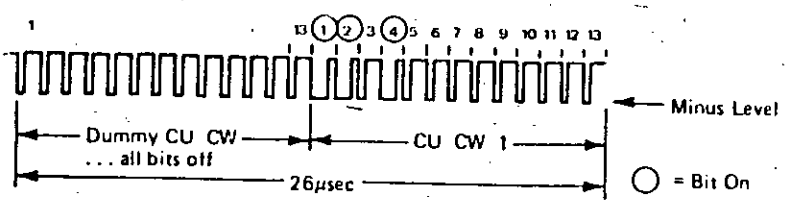
minimum duration of the Up level, after crossing the 10% point going in the positive direction for a 1 or 0 bit until the start of the next consecutive bit will be 30 ns as depicted in figure 3.2.

3.2 3270 COMMUNICATION PROTOCOL

The 3270 data stream consists of control words, status words and data words which are transmitted between the control unit and the devices attached to it. A word is composed of 13 bits whose timings are shown in figures 3.1 and 3.2. Bit width for the bits 1 to 10 are same while bit 11 and bit 12 are of equal width but the duration is bigger than bits 1 to 10. Width of bit 13 is maximum in the series. Again these three levels of timings are different for signals from the control unit to device and signals from the device to control unit. Whether the bit is one or zero is decided by the duration for which it was low. A one is three times wide than a zero.

The control unit performs an internal poll by sending a dummy word followed by a Control Word 1. The dummy word is composed of 13 zeros. The word following the dummy word is detected as control word 1 by bit 2 and 3 while bit 4 indicates a poll. The first bit of a word indicates busy condition and bit 12 is the parity bit for bit 1 through 12. The parity must be odd. During transmission from control unit to device bit 13 is not used. When device sends a word to control unit in bit 13 it indicates whether its a 480 character or a 1920 character device. In response to the poll a 3277 display device sends a status word with bit 1 and bit 13 one. This is shown in figure 3.3 .





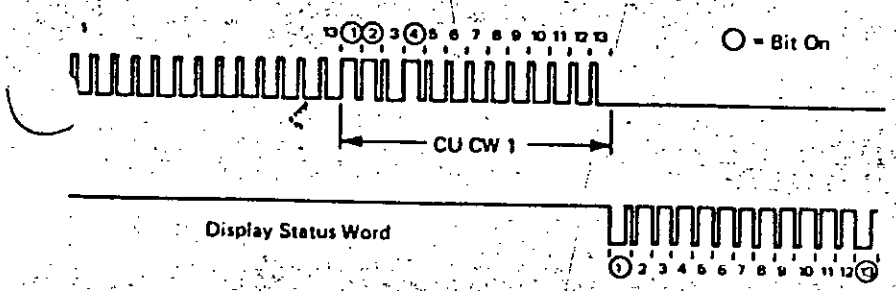


Figure 3.3: Dummy Word, CW1 and Display Status Word

3.2.1 Word Formats

The control unit transmits two types of words in addition to dummy words. These are control words and Data words. The control words are used for sending different control informations which governs the movement of the data stream and commands to initiate such operations as the total or partial writing, reading and erasing of data in a selected 3270 device buffer[16]. Format of the control words are shown in figure 3.4 along with the formats of the display station status word.

The Data Words carry both character data and attribute data. Bit 4 identifies the word as attribute or character. A one in this position indicates that this an attribute word while zero indicates a character word. The formats of data words are shown in figure 3.5.

3.3 INTERFACE HARDWARE

The basic responsibility of the hardware is to interface the System/370 and the IBM PC - working in parallel with a display device. In order to carry out this task the hardware must have the following characteristics:

i) It must be able to synchronize its reception with the words consisting 13 serial bits.

ii) It must be able to distinguish the dummy word, control words and data words.

CU Control Words

Figure 1-16 shows the formats for CU control words.

Control Word 1	Busy Bit	1	0	Poll	Rd	Wrt	Sys Rdy	Unlk Kbd	Erase Unprot	Reset Xmit Chk	Ack	P	0
	1	2	3	4	5	6	7	8	9	10	11	12	13
Control Word 2	Busy Bit	1	1	Poll	Spare	Format		Start Print	Sound Alarm	Reset Xmit Chk	Spare	P	0
	1	2	3	4	5	6	7	8	9	10	11	12	13

Bit 2 = 1 Identifies a control word
Bit 3 = 0 Identifies control word 1
Bit 3 = 1 Identifies control word 2

Bits 6 and 7 = 0,0 Variable line-length format, up to 132 char/line.
= 0,1 40 char/line format
= 1,0 64 char/line format
= 1,1 80 char/line format

Note: Either or both control words may be transmitted to a selected Display Station or Printer.

Display Station or Printer Data Word

Busy Bit	0	Csr	Ctl	Data Bits 1	2	3	4	5	6	7	P	
1	2	3	4	5	6	7	8	9	10	11	12	13

Bit 2 = 0 Identifies data word
Bit 3-11 = These 9 bits are fetched from the device buffer.
Bit 12 = Parity bit assigned by device
Bit 13 = 0 Identifies 480-byte device
= 1 Identifies 1920-byte device

Display Station Status Word

(Response to CU Poll)

Busy Bit	0	Busy	Dev Chk	Xmit Chk	Info Pend	Five Low-Order Bits or AID					P	
1	2	3	4	5	6	7	8	9	10	11	12	13

Bit 13 = 0 Identifies 480-byte device
= 1 Identifies 1920-byte device

Printer Status Word

(Response to CU Poll)

Busy Bit	1	Busy	Dev Chk	Xmit Chk	Info Pend	Not Rdy	Spare	Equip Chk	Printer Hang	Spare	P	
1	2	3	4	5	6	7	8	9	10	11	12	13

Bit 13 = 0 Identifies 480-byte device
= 1 Identifies 1920-byte device

Figure 3.4: CW and Display Word Format

CU Data Words

Figure 1-15 shows the formats for CU data words. The formats for transfer to the display station and to the printer are identical.

a. Character Format

Busy Bit	0	Csr	Data Bits 0	1	2	3	4	5	6	7	P	0
1	2	3	4	5	6	7	8	9	10	11	12	13

Bit 2 = 0 Identifies data
Bit 4 = 0 Identifies character data

b. Attribute Format

Busy Bit	0	Csr	1		Unprot Prot	A/N	I/SPD		Esc	MDT	P	0
1	2	3	4	5	6	7	8	9	10	11	12	13

Bit 2 = 0 Identifies data
Bit 4 = 1 Identifies attribute data
Bit 5 = Unused
Bit 6 = 0 Unprotected data
= 1 Protected data
Bit 7 = 0 Alphameric data
= 1 Numeric only data
Bits 8,9 = 0, 0 Normal intensity/nondetectable data
= 0, 1 Normal intensity/selector pen detectable
= 1, 0 High intensity/selector pen detectable
= 1, 1 Nondisplay/nonprint/non-detectable
Bit 10 (Escape) = This bit is not decoded.
Bit 11 (Modified Data Tag) = 0 Field data not modified
= 1 Field data modified

Bits 5-11 all zeros = Default option
= Unprotected, A/N, normal intensity/nondetectable data.

Figure 3.5: CU Data Word Format

iii) It must be able to identify the direction of transmision: whether it is from the control unit to device or from the device to control unit.

iv) Since the System/370 computers are much faster than the IBM PC and the control unit transmits a full buffer of 1920 characters at a burst the hardware must be buffered and able to store data transmitted by the control unit into its buffer at the high speed of transmission of the control unit which is connected to a multiplexer channel of the System/370.

v) The hardware must also contain the interface circuit to transmit data stored in its buffer to the personal computer at a variable speed.

The hardware that will have the above characteristics may obviously be divided into two distinct parts. The first part will generate all the control signals from the bit stream transmitted from the control unit which is called the synchron and control signal generator while the other part will convert the serial bits to parallel data, store it in the memory of the interface circuit and will transmit parallel data from memory to a parallel port of IBM PC for input to data transfer software running in IBM PC which is termed as the Data acquisition and retrieval circuit.

3.3.1 Synchron and Control Signal Generator

The major responsibility of this part is to generate different synchronization and control signals and to drive

the data acquisition and retrieval part. The block diagram of this part is given in figure 3.6. The synchron and control signal generator is composed of a clock, bit synchronizer, word synchronizer and Synchron separator.

The clock generates a clock signal of frequency 20MHz. It uses a 20MHz crystal and two inverters to do it. The clock pulses are used to measure the width of the signal bits transmitted by both control unit and device.

Bit Synchronizer takes signal and clock pulses of frequency 20MHz. It measure the width of the signal and helps decide whether it is a one or zero and at the same time produces a pulse at the middle of each serial bit of the signal transmitted by the control unit and the device.

Word synchronizer takes input from bit synchronizer to produce word synchronization signals. The word synchron pulses are generated at the end of 13th bit of the word. It monitors the thirteenth bit to produce the synchron pulses and if in any case transmission of next word starts before completion of the 13th bit it produces a synchron pulse of its own and helps synchronize the reception of signal with the words.

Synchron separator takes bit synchronization and word synchronization pulses as input and produce a one or zero and its complement at the end of the thirteenth bit depending on the direction of transmission of signal. It is the synchron separator who decides that signal from which direction is to be received by the interface hardware.

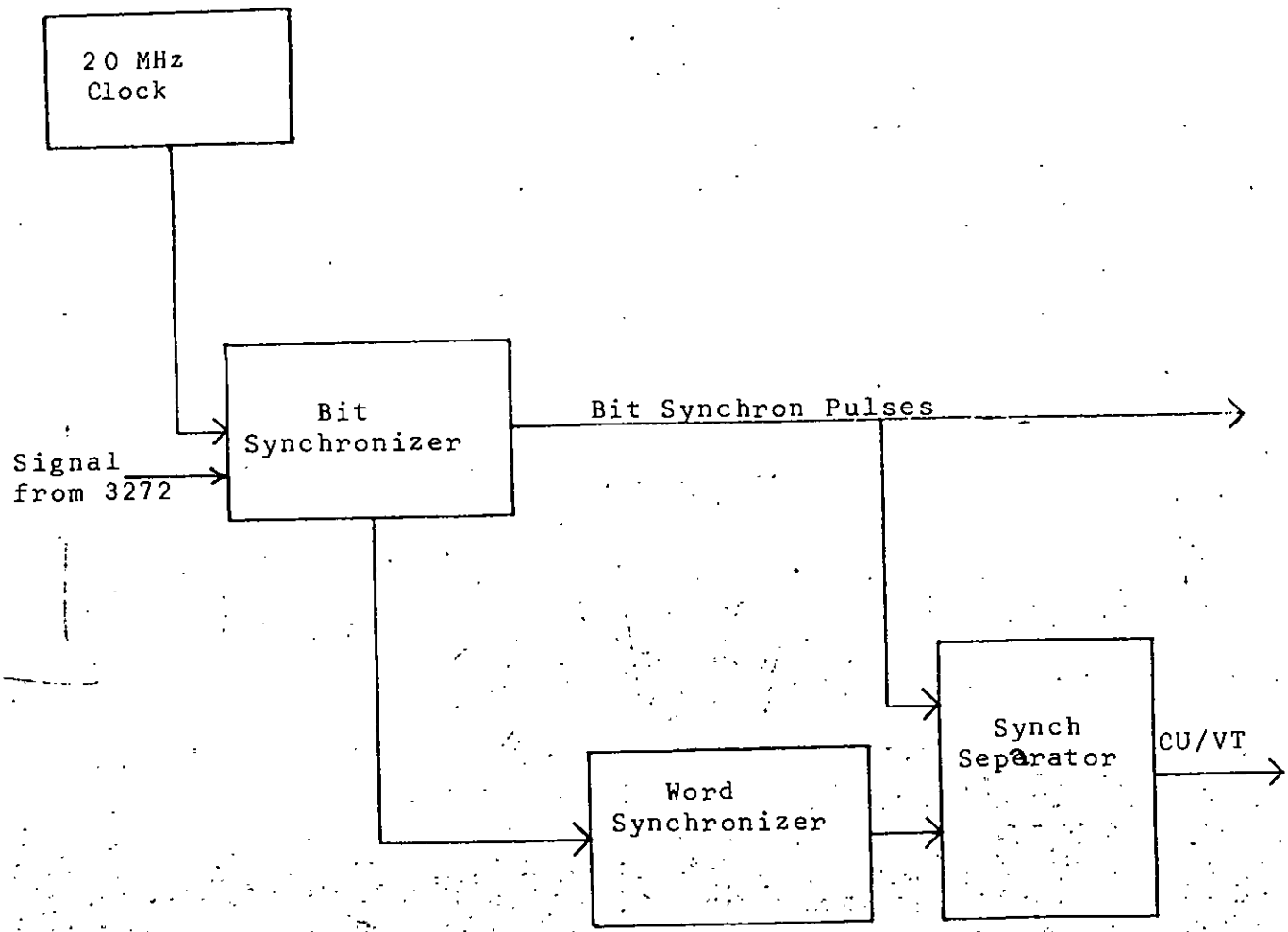


Figure 3.6: Block Diagram of Synchron Generator

3.3.2 Data Acquisition and Retrieval Circuit

This part is driven by the control and synchron signals generated by the first part. It is composed of four different blocks which are: Deserializer, Buffer, Control unit, Address builder and Memory as depicted in figure 3.7.

Deserializer receives signals from the data transmission line connecting the control unit and the display device and from the bit synchronizer. It shifts the serial data received from the line to parallel data with the help of the bit synchron pulses and makes it available at the output.

The main responsibility of the buffer is to interface the memory and deserializer. It also helps in controlling the direction of flow of data to and from the memory.

Control unit is the brain of the data acquisition and retrieval part. It produces all the control signals necessary for the circuit. It takes input from the synchron separator during data acquisition and gives appropriate signals to address builder, memory and buffer and sends a signal to IBM PC to hold data retrieval from memory. During data retrieval it holds data receiving from the signal line and takes clock pulses from PC to build address and keeps the data of the selected location available at the parallel port of IBM PC for input.

Address builder works with input signal from the control unit of the circuit and builds the address to select a memory location. It remains active during the

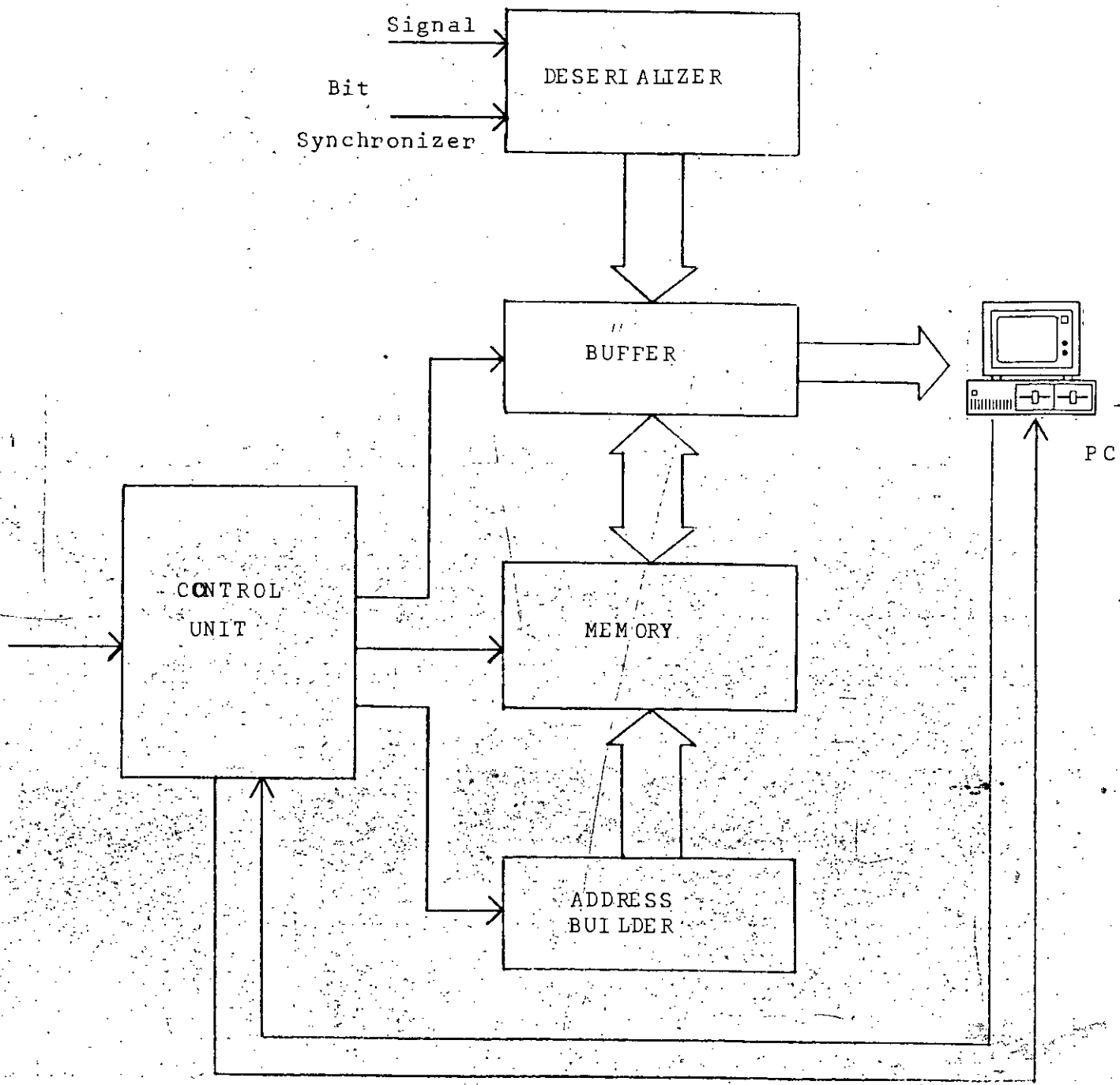


Figure 3.7: Block Diagram of Data Acquisition Hardware

storing and retrieval of data from memory.

The memory of the data acquisition and retrieval circuit works as the buffer of the interface hardware. Between the control unit and the display device data transmission in either direction always takes place in a burst of the size of display device. Therefore, the interface hardware must have a buffer of at least the size of the display device which is 1920 for a 3277/78 display. However, the interface hardware buffer is 2K bytes.

3.3.3 Interface Circuits

3.3.3.1 Synchron and Control Signal Generator Circuit

It has been described previously in 3.1 that in a word there are three different bit timings. In the CU transmitted words bit duration for bit 1 through 10 is 1.050 usec ±20 ns for bits 11 and 12 it is 1.26 usec ± 20 ns and for bit 13 it is 1.470 usec ± 20 ns. In the display transmitted words, the duration for bit 1 through 10 is .630 usec ± 20ns, for bits 11 and 12 it is .840 usec ± 20 ns and for bit 13 is 1.05 usec ± 20ns. Bit off condition is indicated by a minus level of .25usec and a bit on condition is indicated by a minus level of .75 usec. The synchron and control signal generator circuit is responsible for synchronizing the signal reception with words and producing a bit synchron pulse, a word synchron pulse and a control signal indicating the direction of transmission. To do this, a 20MHz clock is being used which gives a pulse at every 50ns. Since the bit off condition is indicated by a minus level of .25usec and bit on by

.75usec, the signal may be sampled at .30 usec from its start to see whether it is on or off. To synchronize data reception bit synchronizer and storing a synchron pulse is required at the end of the word.

There may be two ways to do it:

i) By counting the signal pulses and producing a pulse at the end of bit 13 but in that case a way is to be found out to indicate the beginning of a word.

ii) By observing the pulse width: since the width of the bit 13 is the largest in the word irrespective of direction of transmission by measuring and finding out the widest bit in the word a word synchron pulse can be generated.

It has been observed that the timing of the display device bits 11 and 12 are .84 usec and that of bits 1 through 10 is less and that of bit 13 is 1.05 usec while timing of all the CU bits are more than or equal to 1.05 usec. Therefore if a counter starts counting from zero at the beginning of every bit for a word transmitted by display device the counter will not be able to count upto .9 usec, on the other hand for a word transmitted by CU it will exceed this value for every bit. Again the same counter will count upto 1.3 usec in case of bit 13 of CU transmitted words since the timing of CU word bits 11 and 12 is 1.26 usec and that of bits 1 though 10 is less while that of bit 13 is more.

Therefore the bit synchronizer (T_{300}) with 300 nsec

delay from starting of a bit may be obtained by counting 6 at the counter driven by 20MHz clock (50 nsec time period). So the logical equation for T_{300} is as follows.

$$T_{300} = \overline{Q_{A1}}.Q_{B1}.Q_{C1}.\overline{Q_{D1}}$$

Similarly the signal delay from the starting of the bit of the values 900 nsec and 1.3 usec to generate word synchron signal for the CU and the device may be obtained by counting 18 and 26 respectively. The formulae for these two cases will be as follows

$$T_{900} = Q_{A2}.Q_{B1}$$

$$T_{1300} = Q_{A2}.Q_{D1}.Q_{B1}.\overline{Q_{A1}}$$

The inverted signal is fed to a monostable multivibrator (U_4) to produce a small width signal which resets the counter. This signal along with the 900 nsec signal is fed to a SR flipflop (U_5). The complementary output $\overline{Q_5}$ of U_5 is applied to the input of another monostable multivibrator (U_3). For display bits 1 through 12, $\overline{Q_5}$ of the SR remains high but at bit 13 of display input signal S of SR goes low while R input of SR remains high except at the start of every bit. Therefore, $\overline{Q_5}$ goes low and remains there until the first bit of next word comes to change the state and the output Q of U_3 will be high at beginning and low in the middle of display words. Since all bit timings are greater than 900 nsec for the control unit words; $\overline{Q_5}$ of SR flip-flop U_5 will be low at the start of every bit and it will be made high by 900ns signal. If the timing of U_3 is set to a value greater than

that of any bit (1.47 usec), then during transmission from CU the output of the monostable U_3 will remain high because of the retriggering at the starting of every bit. On the other hand, since the timings of the bits transmitted by the device is less than 900 nsec, the monostable will not be triggered once at the beginning only. So the monostable output Q_3 (U_3 pin 13) will remain at level zero during transmission from the display device, thus indicating the direction of transmission. If the output of U_3 is indicated by C_U, it may be written that

i) if transmission from CU to device then $C_U = 1$

ii) if transmission from device then $C_U = 0$

Therefore for identifying the word synchron signal for the CU(W_{CU}) and display terminal (W_D) the following logical equations may be used

$$W_{CU} = C_U \cdot T_{1300}$$

$$= \overline{\overline{C_U} + \overline{T_{1300}}}$$

and

$$W_D = \overline{C_U} \cdot T_{900}$$

$$= \overline{C_U + \overline{T_{900}}}$$

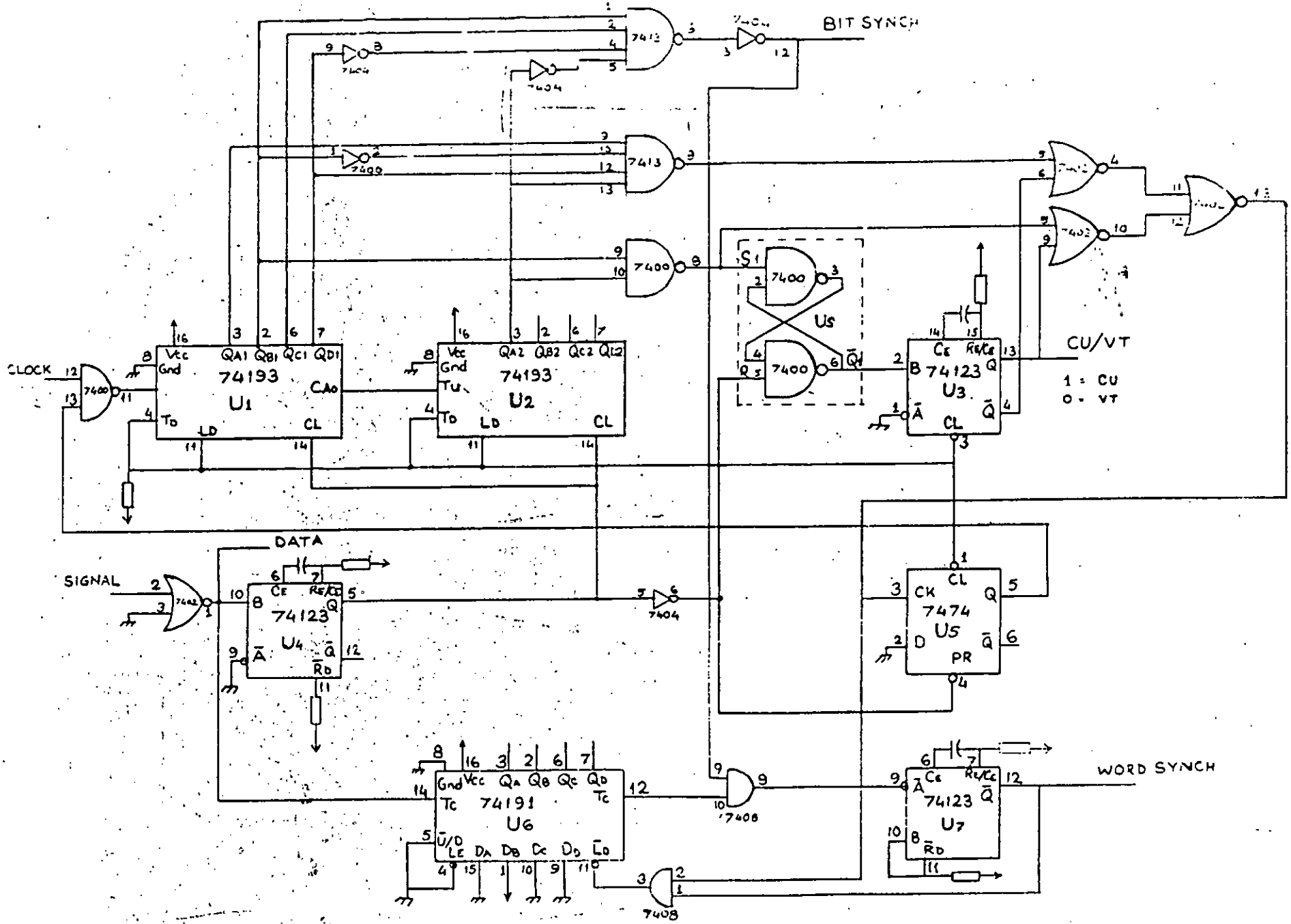


Figure 3.8: Circuit Diagram of Synchron Generator

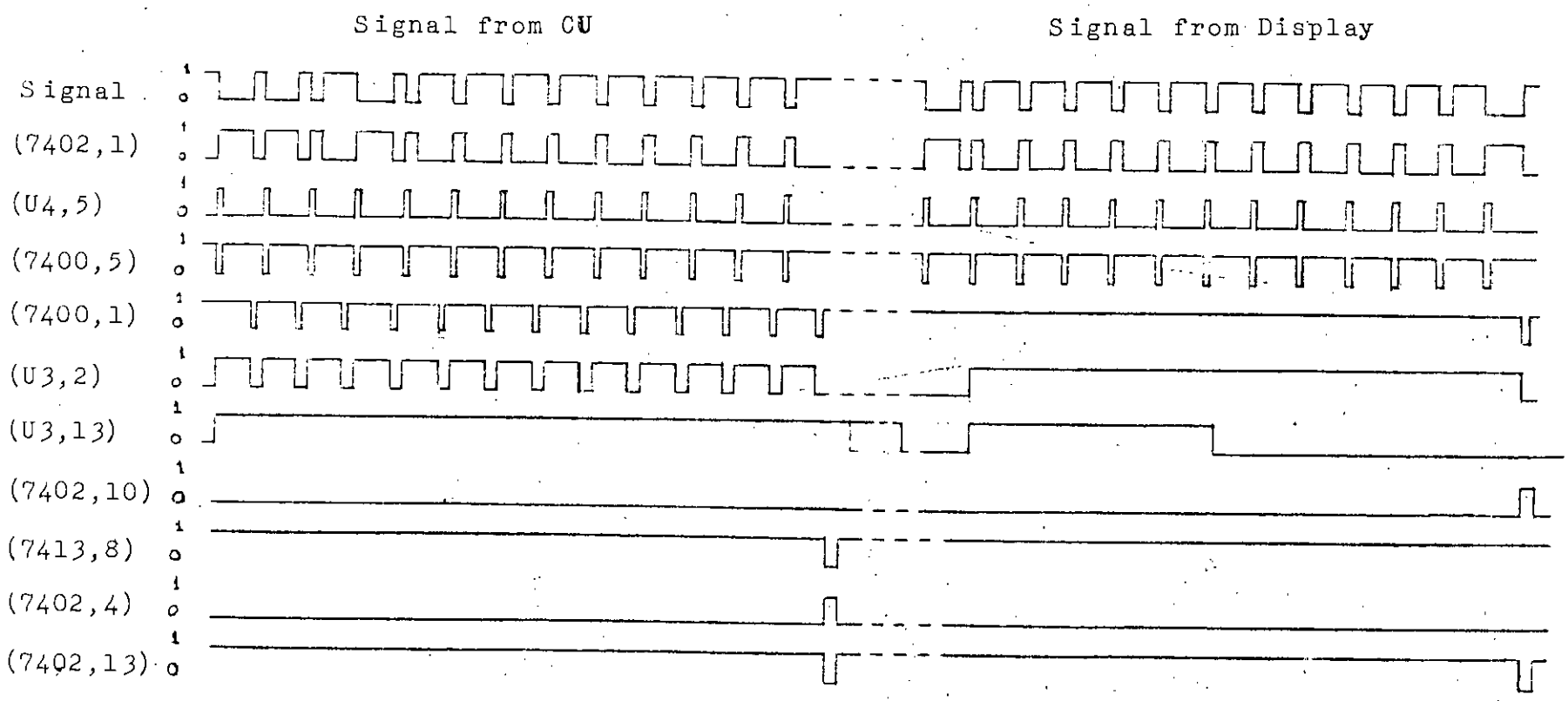


Figure 3.8a: Timing Diagram of Synchron Generator

It has been observed that during the transition of direction of transmission from "CU to display" to "display to CU" the timing of bit 13 is not maintained at the specified a value. So the synchron pulse thus produced is missed at the transition. This missing synchron is reproduced using a counter and producing a pulse at count 13. The word synchron pulse is held at up level for a while using a monostable multivibrator so that the data shift in deserializer can be completed before storing it in memory. The circuit diagram is given in figure 3.8.

3.3.3.2 Data Acquisition and Retrieving Circuit

The main task of the interface hardware is to acquire data and store it for later retrieval. Since the signal received from the control unit is serial and the control unit sends a full buffer of 1920 characters at a burst the hardware must be able to deserial the serial data and must have a memory circuit to store the characters sent in a burst and it must also have circuitry to make the data available at its parallel port for inputing to IBM PC.

The signals received from the control unit is inputted to a shift registers (U_1 and U_2). The bit synchron pulses are used as clock pulses to shift the serial data to make it parallel. The parallel data then go to two buffers (U_4 and U_5) which interface the memory and shift registers. The control circuit receives word synchron pulses from the synchron generator circuit to build address to send output control pulses to buffer and write-pulses to memory. The address is built by a counter (U_{12}, U_{13} and U_{14}) which uses

word synchron pulses as its clock input. Since the signals are 13 bit words to store a word 13 parallel bit locations are required. Because of unavailability of such memory two 8 bit memory chips (TC5516) are used each of which can store 2048 bytes. A data selector circuit is used to select between the low order and high order bytes at the same memory address. A circuit diagram of the data acquisition and retrieval circuit is given in figure 3.9.

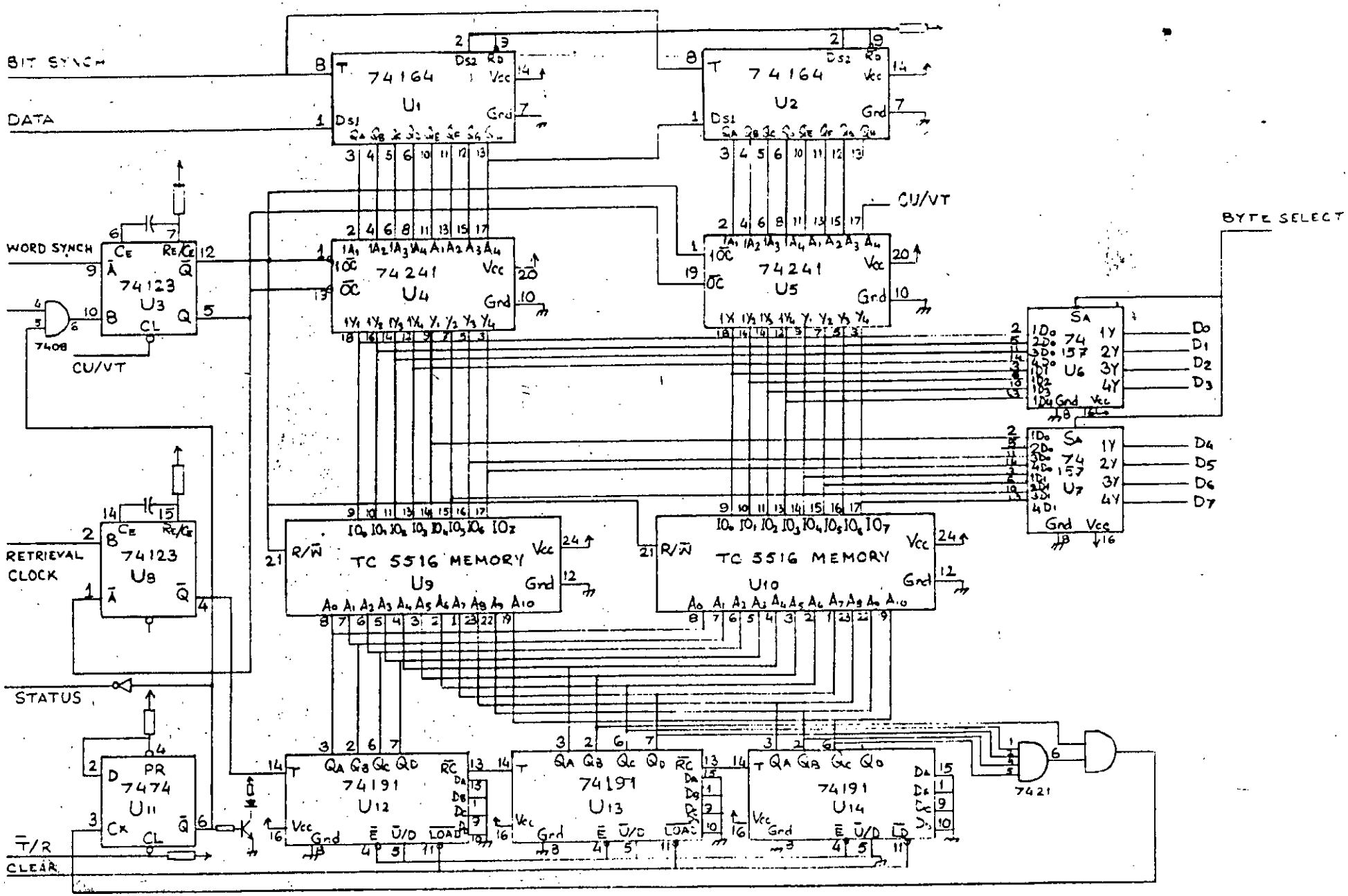


Figure 3.9: Circuit Diagram of Data Acquisition Hardware

Chapter 4

INTERFACE SOFTWARE DESIGN

4.1 METHODOLOGY

The interface hardware picks up the data words and attribute words transmitted by the control unit to the device in parallel with which it is working and stores the words in its memory. Therefore an image of the display screen is being saved by the interface hardware. The contents of the interface memory can be retrieved by sending some control signals to the control unit of the data acquisition and retrieval circuit. The reception of the interface hardware can also be controlled from the PC. The interface hardware is also capable of making the data available from any memory location at its output port selected by sending control signals from PC.

The 3270 Information display system control unit transmits only a set of predefined characters in a structured EBCDIC form. The attribute characters are transmitted intermixed with the structured data. For these reasons, initially the hardware was designed to make it a data acquisition unit to pick up and store all data transmitted by the control unit and device. A software was written to display or print at option all the 13 bits of

the words along with a control bit which would determine the direction of data flow. The data received this way were then checked and verified with actual data transmitted and the control words. This helped determine the ability of hardware designed and also helped determine the basic characteristics of software. A listing of the program along with sample output is included in the appendix - VI.

4.2 INTERFACE SOFTWARE

It has already been mentioned that the interface hardware stores a copy of what is displayed on the 3270 display station screen along with the attributes in its memory. So, the basic responsibility of the interface software is to retrieve the stored bits and decipher it to reproduce the characters transmitted by the control unit. In addition to this it must give the users all ease that he deserves from a standard software. This suggests that the interface software must have the following characteristics.

i) It must be able to control the interface hardware so that at the beginning of the data reception memory address is reset.

ii) It must be able to detect when the interface hardware stops reception so that it can start data retrieving.

iii) It must be capable of analyzing the bit streams to identify data and attribute and the attributes must be replaced by blanks as they occupy a location in the display device buffer.

iv) Since the control unit transmits a set of predefined characters in a structured EBCDIC form, it must take the responsibility of rebuilding the EBCDIC characters and convert to ASCII as in all microcomputers ASCII is the standard exchange form.

v) It must allow the user to define a beginning and end of a block of data or program he wants to transfer from mainframe to IBM PC.

vi) It must let the user concatenate a number of files during transfer operation.

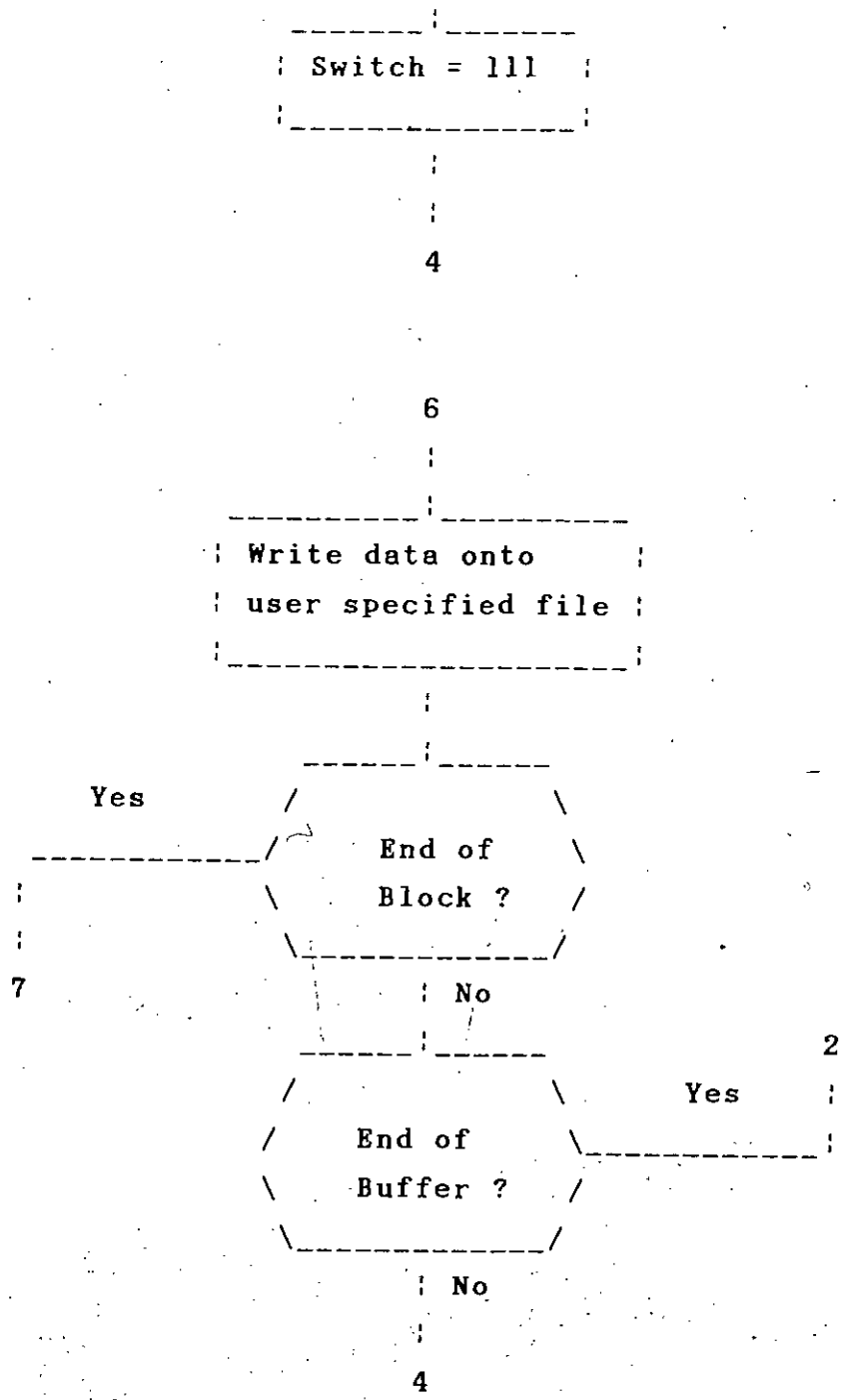
vii) It must be able to store the transferred data onto some computer storage medium in a file specification defined by user.

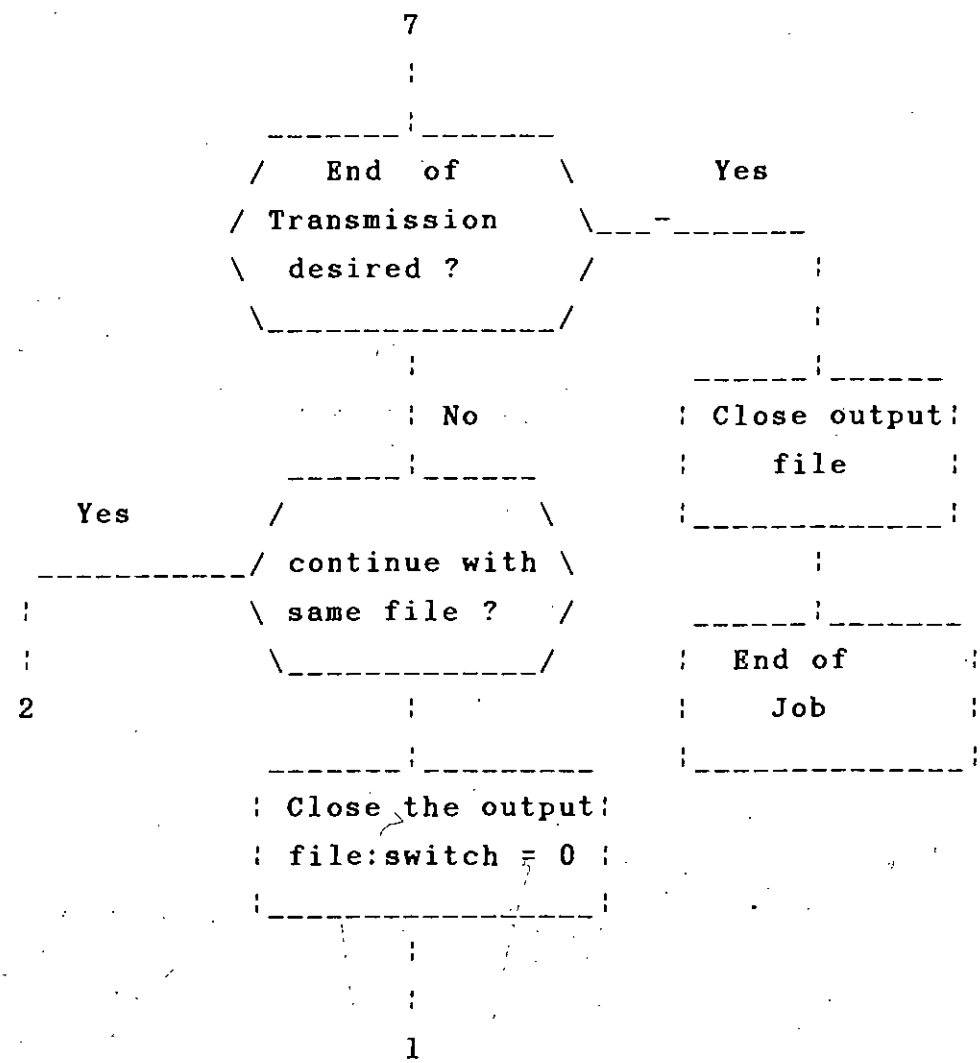
The software controls the interface hardware outputting signals through an output port and it monitors the status using an input port. In addition to these ports a 8 bit parallel port is used by the software to input the contents of the memory circuits of interface hardware. To define a block a data to be transferred to PC a beginning-of-block and an end-of-block symbols are used.

4.3 FLOW DIAGRAM OF INTEFACE SOFTWARE

```
                    Start
                      |
 ---------------------|---------------
| Read the data                        |
| Conversion table                     |
|--------------------------------------|
                      |
                      |
 --------------------|-------
| Switch = 0                           |
|--------------------------------------|
                      |
                      |<-------  1
 ---------------------|---------------
| Accept output file                   |
| specification from user              |
| and open it                          |
|--------------------------------------|
                      |
                      |
 ---------------------|---------------
| Reset all counters to                |
| predefined value                     |
|--------------------------------------|
                      |
```

```
                    :
 ____________________:____________
: Start data acquization          :
:__________________________________:
                    :
                    :<-----------------:
        ________:________             :
       /                 \   No       :
      /   Interface       \___________:
      \   Buffer full ?   /
       \_______________/
                    : Yes
                    :
                    4
                    :
 ____________________:____________
: Read a data word from           :
: interface memory and            :
: rebuild the character           :
:__________________________________:
                    :
                    :<------------- 5
  Yes     _______:______
 _________/ Switch=111?  \
:         \______________/
6                   : No
        ________:________
       /                 \
      /  Beginning of     \_____ 4
      \  Block found ?    /  No
       \_________________/
                    :Yes
```


Switch = 111
4
6
Write data onto
user specified file
Yes
End of
Block ?
7
No
2
Yes
End of
Buffer ?
No
4



The listing of the program written following this diagram is given in appendix - VII.

Chapter 5

DISCUSSIONS, CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

5.1 DISCUSSION OF RESULTS OBTAINED

The ultimate objective of the work was to devise a method of down-loading data from mainframe computers to any personal computer. To attain this objective the following steps were followed:

i) Study the IBM 3270 Information Display System and the Basic Telecommunication Access Method (BTAM).

ii) Investigate into the information content of the signal transmitted in either direction.

iii) Study of information content of the signal so that a data acquisition unit (DAU) could be designed to store word in synchronization with the transmission.

iv) Design of the data acquisition hardware DAU.

v) Design an interface circuitry for retrieval of data from the DAU.

vi) Data analysis to study the communication protocol of the 3270 Information Display System, specially from the downloading point of view.

vii) Design of a downloading hardware and software to establish communication with the 3272 control unit which can interface any personal computer.

As a result of the above studies a hardware has been designed to take an image of display device buffer. The hardware works in parallel with display terminals of the mainframe and have the following characteristics.

i) It is able to synchronize its reception with a serial bit stream consisting of 13 bit words.

ii) It is able to identify the dummy word, Control Word 1 (CW1), Control Word 2 (CW2) and Data Word (DW) transmitted by the Display System Control Unit.

iii) The hardware is able to store data transmitted by the Control Unit into its buffer at the high speed of transmission of the Control Unit which is connected to a multiplexer channel of the mainframe computer.

iv) The hardware also contains the interface circuitry to transmit the stored data in its buffer to the microcomputer.

The interface software has been designed to input data from the interface hardware and to rebuild the characters to store onto some personal computer storage medium. The software has the following characteristics:

i) It is capable of analyzing all the bits of the bit stream.

ii) The software rebuilds the EBCDIC characters from the predefined structured EBCDIC form and translates them to corresponding ASCII Codes.

iii) It is able to store the received data onto some microcomputer storage medium for later use.

Unlike the hardware and software available in the market it is independent of the Operating System and Time Sharing System running in the mainframe. To transfer the data from mainframe to personal computer the user is to define a beginning-of-block and an end-of-block and the data is to be displayed on the screen so that the interface hardware can take an image of that. This can be done by an assembly language program, a time sharing system like Entry Time Sharing System (ETSS) or Interactive Computing and Control Facility (ICCF) or even by Virtual Machine/Conversational Monitoring System (VM/CMS), the time sharing system of virtual machine environment on System/370. The software recognizes the beginning of block and starts filing data until it finds the end-of-block.

Since the 3270 Information Display System can be connected to all IBM System/370, IBM 30XX and IBM 4300 computers, the hardware and software will be able to download data from all the mainframe computers mentioned above.

The popularity of personal computer encouraged computer industries to produce a variety of input and output devices for the personal computers. The ability of downloading data from mainframe to personal computer will open a new horizon of data representation to the user.

5.2 SUGGESTIONS FOR FUTURE WORK

The present work was limited to establishing a simplex communication link. Using the hardware and software developed, data transfer is possible only from mainframe to personal computer. This work can be expanded to establish a duplex communication link. The hardware can be modified to collect more information about the data communication in IBM mainframe computers which can help build an inter-mainframe computer communication network in the country.

APPENDIX - I

EBCDIC CHARACTER CODES

CODE TRANSLATION TABLE

Dec.	Hex	Instruction (RR)	Graphics and Controls BCDIC	EBCDIC(1)	ASCII	7-Track Tape BCDIC(2)	Card Code EBCDIC	Binary
0	00			NUL	NUL		12-0-1-8-9	0000 0000
1	01			SOH	SOH		12-1-9	0000 0001
2	02			STX	STX		12-2-9	0000 0010
3	03			ETX	ETX		12-3-9	0000 0011
4	04	SPM		PF	EOT		12-4-9	0000 0100
5	05	BALR		HT	ENQ		12-5-9	0000 0101
6	06	BCTR		LC	ACK		12-6-9	0000 0110
7	07	BCR		DEL	BEL		12-7-9	0000 0111
8	08	SSK		GE	BS		12-8-9	0000 1000
9	09	ISK		RLF	HT		12-1-8-9	0000 1001
10	0A	SVC		SMM	LF		12-2-8-9	0000 1010
11	0B			VT	VT		12-3-8-9	0000 1011
12	0C			FF	FF		12-4-8-9	0000 1100
13	0D			CR	CR		12-5-8-9	0000 1101
14	0E	MVCL		SO	SO		12-6-8-9	0000 1110
15	0F	CLCL		SI	SI		12-7-8-9	0000 1111
16	10	LPR		DLE	DLE		12-11-1-8-9	0001 0000
17	11	LNR		DC1	DC1		11-1-9	0001 0001
18	12	LTR		DC2	DC2		11-2-9	0001 0010
19	13	LCR		TM	DC3		11-3-9	0001 0011
20	14	NR		RES	DC4		11-4-9	0001 0100
21	15	CLR		NL	NAK		11-5-9	0001 0101
22	16	OR		BS	SYN		11-6-9	0001 0110
23	17	XR		IL	ETB		11-7-9	0001 0111
24	18	LR		CAN	CAN		11-8-9	0001 1000
25	19	CR		EM	EM		11-1-8-9	0001 1001
26	1A	AR		CC	SUB		11-2-8-9	0001 1010
27	1B	SR		CU1	ESC		11-3-8-9	0001 1011
28	1C	MR		IFS	FS		11-4-8-9	0001 1100
29	1D	DR		IGS	GS		11-5-8-9	0001 1101
30	1E	ALR		IRS	RS		11-6-8-9	0001 1110
31	1F	SLR		IUS	US		11-7-8-9	0001 1111
32	20	LPDR		DS	SP		11-0-1-8-9	0010 0000
33	21	LNDR		SOS	!		0-1-9	0010 0001
34	22	LTDR		FS	"		0-2-9	0010 0010
35	23	LCDR			#		0-3-9	0010 0011
36	24	HDR		BYP	$		0-4-9	0010 0100
37	25	LRDR		LF	%		0-5-9	0010 0101
38	26	MXR		ETB	&		0-6-9	0010 0110
39	27	MXDR		ESC	'		0-7-9	0010 0111
40	28	LDR			(		0-8-9	0010 1000
41	29	CDR			)		0-1-8-9	0010 1001
42	2A	ADR		SM	*		0-2-8-9	0010 1010
43	2B	SDR		CU2	+		0-3-8-9	0010 1011
44	2C	MDR			,		0-6-8-9	0010 1100
45	2D	DDR		ENQ	-		0-5-8-9	0010 1101
46	2E	AWR		ACK	.		0-6-8-9	0010 1110
47	2F	SWR		BEL	/		0-7-8-9	0010 1111
48	30	LPER			0		12-11-0-1-8-9	0011 0000
49	31	LNER			1		1-9	0011 0001
50	32	LTER		SYN	2		2-9	0011 0010
51	33	LCER			3		3-9	0011 0011
52	34	HER		PN	4		4-9	0011 0100
53	35	LRER		RS	5		5-9	0011 0101
54	36	AXR		UC	6		6-9	0011 0110
55	37	SXR		EOT	7		7-9	0011 0111
56	38	LER			8		8-9	0011 1000
57	39	CER			9		1-8-9	0011 1001
58	3A	AER			:		2-8-9	0011 1010
59	3B	SER		CU3	;		3-8-9	0011 1011
60	3C	MER		DC4	<		4-8-9	0011 1100
61	3D	DER		NAK	=		5-8-9	0011 1101
62	3E	AUR			>		6-8-9	0011 1110
63	3F	SUR		SUB	?		7-8-9	0011 1111

1. Two columns of EBCDIC graphics are shown. The first gives IBM standard U.S. bit pattern assignments. The second shows the T-11 and TN text printing chains (120 graphics).
2. Add C (check bit) for odd or even parity as needed, except as noted.
3. For even parity use CA.

TWO-CHARACTER BSC DATA LINK CONTROLS

Function	EBCDIC	ASCII
ACK-0	DLE,X'70'	DLE,0
ACK-1	DLE,X'61'	DLE,1
WACK	DLE,X'6B'	DLE,;
RVI	DLE,X'7C'	DLE,<

CODE TRANSLATION TABLE (Contd)

Dec.	Hex	Instruction (RX)	Graphics and Controls BCDIC	EBCDIC(1)	ASCII	7-Track Tape BCDIC(2)	Card Code EBCDIC	Binary
64	40	STH		Sp	Sp	(3)	no punches	0100 0000
65	41	LA			@		12-0-1-9	0100 0001
66	42	STC			A		12-0-2-9	0100 0010
67	43	IC			B		12-0-3-9	0100 0011
68	44	EX			C		12-0-4-9	0100 0100
69	45	BAL			D		12-0-5-9	0100 0101
70	46	BCT			E		12-0-6-9	0100 0110
71	47	BC			F		12-0-7-9	0100 0111
72	48	LH			G		12-0-8-9	0100 1000
73	49	CH			H		12-1-8	0100 1001
74	4A	AH		¢	¢		12-2-8	0100 1010
75	4B	SH	.	.	.		12-3-8	0100 1011
76	4C	MH	¤)	<	<	B A 8 4	12-4-8	0100 1100
77	4D		[	(	(	B A 8 4 1	12-5-8	0100 1101
78	4E	CVD	<	+	+	B A 8 4 2	12-6-8	0100 1110
79	4F	CVB	$	|	|	B A 8 4 2 1	12-7-8	0100 1111
80	50	ST	& +	&	&	B A	12	0101 0000
81	51				P		12-11-1-9	0101 0001
82	52				Q		12-11-2-9	0101 0010
83	53				R		12-11-3-9	0101 0011
84	54	N			S		12-11-4-9	0101 0100
85	55	CL			T		12-11-5-9	0101 0101
86	56	O			U		12-11-6-9	0101 0110
87	57	X			V		12-11-7-9	0101 0111
88	58	L			W		12-11-8-9	0101 1000
89	59	C			X		11-1-8	0101 1001
90	5A	A		!	Y		11-2-8	0101 1010
91	5B	S	$	$	Z	B 8 2 1	11-3-8	0101 1011
92	5C	M	*	*	[	B 8 4	11-4-8	0101 1100
93	5D	D	]	)	)	B 8 4 1	11-5-8	0101 1101
94	5E	AL	:	:	]	B 8 4 2	11-6-8	0101 1110
95	5F	SL	Δ	¬	¬	B 8 4 2 1	11-7-8	0101 1111
96	60	STD	-	-	-	B	11	0110 0000
97	61		/	/	/	A 1	0-1	0110 0001
98	62				b		11-0-2-9	0110 0010
99	63				c		11-0-3-9	0110 0011
100	64				d		11-0-4-9	0110 0100
101	65				e		11-0-5-9	0110 0101
102	66				f		11-0-6-9	0110 0110
103	67	MXD			g		11-0-7-9	0110 0111
104	68	LD			h		11-0-8-9	0110 1000
105	69	CD			i		0-1-8	0110 1001
106	6A	AD		¦	j		12-11	0110 1010
107	6B	SD	,	,	k	A 8 2 1	0-3-8	0110 1011
108	6C	MD	%(	%	%	A 8 4	0-4-8	0110 1100
109	6D	DD	¥	_	_	A 8 4 1	0-5-8	0110 1101
110	6E	AW	\	>	>	A 8 4 2	0-6-8	0110 1110
111	6F	SW	**	?	?	A 8 4 2 1	0-7-8	0110 1111
112	70	STE			p		12-11-0	0111 0000
113	71				q		12-11-0-1-9	0111 0001
114	72				r		12-11-0-2-9	0111 0010
115	73				s		12-11-0-3-9	0111 0011
116	74				t		12-11-0-4-9	0111 0100
117	75				u		12-11-0-5-9	0111 0101
118	76				v		12-11-0-6-9	0111 0110
119	77				w		12-11-0-7-9	0111 0111
120	78	LE			x		12-11-0-8-9	0111 1000
121	79	CE		`	y		1-8	0111 1001
122	7A	AE	↑	:	:	A	2-8	0111 1010
123	7B	SE	# =	#	#	8 2 1	3-8	0111 1011
124	7C	ME	@ '	@	@	8 4	4-8	0111 1100
125	7D	DE	:	'	'	8 4 1	5-8	0111 1101
126	7E	AU	>	=	=	8 4 2	6-8	0111 1110
127	7F	SU	✓	"	"	DEL 8 4 2 1	7-8	0111 1111

CODE TRANSLATION TABLE (Contd)

Dec.	Hex	Instruction and Format	Graphics and Controls BCDIC	EBCDIC(1)	ASCII	7-Track Tape BCDIC(2)	Card Code EBCDIC	Binary
128	80	SSM -S					12-0-1-8	1000 0000
129	81		a	a			12-0-1	1000 0001
130	82	LPSW -S	b	b			12-0-2	1000 0010
131	83	Diagnose	c	c			12-0-3	1000 0011
132	84	WRD S1	d	d			12-0-4	1000 0100
133	85	RDD	e	e			12-0-5	1000 0101
134	86	BXH	f	f			12-0-6	1000 0110
135	87	BXLE	g	g			12-0-7	1000 0111
136	88	SRL	h	h			12-0-8	1000 1000
137	89	SLL	i	i			12-0-9	1000 1001
138	8A	SRA					12-0-2-8	1000 1010
139	8B	SLA RS		‹			12-0-3-8	1000 1011
140	8C	SRDL		≤			12-0-4-8	1000 1100
141	8D	SLDL		(			12-0-5-8	1000 1101
142	8E	SRDA		+			12-0-6-8	1000 1110
143	8F	SLDA		+			12-0-7-8	1000 1111
144	90	STM					12-11-1-8	1001 0000
145	91	TM S1	j	j			12-11-1	1001 0001
146	92	MVI	k	k			12-11-2	1001 0010
147	93	TS -S	l	l			12-11-3	1001 0011
148	94	NI	m	m			12-11-4	1001 0100
149	95	CLI S1	n	n			12-11-5	1001 0101
150	96	OI	o	o			12-11-6	1001 0110
151	97	XI	p	p			12-11-7	1001 0111
152	98	LM -RS	q	q			12-11-8	1001 1000
153	99		r	r			12-11-9	1001 1001
154	9A						12-11-2-8	1001 1010
155	9B			)			12-11-3-8	1001 1011
156	9C	SIO, SIOF S		¤			12-11-4-8	1001 1100
157	9D	TIO, CLRIO		!			12-11-5-8	1001 1101
158	9E	HIO, HDV		±			12-11-6-8	1001 1110
159	9F	TCH		●			12-11-7-8	1001 1111
160	A0			-			11-0-1-8	1010 0000
161	A1		_	°			11-0-1	1010 0001
162	A2		s	s			11-0-2	1010 0010
163	A3		t	t			11-0-3	1010 0011
164	A4		u	u			11-0-4	1010 0100
165	A5		v	v			11-0-5	1010 0101
166	A6		w	w			11-0-6	1010 0110
167	A7		x	x			11-0-7	1010 0111
168	A8		y	y			11-0-8	1010 1000
169	A9		z	z			11-0-9	1010 1001
170	AA						11-0-2-8	1010 1010
171	AB			⌊			11-0-3-8	1010 1011
172	AC	STNSM S1		⌈			11-0-4-8	1010 1100
173	AD	STOSM		[			11-0-5-8	1010 1101
174	AE	SIGP -RS		≥			11-0-6-8	1010 1110
175	AF	MC -S1		●			11-0-7-8	1010 1111
176	B0			0			12-11-0-1-8	1011 0000
177	B1	LRA -RX		1			12-11-0-1	1011 0001
178	B2	See below		2			12-11-0-2	1011 0010
179	B3			3			12-11-0-3	1011 0011
180	B4			4			12-11-0-4	1011 0100
181	B5			5			12-11-0-5	1011 0101
182	B6	STCTL RS		6			12-11-0-6	1011 0110
183	B7	LCTL		7			12-11-0-7	1011 0111
184	B8			8			12-11-0-8	1011 1000
185	B9			9			12-11-0-9	1011 1001
186	BA	CS RS					12-11-0-2-8	1011 1010
187	BB	CDS		⌋			12-11-0-3-8	1011 1011
188	BC			¬			12-11-0-4-8	1011 1100
189	BD	CLM		]			12-11-0-5-8	1011 1101
190	BE	STCM RS		+			12-11-0-6-8	1011 1110
191	BF	ICM		—			12-11-0-7-8	1011 1111

Op code (S format)

B202 - STIDP	B207 - STCKC	B200 - PTLB
B203 - STIDC	B208 - SPT	B210 - SPX
B204 - SCK	B209 - STPT	B211 - STPX
B205 - STCK	B20A - SPKA	B212 - STAP
B206 - SCKC	B20B - IPK	B213 - RRB

CODE TRANSLATION TABLE (Contd)

Dec.	Hex	Instruction (SS)	Graphics and Controls BCDIC	EBCDIC(1)	ASCII	7-Track Tape BCDIC(2)	Card Code EBCDIC	Binary
192	C0		?	{		B A 8 2	12-0	1100 0000
193	C1		A	A	A	B A 1	12-1	1100 0001
194	C2		B	B	B	B A 2	12-2	1100 0010
195	C3		C	C	C	B A 2 1	12-3	1100 0011
196	C4		D	D	D	B A 4	12-4	1100 0100
197	C5		E	E	E	B A 4 1	12-5	1100 0101
198	C6		F	F	F	B A 4 2	12-6	1100 0110
199	C7		G	G	G	B A 4 2 1	12-7	1100 0111
200	C8		H	H	H	B A 8	12-8	1100 1000
201	C9		I	I	I	B A 8 1	12-9	1100 1001
202	CA						12-0-2-8-9	1100 1010
203	CB						12-0-3-8-9	1100 1011
204	CC			ƒ			12-0-4-8-9	1100 1100
205	CD						12-0-5-8-9	1100 1101
206	CE			Y			12-0-6-8-9	1100 1110
207	CF						12-0-7-8-9	1100 1111
208	D0		!	}		B 8 2	11-0	1101 0000
209	D1	MVN	J	J	J	B 1	11-1	1101 0001
210	D2	MVC	K	K	K	B 2	11-2	1101 0010
211	D3	MVZ	L	L	L	B 2 1	11-3	1101 0011
212	D4	NC	M	M	M	B 4	11-4	1101 0100
213	D5	CLC	N	N	N	B 4 1	11-5	1101 0101
214	D6	OC	O	O	O	B 4 2	11-6	1101 0110
215	D7	XC	P	P	P	B 4 2 1	11-7	1101 0111
216	D8		Q	Q	Q	B 8	11-8	1101 1000
217	D9		R	R	R	B 8 1	11-9	1101 1001
218	DA						12-11-2-8-9	1101 1010
219	DB						12-11-3-8-9	1101 1011
220	DC	TR					12-11-4-8-9	1101 1100
221	DD	TRT					12-11-5-8-9	1101 1101
222	DE	ED					12-11-6-8-9	1101 1110
223	DF	EDMK					12-11-7-8-9	1101 1111
224	E0		+	\		A 8 2	0-2-8	1110 0000
225	E1						11-0-1-9 —	1110 0001
226	E2		S	S	S	A 2	0-2	1110 0010
227	E3		T	T	T	A 2 1	0-3	1110 0011
228	E4		U	U	U	A 4	0-4	1110 0100
229	E5		V	V	V	A 4 1	0-5	1110 0101
230	E6		W	W	W	A 4 2	0-6	1110 0110
231	E7		X	X	X	A 4 2 1	0-7	1110 0111
232	E8		Y	Y	Y	A 8	0-8	1110 1000
233	E9		Z	Z	Z	A 8 1	0-9	1110 1001
234	EA						11-0-2-8-9	1110 1010
235	EB						11-0-3-8-9	1110 1011
236	EC			⌈			11-0-4-8-9	1110 1100
237	ED						11-0-5-8-9	1110 1101
238	EE						11-0-6-8-9	1110 1110
239	EF						11-0-7-8-9	1110 1111
240	F0	SRP	0	0	0	8 2	0	1111 0000
241	F1	MVO	1	1	1	1	1	1111 0001
242	F2	PACK	2	2	2	2	2	1111 0010
243	F3	UNPK	3	3	3	2 1	3	1111 0011
244	F4		4	4	4	4	4	1111 0100
245	F5		5	5	5	4 1	5	1111 0101
246	F6		6	6	6	4 2	6	1111 0110
247	F7		7	7	7	4 2 1	7	1111 0111
248	F8	ZAP	8	8	8	8	8	1111 1000
249	F9	CP	9	9	9	8 1	9	1111 1001
250	FA	AP		]			12-11-0-2-8-9	1111 1010
251	FB	SP					12-11-0-3-8-9	1111 1011
252	FC	MP					12-11-0-4-8-9	1111 1100
253	FD	DP					12-11-0-5-8-9	1111 1101
254	FE						12-11-0-6-8-9	1111 1110
255	FF			EO			12-11-0-7-8-9	1111 1111

ANSI-DEFINED PRINTER CONTROL CHARACTERS
(A in RECFM field of DCB)

Code	Action before printing record
blank	Space 1 line
0	Space 2 lines
–	Space 3 lines
+	Suppress space
1	Skip to line 1 on new page

APPENDIX - II

CONTENTS OF CONTROL REGISTERS

Word	Bits	Name of Field	Function	Value after System Reset
0	0	Block multiplex mode	Block multiplexing control	0
0	24	Timer mask	Extended external masking	1
0	25	Interrupt key mask	Extended external masking	1
0	26	External signal mask	Extended external masking	1
2	0-31	Channel masks	Extended IO masking	1
8	16-31	Monitor masks	Monitoring	0
14	0	Hard stop mode	Machine check handling	1
14	1	Synchronous MCEL mask	Machine check handling	1
14	2	IO extended logout	Machine check handling	0
14	4	Recovery report mask	Machine check handling	0
14	5	Configuration report mask	Machine check handling	0
14	6	External damage report mask	Machine check handling	1
14	7	Warning mask	Machine check handling	0
14	8	Asynchronous MCEL mask	Machine check handling	0
14	9	Asynchronous fixed log mask	Machine check handling	0
15	8-31	MCEL pointer	Machine check handling	512

MCEL: Machine Check Extended Logout

APPENDIX-III

INTERRUPTION CODE FOR PROGRAM INTERRUPTION

	Interruption Code (bits 24-31 of old PSW)	Program Interruption Cause
1	00000001	Operation
2	00000010	Privileged operation
3	00000011	Execute
4	00000100	Protection
5	00000101	Addressing
6	00000110	Specification
7	00000111	Data
8	00001000	Fixed-point overflow
9	00001001	Fixed-point divide
10	00001010	Decimal overflow
11	00001011	Decimal divide
12	00001100	Exponent overflow
13	00001101	Exponent underflow
14	00001110	Significance
15	00001111	Floating-point divide
	10000000	Monitoring

APPENDIX - IV

PROGRAM STATUS WORDS

0–5 Channel Mask

Bits 0 to 5 are assigned to channels 0 to 5. When a channel mask bit is set, I/O interruptions are enabled for the respective channel. If a bit is off (zero), interruptions are disabled for that channel. The interruption conditions remain pending.

6 Input/Output Mask

1 = Interruptions are enabled for channel 6 and above.
0 = Interruptions are disabled for channel 6 and above.
These channels are not available on the Model 115.

7 External Mask

1 = Interruptions are enabled from the following external sources:

- Interval timer
- CPU timer
- Clock comparator
- Interrupt key on console
- External signals

0 = External interruptions are disabled.
Note: CPU timer and clock comparator interruption conditions remain pending only if no new values are set before the interruption is taken.

8–11 Key

This is a binary key which is compared with a key in storage when the CPU stores a result or fetches data from a fetch protected location. Fetching and storing only succeed if these two keys match or the PSW key is zero.

12 Extended Control Mode

1 = EC mode is set and the PSW bits are interpreted as shown in Figure 15.
0 = Basic control (BC) mode is set and the PSW bits are interpreted as shown in this Figure.

13 Machine Check Mask

1 = Interruptions due to machine checks (such as parity errors, system, processing, or timer damage) are enabled.
0 = Interruptions due to suppressible machine checks are disabled. They remain pending.

14 Wait State

1 - The CPU is in the wait state (no instruction processing by MIP and no CPU meter recording).
0 - The CPU is in the running state.

15 Problem State

1 The CPU is in the problem state, and only unprivileged instructions are executed.
0 The CPU is in the supervisor state, and both privileged and unprivileged instructions are executed.

16–31 Interruption Code

This is a binary code which identifies the source of an interruption.

32–33 Instruction Length Code

This is a binary code which shows the length of the last interpreted instruction (1, 2, or 3 halfwords) when a program or supervisor call interruption occurs.

34–35 Condition Code

This is a binary number set by the results of various instructions, so that branching decisions can be made.

36–39 Program Mask

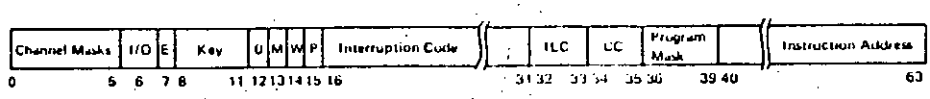
When set the program mask bits enable interruptions due to:

- Fixed point overflow (bit 36)
- Decimal overflow (bit 37)
- Exponent underflow (bit 38)
- Significance (bit 39)

If the bits are 0, the relevant interruptions are disabled.

40–63 Instruction Address

This is a binary field representing the main storage address of the next instruction to be executed.



Program Status Word (BC Mode)

1 Program Event Recording Mask

1 = Program events specified in control register 9 (such as successful branching) will cause an interruption.
0 = Program event recording is disabled.

5 Translation Mode

1 = Dynamic address translation is used (the address base + displacement is translated via segment and page tables into the real storage address).
0 = Address translation is not used.

6 Input/Output Mask

1 = I/O interruptions are enabled for all channels whose mask bits in control register 2 are set.
0 = All I/O interruptions are disabled, and requests remain pending.

7 External Mask

1 = External interruptions are enabled for all sources whose mask bits in control register 0 are set.
0 = External interruptions are disabled.

8–11 Key

This is a binary key which is compared with a key in storage when the CPU stores a result or fetches data from a fetch-protected location. Fetching and storing only succeed if the keys match or the PSW key is zero.

12 Extended Control Mode

1 = Model 115 is in EC mode and the PSW bits are interpreted as shown in this Figure.
0 = Model 115 is in BC mode and the PSW bits are interpreted as shown in Figure 14.

13 Machine Check Mask

1 = Machine check interruptions are enabled for:
- System and processing damage.
- All other machine checks whose mask bits in control register 14 are set.

0 = Suppressible machine check interruptions are disabled.

14 Wait State

1 = The CPU is in the wait state (no instruction processing by MIP and no CPU meter recording).
0 = The CPU is in the running state.

15 Problem State

1 = The CPU is in the problem state, and only privileged instructions are executed.
0 = The CPU is in the supervisor state, and both privileged and unprivileged instructions are executed.

18,19 Condition Code

This is a binary number set by the results of various instructions, so that branching decisions can be made.

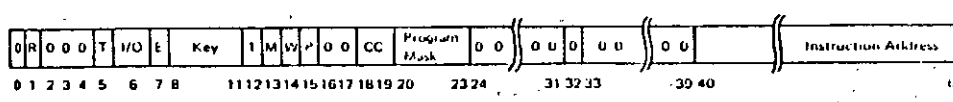
20–23 Program Mask

When set, these bits enable interruptions due to
- Fixed point overflow (bit 20)
- Decimal overflow (bit 21)
- Exponent underflow (bit 22)
- Significance (bit 23)

If a bit is 0, relevant interruption is disabled (the condition remains pending).

40–63 Instruction Address

This is a binary field representing the main storage address of the next instruction to be executed.



Program Status Word (EC Mode)

APPENDIX - V

STRUCTURED EBCDIC CODES

Bits 2–7	Graphic	EBCDIC	ASCII
00 0000	SP	40	20
00 0001	A	C1	41
00 0010	B	C2	42
00 0011	C	C3	43
00 0100	D	C4	44
00 0101	E	C5	45
00 0110	F	C6	46
00 0111	G	C7	47
00 1000	H	C8	48
00 1001	I	C9	49
00 1010	¢	4A	–
	[	–	5B
00 1011		4B	2E
00 1100	<	4C	3C
00 1101	(	4D	28
00 1110	+	4E	2B
00 1111	!	4F	–
	]	–	21
01 0000	&	50	26
01 0001	J	D1	4A
01 0010	K	D2	4B
01 0011	L	D3	4C
01 0100	M	D4	4D
01 0101	N	D5	4E
01 0110	O	D6	4F
01 0111	P	D7	50
01 1000	Q	D8	51
01 1001	R	D9	52
01 1010	!	5A	–
	]	–	5D
01 1011	$	5B	24
01 1100	*	5C	2A
01 1101	)	5D	29
01 1110	;	5E	3B
01 1111	¬	5F	
	^	–	5E

Bits 2–7	Graphic	EBCDIC	ASCII
10 0000	-	60	2D
10 0001	/	61	2F
10 0010	S	E2	53
10 0011	T	E3	54
10 0100	U	E4	55
10 0101	V	E5	56
10 0110	W	E6	57
10 0111	X	E7	58
10 1000	Y	E8	59
10 1001	Z	E9	5A
10 1010	¦ (EBCDIC)	6A	7C
10 1011	,	6B	2C
10 1100	%	6C	25
10 1101	–	6D	5F
10 1110	>	6E	3E
10 1111	?	6F	3F
11 0000	0	F0	30
11 0001	1	F1	31
11 0010	2	F2	32
11 0011	3	F3	33
11 0100	4	F4	34
11 0101	5	F5	35
11 0110	6	F6	36
11 0111	7	F7	37
11 1000	8	F8	38
11 1001	9	F9	39
11 1010	:	7A	3A
11 1011	#	7B	23
11 1100	@	7C	40
11 1101	'	7D	27
11 1110	=	7E	3D
11 1111	"	7F	22

Note: The characters above are used as attribute, AID, write control (WCC), copy control (CCC), CU and device address, and buffer address. They are also used as status and sense, except by the 3274 and 3276 when operating in BSC. When any of these characters is transmitted to the program, the CU assigns the appropriate EBCDIC code. If transmission is in ASCII, the CU translates the EBCDIC code to ASCII code prior to transmission.

To use this table to determine the hex code transmitted for an address or control character, first determine the values of bits 2–7. Select this bit configuration from the "Bits 2–7" column. The hex code that will be transmitted (either in EBCDIC or in ASCII) is to the right of the bit configuration.

Use this table also to determine equivalent EBCDIC and ASCII hex codes and their associated graphic characters. See Figure 2-4, Note 5, for ASCII A and B graphic character difference for ASCII codes 21 and 5E (hex).

Graphic characters for the United States I/O interface codes are shown. Graphic characters might differ for particular World Trade I/O interface codes. Refer to IBM 3270 Information Display System: Character Set Reference, *GA27-2837, for possible graphic differences when these codes are used.*

APPENDIX VI

A SAMPLE DATA ACQUISITION PROGRAM WITH OUTPUT

```
10 DIM B$(64),AB(20)
20 FOR I=0  TO 63
30 READ B$(I)
40 NEXT I
50 AD=0
60 OUT 634,4    'Set all HI
70 OUT 634,6    'Clear address Counter
80 OUT 634,4    'Set all HI
90 OUT 634,0    'Start transmission from CU
100 OUT 634,4   'Set all HI
110 REM wait few seconds
120 PRINT "At end of transmission Hit any key to resume"
130 V$=INKEY$:IF V$="" THEN 130
140 REM IF V$<>CHR$(27) THEN 60
150 PRINT " 1   2   3   4   5   6   7   8   9  10  11  12  13  CU/VT
160 OUT 634,4   'Set all HI
170 OUT 634,6   'Clear address counter
180 OUT 634,4   'Set all HI
190 OUT 634,5   'Set BYTE SELECT LO
200 A=INP(632)  'Read Lo byte
210 SA=A        'Save it.
220 OUT 634,4   'Set BYTE SELECT HI
230 B=INP(632)  'Read Hi byte
240 SB=B        'Save it
250 IF B AND  16 THEN AB(1)=1 ELSE AB(1)=0
260 IF B AND   8 THEN AB(2)=1 ELSE AB(2)=0
270 IF B AND   4 THEN AB(3)=1 ELSE AB(3)=0
280 IF B AND   2 THEN AB(4)=1 ELSE AB(4)=0
290 IF B AND   1 THEN AB(5)=1 ELSE AB(5)=0
300 IF A AND 128 THEN AB(6)=1 ELSE AB(6)=0
310 IF A AND  64 THEN AB(7)=1 ELSE AB(7)=0
320 IF B AND  64 THEN AB(8)=1 ELSE AB(8)=0
330 IF A AND  16 THEN AB(9)=1 ELSE AB(9)=0
340 IF A AND   8 THEN AB(10)=1 ELSE AB(10)=0
350 IF A AND   4 THEN AB(11)=1 ELSE AB(11)=0
360 IF A AND   2 THEN AB(12)=1 ELSE AB(12)=0
370 IF A AND   1 THEN AB(13)=1 ELSE AB(13)=0
380 IF AB(1)=1 AND AB(4)=0 THEN GOSUB 520:GOTO 410
390 IF AB(1)=0 AND AB(3)=0 THEN GOSUB 540:GOTO 410
400 GOTO 480
410 FOR I=1 TO 13:PRINT AB(I);" ";:NEXT I
420 IF B AND 128 THEN PRINT "  1 "; ELSE PRINT "  0";
430 PRINT "    ";S$;"    ";
440 REM IF B AND 128 THEN PRINT "  1 "; ELSE PRINT "  0";
450 PRINT TAB(75);AD
460 F$=INKEY$:IF F$="" THEN 460
470 IF F$=CHR$(27) THEN END
480 OUT 634,12      'Increment address counter
490 OUT 634,4       'Set all HI
500 AD=AD+1
```

```
510 GOTO 190
520 CHA=AB(6)*32+AB(7)*16+AB(8)*8+AB(9)*4+AB(10)*2+AB(11)*1
530 GOTO 550
540 CHA=AB(5)*32+AB(6)*16+AB(7)*8+AB(8)*4+AB( 9)*2+AB(10)*1
550 S$=B$(CHA)
560 RETURN
570 DATA " "
580 DATA A,B,C,D,E,F,G,H,I
590 DATA [,.,<,(,+,!
600 DATA "&"
610 DATA J,K,L,M,N,O,P,Q,R
620 DATA ],$,*,)
630 DATA ";"
640 DATA "{"
650 DATA -,/
660 DATA S,T,U,V,W,X,Y,Z
670 DATA ¦
680 DATA ","
690 DATA %,_,>,?
700 DATA 0,1,2,3,4,5,6,7,8,9
710 DATA ":"
720 DATA "#"
730 DATA "@"
740 DATA "'"
750 DATA =
760 DATA "}"
```

OUTPUT OF THE DATA ACQUISITION PROGRAM

RUN
At end of transmission Hit any key to resume

1	2	3	4	5	6	7	8	9	10	11	12	13	CU/VT	
1	0	1	0	1	0	0	0	0	0	0	0	0	1	
1	0	0	0	0	0	0	0	0	0	0	0	0	1	
1	0	0	0	0	0	0	0	0	0	0	0	0	1	
1	0	0	0	0	0	0	0	0	0	0	0	0	1	
1	0	0	0	0	0	0	0	0	0	0	0	0	1	
1	0	0	0	1	0	1	0	0	1	1	0	0	1	L
1	0	0	0	1	0	0	1	0	0	1	1	0	1	I
1	0	0	0	1	0	1	0	1	0	1	0	0	1	N
1	0	0	0	1	0	1	0	0	1	0	1	0	1	K
1	0	0	0	1	0	0	0	0	0	1	0	0	1	A
1	0	0	0	1	0	0	0	1	1	1	0	0	1	G
1	0	0	0	1	0	0	0	1	0	1	1	0	1	E
1	0	0	0	1	1	0	0	0	0	0	0	0	1	-
1	0	0	0	1	0	0	0	1	0	1	1	0	1	E
1	0	0	0	1	0	0	0	1	0	0	0	0	1	D
1	0	0	0	1	0	0	1	0	0	1	1	0	1	I
1	0	0	0	1	1	0	0	0	1	1	0	0	1	T
1	0	0	0	1	0	0	0	1	0	1	1	0	1	E
1	0	0	0	1	0	0	0	1	0	0	0	0	1	D
1	0	0	0	1	0	0	0	0	0	0	1	0	1	
1	0	0	0	1	1	0	0	0	1	1	0	0	1	T
1	0	0	0	1	0	1	0	1	1	0	0	0	1	O
1	0	0	0	1	0	0	0	0	0	0	1	0	1	
1	0	0	0	1	1	0	0	0	1	1	0	0	1	T
1	0	0	0	1	0	0	1	0	0	0	0	0	1	H
1	0	0	0	1	0	0	0	1	0	1	I	0	1	E
1	0	0	0	1	0	0	0	0	0	0	1	0	1	
1	0	0	0	1	0	0	0	0	1	1	1	0	1	C
1	0	0	0	1	0	1	0	1	1	0	0	0	1	O
1	0	0	0	1	0	1	1	0	0	1	0	0	1	R
1	0	0	0	1	0	0	0	1	0	1	1	0	1	E
1	0	0	0	1	0	0	0	0	0	0	1	0	1	
1	0	0	0	1	0	0	1	0	0	1	1	0	1	I
1	0	0	0	1	0	1	0	1	0	0	1	0	1	M

APPENDIX VII

INTERFACE PROGRAM

```
10 DIM B$(252)
20 FOR I=0  TO 248 STEP 4
30 READ B$(I)
40 NEXT I
50 B$(252)=CHR$(34)
60 CLS
70 SW=0
80 LOCATE 3,10:PRINT"/# in columns 1-2 indicates Start-of-Data "
90 LOCATE 5,10:PRINT"#/ in columns 1-2 indicates End-of-Data   "
100 LOCATE 12,10: PRINT "Enter Output file specification please :"
110 PLAY "a"
120 LOCATE 12,50:INPUT " ",FIL$
130 IF FIL$="" THEN 100
140 OPEN "O",#3,FIL$
150 LN=0
160 AD=160
170 LC=0
180 OUT 634,4   'Set all HI
190 OUT 634,6   'Clear address Counter
200 OUT 634,4   'Set all HI
210 OUT 634,0   'Start transmission from CU
220 OUT 634,4   'Set all HI
230 LOCATE 12,10:PRINT"Stand by for receiving data from Terminal
"
240 TC=INP(633)
250 IF TC AND 64 THEN 260 ELSE GOTO 240
260 LOCATE 13,10:PRINT"
    "
270 LOCATE 12,10:PRINT "Data filing in process : Please  hold on
"
280 OUT 634,4   'Set all HI
290 OUT 634,6   'Clear address counter
300 OUT 634,4   'Set all HI
310 FOR ADC=1 TO 160
320 OUT 634,12
330 OUT 634,4
340 NEXT ADC
350 IF SW=111 THEN 470
360 GOSUB 830
370 IF AD=1760 THEN 520
380 IF S$<>"/" THEN GOTO 360
390 GOSUB 830
400 IF S$<>"#" THEN 360
410 SW=111
420 FOR ADC=1 TO 78
430 OUT 634,12
440 OUT 634,4
450 NEXT ADC
460 AD=AD+78
470 GOSUB 830
480 PRINT #3,S$; :LC=LC+1
```

```
490 IF LC<80 THEN 470
500 PRINT #3,CHR$(13);:LC=0:LN=LN+1
510 IF AD<1760 THEN GOTO 560
520 LOCATE 11,10:PRINT "      ";LN;" Lines transferred
   "
530 PLAY "a"
540 LOCATE 13,10 :PRINT "Please hit ENTER key on the TERMINAL for next
550 GOTO 160
560 GOSUB 830
570 IF S$<>"#" THEN 480
580 GOSUB 830
590 IF S$="/" THEN 620
600 PRINT #3,"#";S$; :LC=LC+2
610 GOTO 470
620 LOCATE 11,10:PRINT"      ";LN;" Lines transfered
630 LOCATE 12,10:PRINT"End-of-Data condition reached
640 PLAY "a"
650 LOCATE 13,10:PRINT "Do you want to terminate transmission (Y/N)
"
660 LOCATE 13,55
670 V$=INKEY$
680 IF V$="Y" THEN 800
690 IF V$<>"N" THEN 670
700 LOCATE 13,10:PRINT"Do you want to continue with ";FIL$;" (Y/N)

710 PLAY "a"
720 V$=INKEY$
730 IF V$="Y" THEN SW=0:LOCATE 13,10:PRINT"
                ":GOTO 160
740 IF V$<>"N" THEN 720
750 CLOSE #3
760 LOCATE 11,10:PRINT"
770 LOCATE 13,10:PRINT"
780 SW=0
790 GOTO 100
800 LOCATE 14,10:PRINT"End of transmission : Thank You                    "
810 CLOSE #3
820 END
830 REM
840 OUT 634,5    'Set BYTE SELECT LO
850 A=INP(632)   'Read Lo byte
860 OUT 634,4    'Set BYTE SELECT HI
870 B=INP(632)   'Read Hi byte
880 IF B AND 2 THEN S$=CHR$(32): GOTO 920
890 A=A AND 220
900 IF B AND 64 THEN A=A OR 32
910 S$=B$(A)
920 OUT 634,12      'Increment address counter
930 OUT 634,4       'Set all HI
940 AD=AD+1
950 RETURN
960 DATA " "
```

```
970 DATA A,B,C,D,E,F,G,H,I
980 DATA [,.,<,(,+,!
990 DATA "&"
1000 DATA J,K,L,M,N,O,P,Q,R
1010 DATA ],$,*,)
1020 DATA ";"
1030 DATA "{"
1040 DATA -,/
1050 DATA S,T,U,V,W,X,Y,Z
1060 DATA ¦
1070 DATA ","
1080 DATA %,_,>,?
1090 DATA 0,1,2,3,4,5,6,7,8,9
1100 DATA ":"
1110 DATA "#"
1120 DATA "@"
1130 DATA "'"
1140 DATA =
```

REFERENCES

1. Abramson Naorman,Kuo Franklin F. - editors; Computer-Communication Networks, Prentice-Hall, Inc (1973)

2. Brenner Robert C; IBM PC Trouldeshooting & Repair Guide, Howard W Sams & Co Inc. Indiana, U.S.A.(1985)

3. Burian B.J.; A simplified Approach to S/370 Assembly Language Programming. N.J. (1977)

4. Davis Donald W., Barber Derek L.A.; Communication Networks for Computers, National Physical Laboratory (1977)

5. Doll Dixon R.; Data Communications, John Wiley & Sons (1978)

6. Flores I., Computer Organization, Prentice-Hall Inc, N.J.(1969)

7. Hakimi S.L.; Simultaneous flows through Communication Networks, IRE Trans, Circuit Theory (1962)

8. Karen Hary, Computer Organization and Architecture of System/370.

9. Longley D. and Shain M.; Expanding and Networking Microcomputers, MacMillan Press, London (1985)

10. Madnick S.E. and Donovan J.J.; Operating Systems, McGraw-Hill Book Company (1974)

11. Sanders Donald H.; Computers Today, McGraw-Hill, Book Company, N.Y. (1985)

12. IBM Corporation; Introduction to DOS/VS. GC33-5370-5, N.Y.(1977)

13. IBM Corporation; IBM System/370 Principals of Operation, GA-7000-5, New York (1976)

14. IBM Corporation; IBM System/370 Model 115 Functional Characteristics, GA33-1510-2, New York (1977)

15. IBM Corporation; IBM 3270 Information Display System Component Description, GA27-2749-10, New York (1980)

16. IBM Corporation Entry Time Sharing System, User's Guide, SB21-2122-0, New York.

17. IBM Corporation; IBM Maintenance Handbook, 5229-7037-4, New York.