ICL System Ten
Model 22 Processor

Reference

ICL endeavours to ensure that the information in this document is correct and fairly stated, but does not accept liability for any error or omission.

The development of ICL products and services is continuous and published information may not be up-to-date. Any particular issue of a product may contain part only of the facilities described in this document or may contain facilities not described here. It is important to check the current position with ICL.

Specifications and statements as to performance in this document are ICL estimates intended for general guidance. They may require adjustment in particular circumstances and are therefore not formal offers or undertakings.

Statements in this document are not part of a contract or program product licence save insofar as they are incorporated into a contract or licence by express reference. Issue of this document does not entitle the recipient to access to or use of the products described, and such access or use may be subject to separate contracts or licences.

From 1 October 1976, International Computers Limited has acquired certain of the computer activities previously carried on by the Singer Company.

The products and services now offered by ICL may differ in detail from those formerly offered by your Singer supplier. Readers should therefore consult their local ICL office to confirm the current availability and support status of any products or services referenced in this publication.

Technical Publication 7529

First Edition June 1977

ICL will be pleased to receive readers' views on the contents and organisation, etc. of this publication. Please write to

The Registry (Readership Survey)
UK Software and Literature Distribution Centre
International Computers Limited
60 Portman Road
Reading
Berks RG3 1NR

Distributed by
UK Software and Literature Distribution Centre
International Computers Limited
Registered Office: ICL House, Putney, London SW15 1SW
Printed by ICL Printing Services
Works Road, Letchworth, Herts SG6 1JY

Preface

This publication describes the hardware and method of operation of the System Ten Model 22 processor.

Chapter 1 describes the components of the machine, and Chapter 2 describes the way that core store is organised and used. The machine instructions are detailed in Chapter 3 and the Input/Output controllers are described in Chapter 4. Appendix 1 provides a convenient reference to the System Ten character set.

Contents

The text of this publication is divided into chapters in the normal way, and each chapter is subdivided into sections. A section's level in the hierarchy is indicated by its number. Therefore, within Chapter *n*, first level section headings are numbered *n*.1, *n*.2 and so on; second level headings are numbered *n*.1.1, *n*.1.2 . . . *n*.2.1 and so on; third level headings are numbered *n*.1.1.1, *n*.1.1.2 . . . *n*.1.2.1 and so on.

The contents list and index, and cross-references in the text, all refer to section numbers.

Pages are numbered within chapters, in the form *c-p*, where *c* is the chapter number and *p* the page number within that chapter. Figures and tables, where they appear, are also numbered within chapters, so that Figure *n*.2 is the second figure in Chapter *n*, and Table *n*.2 is the second table in that chapter.

Section numbers, page numbers and figure and table numbers in appendices are preceded by the letter A.

Architecture of the Model 22 processor

Chapter 1

1.1 Introduction

The System Ten computer is a multi-programming system capable of executing up to 20 independent programs concurrently. This is achieved without the overhead of an executive program: core store is divided into a shareable area of store called Common and up to 20 user partitions. A hardware switching system allocates processing time to each partition in turn.

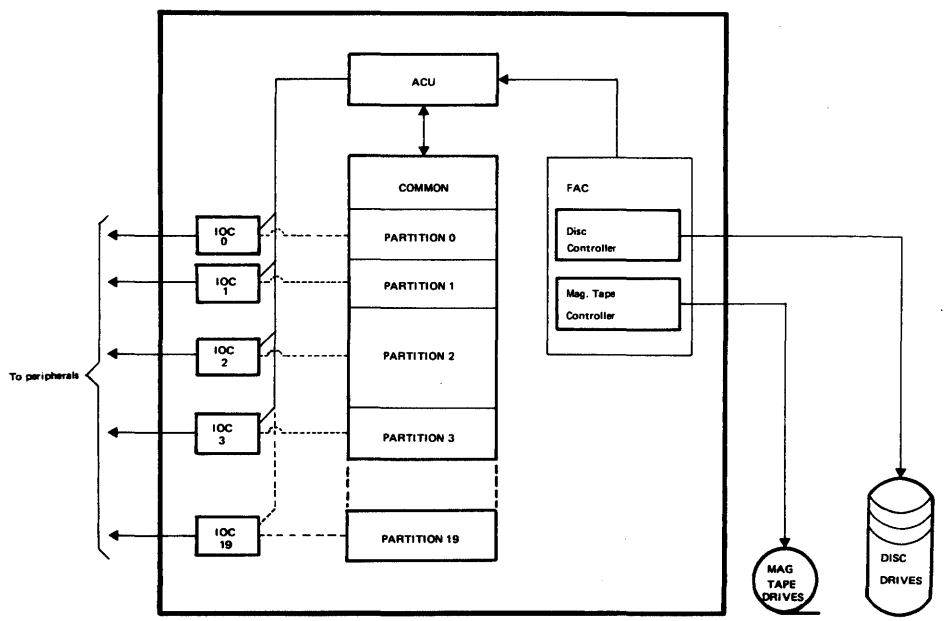


Figure 1.1 The main components of a System Ten computer and how they are logically connected

The execution of instructions and all transfers to and from core store are performed by the Arithmetic and Control Unit (ACU). Additional large volume storage is provided by magnetic tape and disc units through the File Access Channel (FAC). Communication between the ACU and other peripheral devices is by means of Input/Output Controllers (IOCs) of several types, each of which is linked to a partition. A System Ten computer therefore has one FAC and one to 20 IOCs.

The components of the System Ten computer are described in the sections that follow, and how they are logically connected is shown in Figure 1.1.

1.2 Components

1.2.1 Memory description

1.2.1.1 *Definitions*

Throughout this publication, terms have the following definitions:

Term	*Definition*
Bit	One of the two digits 0 or 1 held by one element of core storage

Term	*Definition*
Location	The smallest addressable unit of core store consisting of six bits and holding one character
Character	One symbol from the System Ten set (see Figure 2.2) represented by a unique code of six bits
Field	One or more contiguous characters which together hold an item of information
K	1000 locations

1.2.1.2 *Memory*

System Ten memory provides random access core storage for from 20K to 160K locations in modules of 20K. Each location is addressable and contains one six bit character. The highest address that can be handled, and therefore the maximum size of Common or any partition, is 80K. It is a convention that the address of a location in Common will be specified by five digits followed by the letter C; for example 29480C.

At installation, the available memory is divided between Common memory and the desired number of partitions. Common can be allocated from 1K to 80K locations in modules of 1K. Because of system requirements the minimum allocated is usually 10K.

A partition can be allocated from 1K to 10K in modules of 1K, or from 10K to the maximum of 80K in modules of 10K.

The system of memory allocation is very flexible and a user must select the number of partitions and how memory is divided between them and Common that will provide the most satisfactory performance for his application. An example configuration might be a 60K machine with memory divided into 19K Common, a 1K partition with a digital clock and four application partitions of 10K each.

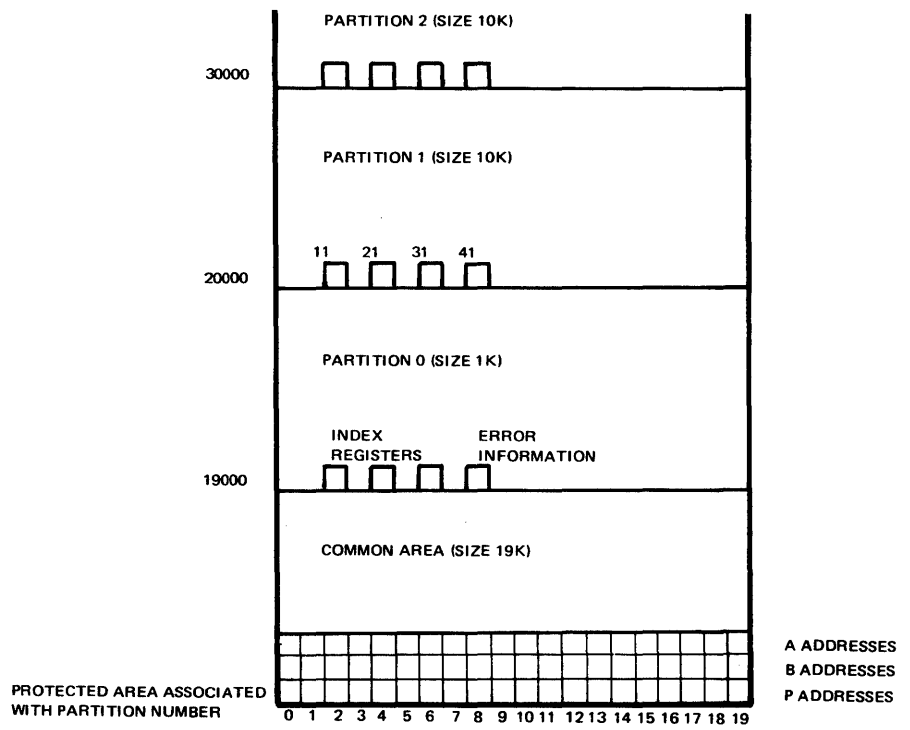


Figure 1.2 Example allocation of memory

1.2.2 The Arithmetic and Control Unit (ACU)

The Arithmetic and Control Unit (ACU) carries out three important functions that are fundamental to the running of the machine:

1 To allocate processing time to each IOC and its associated partition in turn. Each IOC in turn is selected by the ACU and monopolises the unit for (nominally) 40 milliseconds. See the Branch instruction, section 3.2.3

2 To extract each instruction and its associated data from core store; operate on the data and store the results

3 To transfer data character by character between core store and the FAC or one of the IOCs

1.2.3 The File Access Channel (FAC)

The File Access Channel (FAC) provides an interface between the ACU and magnetic tape and disc storage. Depending on the high-speed devices provided in the system, the FAC includes a Magnetic Tape Controller (handling up to four tape drives) and/or a Disc Controller (handling up to 16 logical disc devices). Unlike the IOCs, the FAC is shared. The devices attached to the FAC are directly accessible to all partitions. Thus, several programs may share the same disc or magnetic tape files.

Data is transferred between core store and disc units in groups of 100 characters, and between core store and magnetic tape in groups of up to 10,000 characters.

Note that (unlike the IOCs) the FAC retains control of the processor until a data transfer is complete. It is therefore possible for a lengthy tape transfer (for example reconstituting a disc from a tape security copy) to occupy the processor for a considerable time.

1.2.4 The Input/Output Controllers (IOCs)

The Input/Output Controllers (IOCs) provide communication between the ACU and the various peripheral devices. It is the IOCs that perform the conversion between the ASCII seven-bit character pattern and the System Ten internal six-bit pattern (see section 2.4.1). Each IOC can address any location in its own partition or above 299 Common.

The following IOCs are available:

IOC type	*User device*
Multi-terminal IOC (MTIOC II)	Workstation(typebar or Visual Display Unit (VDU), Line Printer
Multi Device IOC (MDIOC II)	Point of sale (POS) terminal, Job Information terminal, Data Collection terminal
Synchronous Communications Adaptor (SCA)	Remote computer, visual display unit (VDU) or other synchronous device
Asynchronous Communications Adaptor (ACA)	Remote asynchronous devices including computers
Asynchronous Terminal Adaptor (ATA)	Devices from Other Equipment Manufacturers (OEM) using asynchronous communications methods
Digital clock	Provides time of day information for user

1.3 How the ACU operates

The ACU performs all major functions by means of hardware logic controlled by hardware function codes which are similar to, but at a lower level than, instruction function codes. These hardware codes control the manipulation of information and its transfer between components of the processor and to and from peripherals.

Even though the general course of action of the ACU is set by the instruction function code, the ACU must step through and branch between hardware functions many times to perform the various checks and actions necessary to complete one instruction. The sequence of steps and functions is different for each instruction, and during execution the sequence is constantly being modified depending on certain conditions encountered, for example a data error or a busy or inactive peripheral.

Because of the very high speed at which the ACU operates compared with an IOC, or with peripherals, which are even slower, the ACU is able to initiate some action in an IOC or peripheral and then perhaps carry out a number of other functions while waiting for the first to be completed. In this way, the ACU is able to direct and monitor activities in many parts of the machine, and take the necessary action in order to achieve the required result with the greatest efficiency.

Some of the activities to which the ACU reacts are:

1 The power supply to the machine is constantly monitored so that no information is lost in the event of a power failure. See section 1.3.4

2 Data is transferred character by character between core store and the ACU, and between the FAC and IOCs and the ACU, whenever the necessity is indicated by an Interrupt. See section 1.3.3

3 Although it is usually the ACU that initiates a data exchange with an IOC or a peripheral, some peripherals can attract the attention of the ACU by a Service Request, see section 1.3.2

4 The validity of each instruction is checked before and during execution and any fault causes a Program Check (see section 1.3.5) or an Address Check (see section 1.3.6)

1.3.1 The Base Adder

The Base Adder is a part of the ACU that derives the actual address in core store, or *absolute* address, from the address specified in an instruction.

Each location in a partition is addressed by the partition user relative to the beginning of that partition. In order for the ACU to arrive at the absolute address, the store allocated to Common and to all the partitions numbered lower than that partition must be added to the instruction address. Addresses in Common do not use the Base Adder.

1.3.2 Service Request

Service Request is a hardware function that allows a program to recognise when a peripheral such as a terminal has information to pass.

The program will contain a Branch on Service Request instruction which can transfer control to a Read instruction.

A message is typed into and held by the terminal and when the Enter key is pressed, Service Request is indicated to the IOC and thus to the ACU. When the ACU next encounters the Branch on Service Request instruction in the program, the Service Request is recognised, the ACU passes control to the Read instruction and the transfer is handled by Interrupt.

1.3.3 Interrupt

Interrupt is a hardware function that enables characters to be transferred between the ACU and the FAC and the IOCs.

When a Read or Write instruction is executed and therefore a character transfer is required, the IOC and the ACU first establish the housekeeping of the transfer (such as how many characters are to be transferred, from where and to where) and then the transfer is effected character by character by the ACU during slack periods of its internal activity.

The partition requesting the transfer is not allocated processing time again until all the required characters have been transferred. Having initiated the transfer of one character for an IOC requesting such service, the ACU then initiates the transfer of one character for every other IOC requesting interrupt transfer before continuing with processing.

Because character transfers to or from disc or magnetic tape are faster than with other peripherals, the ACU is able to give more priority to such transfers. The interrupt function therefore takes place at one of three levels depending on the peripheral concerned.

1 DISC The ACU is dedicated to the disc until the transfer is complete

2 MAGNETIC TAPE During one interrupt cycle, interrupt after interrupt is generated until the transfer is complete

3 OTHER During one interrupt cycle, one interrupt is satisfied and one character transferred for each requesting IOC

1.3.4 Power failure

The line voltage of the ACU is constantly monitored. When a power failure is sensed, execution of the current program is suspended and status information is saved. When the power returns to normal, processing continues with the same partition, without operator intervention. The contents of core store remain unaltered by a power failure.

1.3.5 Program check

Certain serious errors during the execution of a program can be detected if they occur, and will cause a *program check.* That is, execution of the program in that partition ceases, the user is informed by, for example, the LOAD and LOCAL lights being illuminated on his VDU, and useful information about the program check is stored to assist in later determining the cause. Processing continues with the next partition.

When a program check occurs, the system stores in the error register (locations 41 to 44) of the relevant partition either the address plus ten of the instruction that caused the error, or, in the case of an I/O instruction, the address plus one of the instruction.

The errors which will cause a program check are:

1 An attempt being made to address a location outside the limit of Common or the partition

2 An attempt being made to write to the protected area of Common (the A, B and P registers)

3 Bit 5 of the fifth character fetched as part of an instruction is zero

4 The Binary-Coded Decimal value of the address bits of a character fetched as part of an instruction, an index register, or a disc address exceeds nine

1.3.6. Address check

An Address check occurs because the address specified in an instruction lies outside the allotted partition or Common size. As well as occurring at the initial specification of the instruction, an address check can occur as the instruction is executed. For example, if the instruction is required to access a number of consecutive characters, although the address of the first may be in range, as the instruction steps through, the address generated to access a later character may be out of range.

The instruction specifies to the ACU how many characters are to be accessed, and if for any reason too few can be accessed, an address check will occur.

1.3.7 Load Request

Load Request is the procedure that the ACU follows in order to begin processing in a partition either initially or after a program failure.

When a Load Request is encountered, a hardware Read Control instruction accepts ten characters from device zero, and executes the instruction they form. Typically, this instruction would be ten zeros which causes a Read (op code 0000) of sector zero from disc drive zero into location zero partition. Execution then continues with the next instruction (at location ten) which now contains some software instruction, and processing begins.

The same effect would be achieved if no instruction were entered: if less than ten characters are entered on device zero, remaining characters are zero-filled.

A Load Request occurs if the ACU encounters a Program Check or Address Check, or if a program failure is forced by, for example, the Load and Local keys at a terminal being pressed.

Store organisation and use

Chapter 2

System Ten core store provides direct access storage for a maximum of 160,000 (160K) locations. Store size can be installed in modules of 20K from the minimum of 20K. Each location is addressable and contains one six-bit character.

When the machine is installed, the available store is divided among Common and the desired number of partitions.

2.1 Common

Common may be a maximum of 80K locations in size and may be incremented in steps of 1K from the minimum of 1K. Memory, both Common and partition, is further subdivided into pages of 10K each. If the allocated memory is not a whole number of pages, the part page must have the highest page number. Locations in Common are addressed from 0000. Locations in Common may be used by any partition, except locations 0 to 299 which can be read only.

Locations 0 to 299 are divided into three areas used to store system information when partition switching occurs. Core storage is permanent, and it is this feature and the storage of this system information that, after a power failure allows processing to continue without further interruption.

The three areas are designated the P, B and A registers, and each is further subdivided to hold the information for each partition. The registers are allocated as follows:

Name	*Location*	*Use*
P register	000 to 099	To store the next program address to be accessed when the partition is next serviced
B register	100 to 199	To store the count of characters still to be transferred during interrupts
A register	200 to 299	To store information about the FAC peripherals available to the partition, and the data being used by the current instruction

2.2 Partition memory

Partition memory may be a maximum of 80K locations in size and may be incremented in steps of 1K from the minimum of zero, to 10K and then in steps of 10K to 80K. Memory, both Common and partition is further subdivided logically into pages of 10K each. If the allocated memory is not a whole number of pages, the part page must have the highest page number. Locations in a partition are addressed from 0000 (see section 1.3.1), and may only be accessed by a program in that partition, or by a program residing in Common but activated by a program in the currently executing partition.

Certain locations near the beginning of each partition have special uses, and are allocated as follows:

Locations	*Use*
0011 – 0014	Index register one
0021 – 0024	Index register two
0031 – 0034	Index register three
0041 – 0044	Error register

Each index register is numbered respectively by the tens digit of its address and can be used by the program for indexing, see section 2.3.1. The error register is used by the system to store the value of the P register if an error occurs. Location 40 holds a character that represents the size of the partition, and therefore does not change. The registers are not protected in any way, and can be used in the same way as any other partition locations.

2.3 Store addressing

There is no difference between characters stored as data or instructions except in the way in which each is interpreted. A data character is meaningful to the user and an instruction character is part of a ten-character instruction that is meaningful to the ACU. However, an instruction can be moved or overwritten in exactly the same way as data and it is this feature that allows a program to modify itself.

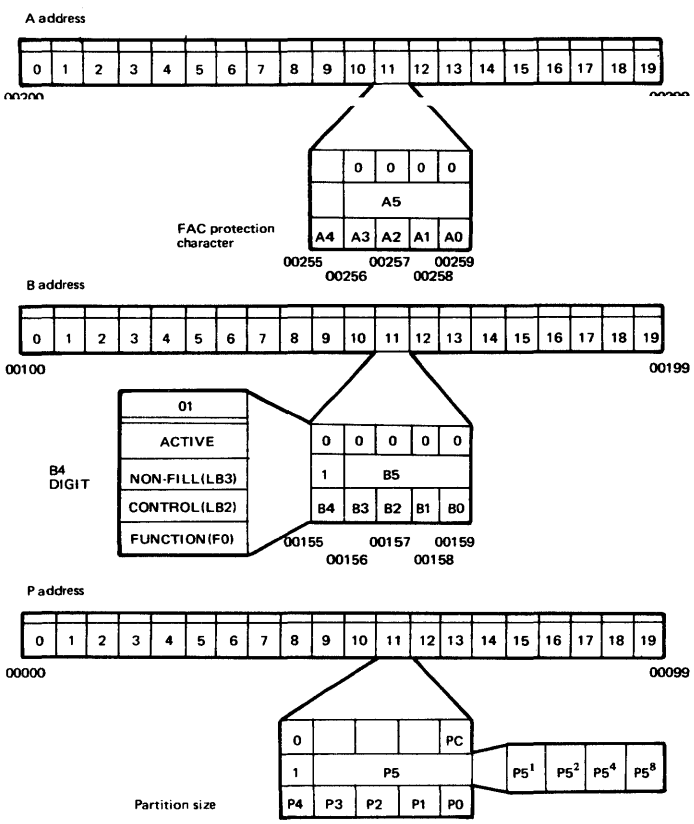


Figure 2.1 How systems information for partition 11 is held in the A, B and P registers in Common

Before the ACU can execute an instruction or operate on data, the address in store of the instruction and/or of the data must be specified. All store addressing is by character position, the six bits of the character being addressed simultaneously and being considered as one store location. A data field or an instruction is addressed by the position of its leftmost character and can be located anywhere in the store available to a partition user, that is, in partition or Common. An instruction (simply a ten-character field) must be located at an address divisible-by-ten.

Although at its lowest level the hardware can address the full 160K store available to the Model 22 processor, there is no need for a partition user to be able to address any location in store, merely any location within the store available to him: 80K Common and 80K partition at the most. Common and each partition is further logically subdivided into pages of 10K. Addresses in Common and each partition begin at 00000. It is convention that the address of a location in Common will be specified by five digits followed by the letter C; for example 29480C.

Thus any address available to a partition user (from 00000 to 79999 in Common or partition) can be indicated by:

1 One bit which specifies that the address is in Common or partition

2 Three page identification bits specify the relevant page (0 to 7)

3 The four numeric characters of the address specify the location (0000 to 9999, see below)

An address may be further modified (so long as it still falls within the store available to a partition user) by adding to the address the contents of another location in store. This is called *indexing* and is discussed fully in section 2.3.1.

In an instruction, an address is held in the four numeric bits of four characters. If these bits were interpreted in the usual (binary) way, this bit arrangement could only hold an address up to 60 (four numeric characters each holding up to the value 15). However, the address is held as four binary-coded decimal (BCD) digits, each capable of holding the digits 0 to 9, enabling addresses up to 10,000 (0 to 9999) to be accessed.

2.3.1 Indexing

Indexing is a convenient programming aid by which an address on which an instruction operates can be modified at execution time. A typical application for indexing is table access.

When the ACU finds that indexing has been specified for an address, the contents of the relevant index register added to the address specified by the instruction to form and effective address. The effective address is then used in the operation.

Each partition has three four-character registers (fields) held in locations 11 to 14, 21 to 24 and 31 to 34 named respectively Index Register one, two and three.

Indexing is specified in an instruction (see section 3.1) by the Indexing specifier (see section 3.1.3) for the A and/or B address being non-zero. If indexing has been specified, the numeric bits (interpreted as Binary-Coded Decimal (BCD)) of the relevant index register are added to the address held in the instruction to form the effective address.

An example of the use of indexing is as follows: the A address in an instruction contains the address 2000 and is indexed by index register two, which contains 0050. When the instruction is executed, the effective address of the A operand is 2050.

2.3.1.1 *Manipulating indexes*

The index registers are located in partition memory and their contents are therefore the responsibility of the user. The index registers may be used in two ways and it is important to differentiate between them:

1 The index may modify the address of a field to be operated on

2 The index may be the operand in any instruction (except Branch) and may therefore be operated on itself. It is in this way that its contents may be changed

2.3.2 Indirect addressing

Indirect addressing is a facility by which the address in an instruction does not point directly to the field to be operated on, but points to a field to be used as the address of the field to be operated on.

Indirect addressing may be specified for the A address and/or the B address of an instruction. An indirect address may refer to a location in partition or Common. That location is considered to be the first of a four-character address which may also refer to a location in partition or Common. For the purposes of indirect indexing, only some bits of the four characters are significant: the numeric bits (1 to 4) and the page bits (bit 5) of the first three characters, and the numeric bits and the address mode bit (bit 7) of the fourth character.

If both indexing and indirect addressing are specified for an address, the indirect address is calculated first and then the contents of the index register are added.

2.3.3 Extended indexing

Extended indexing is a facility that permits, with indexing, an effective address greater than the page specified by the instruction address. That is, the page bits of the instruction and the index register are included in the calculation of the effective address.

Extended indexing is specified by bit 5 of the tenth character of an instruction being set to 0.

2.4 Information storage

The smallest addressable unit of information in the System Ten computer is the character, and each character is represented by a pattern of six bits. Information is therefore stored and manipulated character by character. A number of contiguous characters is called a *field* and is addressed by the address of its leftmost character. Thus, a three-character field occupying locations 0641, 0642 and 0643 is addressed by 0641. Information in a field may be interpreted as data or an instruction

An instruction consists of a field of ten characters, but, an important distinction, it is interpreted bit by bit without regard to character. Each instruction must begin at an address divisible by ten. Instructions are described in Chapter 3.

A data field may be from 1 to 100 characters and may contain one of three types of data:

1 Numeric: any of the numbers 0 to 9

2 Alphabetic: any of the letters A to Z

3 Alphanumeric: any of the characters from the System Ten set

2.4.1 Character set

The bit patterns which represent the characters conform to the requirements of the American National Standards Institute (ANSI) and the System Ten computer character set is therefore a subset of the American Standard Code for Information Interchange (ASCII).

Row	b4	b3	b2	b1	Column 0	1	2	3	4	5	6	7
	NUMERIC BITS				0	0	0	0	1	1	1	1 (b7)
					0	0	1	1	0	0	1	1 (b6 ZONE BITS)
					0	1	0	1	0	1	0	1 (b5)
0	0	0	0	0	NUL	DLE	space	0	@	P	`	p
1	0	0	0	1	SOH	DC1	!	1	A	Q	a	q
2	0	0	1	0	STX	DC2	"	2	B	R	b	r
3	0	0	1	1	ETX	DC3	#	3	C	S	c	s
4	0	1	0	0	EOT	DC4	$	4	D	T	d	t
5	0	1	0	1	ENQ	NAK	%	5	E	U	e	u
6	0	1	1	0	ACK	SYN	&	6	F	V	f	v
7	0	1	1	1	BEL	ETB	'	7	G	W	g	w
8	1	0	0	0	BS	CAN	(	8	H	X	h	x
9	1	0	0	1	HT	EM	)	9	I	Y	i	y
10	1	0	1	0	LF	SUB	*	:	J	Z	j	z
11	1	0	1	1	VT	ESC	+	;	K	[	k	{
12	1	1	0	0	FF	FS	,	<	L	\	l	¦
13	1	1	0	1	CR	GS	-	=	M	]	m	}
14	1	1	1	0	SO	RS	.	>	N	^	n	~
15	1	1	1	1	SI	US	/	?	O	_	o	DEL

Figure 2.2 The ASCII character set with the System Ten character set outlined boldly

By convention the seven bits of an ASCII character are numbered from the right from b1 to b7. Figure 2.2 shows the ASCII character set and bit patterns, with the System Ten character set outlined boldly.

The full ASCII character set requires seven bits to represent all 128 characters. The System Ten set has only 64 characters which is represented by six bits. The four least significant bits are called the numeric bits and the two most significant bits are called the zone bits. The System Ten bit pattern for a character may therefore be derived by omitting bit 6 from the ASCII bit pattern for the character. For example:

Character	*ASCII*	*System Ten*	
	7654321	754321	bit number
	0101100	001100	
D	1000100	100100	

It is a convention that the bits of a System Ten character are numbered 1, 2, 3, 4, 5 and 7.

It is often required to convert a System Ten character back to its ASCII representation; during output, for example. This can be achieved by inserting a bit 6 into the System Ten bit pattern that is the inverse of bit 7. For example:

Character	*System Ten*	*ASCII*	
	754321	7654321	bit number
	 001100	0101100	
D	 100100	1000100	

A different conversion may take place when the special instruction *Write Control* is executed, or during FAC I/O operations. Write control mode is described in section 3.2.15.2, and the control characters are described in section 4.1.2.

2.4.2 Collating sequence

The collating sequence of the System Ten character set determines the relative values of the characters for the purposes of making comparisons and sorting. In the System Ten character set, the space character Δ has the lowest value and the underline character _ has the highest value. Thus, the digits are lower than the letters, the letter B has a greater value than the letter A and ? has a greater value than /.

The collating sequence is as follows:

Lowest *Highest*

Δ!"#$%&'()*+,-./ 0 to 9:;<=> ?@A to Z[\] ^_

2.4.3 Alternative characters in the set

The following alternatives to the standard characters are permitted to allow for national variations:

Standard	*Alternative*
#	£
@	± or ◊
[	Å or Ä or AE
]	Å or Ü
^	° or ‰

2.4.4 Representation of negative numbers

If a numeric field has a sign, the sign is stored in bit 7 of the rightmost (least significant) digit. If bit 7 is one, the field is negative; if bit 7 is zero, the field is positive. Notice that, from Figure 2.2, this causes a negative digit to have a bit configuration identical to an alphabetic character in the range P to Y. The System Ten instruction set recognises the value of numbers that follow this convention and operates algebraically on signed numeric data. Thus, adding a minus one to an eight yields a result of seven.

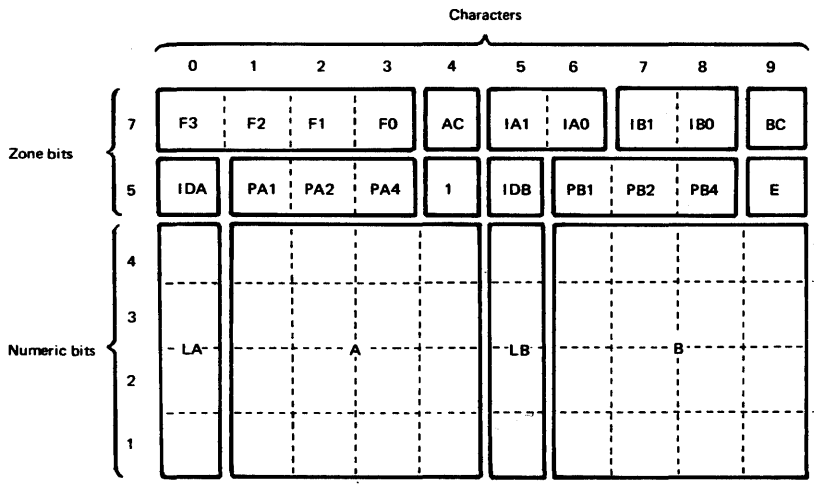
3.1 Format of the instruction

Each System Ten instruction is ten characters in length and must begin so that its leftmost character is situated in a location whose address is evenly divisible by ten.

The first few characters of an instruction as they appear in memory have the following format:

Character	0						1						2					
Bit use	F3	IDA	LA	LA	LA	LA	F2	PA1	A	A	A	A	F1	PA2	A	A	A	A
Bit	7	5	4	3	2	1	7	5	4	3	2	1	7	5	4	3	2	1

but since the ten characters of an instruction are interpreted bit by bit without regard to character, a more useful representation is achieved by giving a vertical orientation to the six bits of each character:



Each of the bit fields shown in the above illustration is used by the ACU in interpreting and executing instructions.

Field	*Size in bits*	*Interpretation and use*
F	4	Operation code of the instruction
AC	1	A address common/partition indicator
IA	2	A address indexing field
IB	2	B address indexing field
BC	1	B address common/partition indicator
IDA	1	Indirect addressing indicator for A address
PA	3	Page indicator for A address
	1	Reserved, must be set to 1
IDB	1	Indirect addressing indicator for B address
PB	3	Page indicator for B address
E	1	Extended indexing bit
LA	4	Length specifier for A address
A	16	A address within page
LB	4	Length specifier for B address
B	16	B address within page

The above fields are explained in the sections that follow.

3.1.1 Operation code field

The operation code is stored in bit 7 of the first four characters of the instruction. The code is interpreted in binary form and indicates which of the 16 (0 to 15) available instructions the ACU is to execute.

The instructions and their binary representations are as follows:

Binary code	*Instruction*
0000	Read
0001	Write
0010	Add Address
0011	Move Address
0100	Add
0101	Divide
0110	Multiply
0111	Subtract
1000	Move Character
1001	Move Numeric
1010	*Reserved*
1011	Branch
1100	Edit
1101	Form Numeric
1110	Compare
1111	Exchange

3.1.2 Address mode

The address mode bits (one each for the A address and the B address) specify, if set to 1, that the appropriate address is in Common or, if set to 0, that it is in partition memory.

3.1.3 Indexing field

Indexing is explained in section 2.3.1 and 2.3.3.

The indexing field is interpreted as follows:

Bit content	*Meaning*
00	No indexing
01	Indexing using index register one
10	Indexing using index register two
11	Indexing using index register three

3.1.4 Indirect addressing indicator

Indirect addressing is explained in section 2.3.1.1. If the indirect addressing indicator bit is set to one, the ACU accesses four characters from store starting at the relevant address specified in the instruction and uses those characters to form the effective address.

3.1.5 Page indicator

The three page indicator bits of an address extend the addressable memory by specifying in which of the eight (0 to 7) pages the address is located. A page contains 10K characters and is therefore written as the most significant characters of a store location. For example, location 5048 in page 2 of Common is usually written 25048C.

The page indicator bits of an address are not held in the conventional binary form, in that they are written from left to right (least significant to the left) and then complemented. For example, to decode a page number held as 001:

1 Complement (invert) its binary representation, giving 110

2 Decode the complement from left to right:

Decimal bit value	1	2	4
Complement	1	1	0

giving 1 plus 2 plus 0 equals 3

So the page number specified is 3.

3.1.6 Address length specifier

The lengths of the operands on which an instruction will act are held in the instruction, one for the A address, one for the B address. However, certain instructions use the length specifiers differently since the instructions may have one of two formats: the one-length format or the two-length format.

In both formats the length specifiers hold the operand length in the same way; it is in the way the instruction interprets this value that they differ.

The length specifier holds the number of characters to be operated upon by the instruction as a binary coded decimal (BCD) value, that is, a value from 0 to 9 (where 0 represents ten) held in binary. The four bits available can, of course, hold up to the value 16 (0 to 15) but values over 9 are illegal. It is therefore not possible to specify a length of 0.

The two-length format is typical of arithmetic instructions, and each address length specifier gives the number of characters addressed. For example, a three-digit operand A is to be added to a four-digit operand B. The A length specifier will hold 3 and the B length specifier will hold 4. Notice that an operand of ten characters is the longest that can be specified in a two-length format instruction.

The one-length format is typical of instructions used to manipulate alphanumeric data, and can specify fields longer than ten characters. In a one-length format instruction, the address length specifiers are considered as a two-digit value. Thus if the A length contains 5 and the B length contains 6, the instruction manipulates 56 characters of data. If both length specifiers are zero, the instruction manipulates 100 characters of data.

Certain instructions use the length specifiers for other purposes. See the relevant instruction for further details.

3.1.7 The address field

Each instruction contains two address fields, A and B. In most instructions they are used to specify the location of the leftmost character of the operands to be used by the instruction. Each address is a four-digit binary coded decimal number from 0000 to 9999. If the numeric bits of any character in the address indicate decimal numbers greater than 9, the ACU will cause Program Check.

The address fields may be modified by other fields in the instruction to indicate whether an address is in Common or partition, in which page it lies, and whether or not indexing is to be applied.

Note that it is possible to specify an A address and a B address that point to the same field, or to fields which overlap.

3.2 Instruction descriptions

This section describes the ACU instructions, in alphabetical order by instruction name.

The interpretation and naming of the bit fields of an instruction is given in section 3.1 and these descriptions should be read in conjunction with that section.

3.2.1 Add

The Add instruction adds the numeric bits of the field indicated by the A address to the numeric bits of the field indicated by the B address, according to the rules of algebra. The result overwrites the contents of the B field. The A field remains unchanged so long as the fields do not overlap.

3.2.1.1 *Instruction fields*

Machine operation code

F Binary 0100 (4).

Address specification

A Address of the leftmost position of Operand-A.

B Address of the leftmost position of Operand-B.

Indexing specification

IA Index register for determining effective address of Operand-A.

IB Index register for determining effective address of Operand-B.

Common partition specification

AC If AC is 0, A is address in partition.

If AC is 1, A is address in Common.

BC If BC is 0, B is address in partition.

If BC is 1, B is address in Common.

Length specification

LA Length of Operand-A.

LB Length of Operand-B.

3.2.1.2 *Operand fields*

Operand-A address

If IA is 0, then A is the effective address.

If IA is 1, 2, or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address.

If IB is 1, 2, or 3, the corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

If LA is 0, the length of Operand-A is 10 characters.
If LA is 1 to 9, the length of Operand-A is 1 to 9 characters.

If LB is 0, the length of Operand-B is 10 characters.
If LB is 1 to 9, the length of Operand-B is 1 to 9 characters.

3.2.1.3 *Operation*

General description

The add operation proceeds from right to left starting with the rightmost characters of Operand-A and Operand-B. Character by character, the algebraic sum is developed in Operand-B.

If Operand-A is shorter than Operand-B, the operation proceeds normally until Operand-A is exhausted. After that, the process continues in similar fashion except that a zero character is automatically substituted every time the adding logic calls for a character from Operand-A. In effect, Operand-A is given enough preceding zeros to make it the same length as Operand-B.

If Operand-A is longer than Operand-B, addition stops after the leftmost position in Operand-B has been added. The remaining positions in Operand-A are ignored, and do not affect the sum or the condition code.

The algebraic sign of the sum is placed in bit-7 of the rightmost position of Operand-B, and bit-5 is turned ON. Except for the rightmost character, the other zone bits of Operand-B are unchanged. Operand-A is unchanged by the add operation.

If the sum exceeds the capacity of Operand-B, a carry-to-the-left from the leftmost position does not occur. Condition code 4 is set to indicate the overflow.

Condition codes

After completion of the Add instruction:

1 = Negative, non-zero sum.
2 = Zero sum.
3 = Positive, non-zero sum.
4 = Overflow.

3.2.1.4 *Programming hints*

Overlapped operand

In case of overlapped operands, the result is unspecified.

3.2.2 Add Address

The Add Address instruction modifies an address, or an address-like field, in memory by adding to it another address, or the contents of the field indicated by another address. The Add Address instruction only affects 20 bits of the four characters: 16 address bits, three page bits and the Common/partition bit. The other four bits remain unaltered after execution.

The effective address (that is, taking account of indexing and indirect addressing) of both fields is calculated and a four-character address or address-like field is fetched from each. The four characters are expanded to five to facilitate calculation. The two addresses are added together, compressed into four-character format and placed in the field indicated by the B address, overwriting the original contents.

The user may specify that the A address itself, not the contents of the field indicated by the A address, are to be added to the B address.

The Common/partition bit in both addresses is tested to determine whether the final address is in Common or partition. If either field indicates an address in Common, the resulting address will be in Common. If both fields indicate a partition address, the resulting address is in partition. When indirect addressing is specified, the Common/partition bit in the indirect address (rather than in the instruction address) determines whether the final address is in Common or partition.

The Add Address instruction will not generate an address greater than 79999. Any attempt to generate a higher address causes a memory wraparound and condition code 4 is set. A memory wraparound means that 80K is subtracted from the generated address to bring it back in range.

3.2.2.1 *Instruction fields*

Machine operation code

F — Binary 0010 (2)

Address specification

A — If LA = 0, the A address itself is to be added to the B address
If LA = 1, the A address points to the leftmost character of the data field to be added to the B address

B — The address of the leftmost character of the field to be added to, and in which the result will be formed

Indexing specification

IA, IB — Index registers for determining effective address of A and B operands

Indirect addressing specification

IDA, IDB Indicate whether or not the A and B addresses are indirect

Extended indexing specification

E If 0, extended indexing is performed

Common/partition specification

AC, BC If 0, the respective address is in partition
If 1, the respective address is in common

Length specification

LA If 0, add to the B address the A address itself
If 1, add to the B address the data indicated by the A address

LA values of 2 to 9 cause Program Check

LB Ignored

Condition codes

The Add Address instruction sets the following condition codes:

Code	*Meaning*
1	Result address is in Common
2	Not used
3	Result address is in partition
4	Result address exceeds 79999

3.2.3 Branch

The Branch instruction permits departure from the sequential path by which instructions are normally executed. Branching can be unconditional, it can depend upon the current status of the condition code, or it can depend upon signals from Input/Output devices requesting service from the CPU. A variant of the Branch instruction passes control to a subroutine after first setting the return address at which the main program will be resumed. Execution of the Branch instruction does not alter the condition code.

3.2.3.1 *Instruction fields*

Machine operation code

F Binary 1011 (11).

Address specification

A Address-A

B Address-B

Indexing specification

IA Ignored. Branch instructions are not indexed.

IB Ignored. Branch instructions are not indexed.

Common/partition specification

AC If AC is 0, A is an address in partition.
If AC is 1, A is an address in Common.

BC If BC is 0, B is an address in partition.
If BC is 1, B is an address in Common.

Variant specification

LA A digit 0–9

LB A digit 0–6, 8, 9

3.2.3.2 *Operation*

Order of presentation

The Branch instruction consists of several variants. The LA and LB instruction fields determine which variant is executed. "Link" (variant 6) and "Branch on Service Request" (variant 7) require that the entire instruction be decoded. These variants are discussed later under separate headings. The other variants are decoded and executed a half instruction at a time and are most conveniently discussed as a group in the next paragraph.

Variants 0, 1, 2, 3, 4, 5, 8, 9

The first five characters of the instruction are fetched. LA is examined. If a branch is required, control passes to Address-A, and the right half of the instruction is ignored. If a branch is not required in the left half of the instruction, the right half is fetched. LB is examined. If a branch is required, control passes to Address-B. If a branch is not required, execution continues with the next sequential instruction.

The following table shows the values which LA and LB may assume. Beside each variant number is the meaning applied by the ACU. Variant 6 and Variant 7 are purposely omitted. They are discussed under "Link" and "Branch on Service Request".

Variant	*Operation*
0	Do not branch ("no operation").
1	Branch if condition code is 1.
2	Branch if condition code is 2.
3	Branch if condition code is 3.
4	Branch if condition code is 4.
5	Branch, unconditionally.
8	Branch and switch partitions, unconditionally.
9	Do not branch ("no operation").

Partition switching

If a Branch instruction does not require a branch, execution simply continues with the next sequential instruction.

If the host partition has been in continuous control for more than the nominal 40 milliseconds when a branch is required, the branch is taken but the execution of the instruction at the branch address is postponed and control passes to the next partition. When control returns, execution resumes at the branch address. If the branch is caused by variant 8 ("Branch and switch, unconditionally"), the branch is taken but the execution of the instruction at the branch address is postponed and control passes to the next partition even though 40 milliseconds have not elapsed.

Link–Branch Variant 6

LA Must be 6.

LE May be 0 to 5, 8, or 9.

If LB is 0 or 9, no link occurs, control simply passes to the next instruction.

If LB is 1–4, the corresponding condition code is tested. If the specified condition code is ON, the link operation is performed. Otherwise, control simply passes to the next instruction.

If LB is 5 or 8, the link operation is performed, unconditionally.

Return address/start address

The address of the next instruction (return address) is inserted into the numerical portion of the four position field starting at Address-A. The zone portions of the three left character positions are unchanged. Bit-5 of the rightmost position is set to 1. Bit-7 is set to 1 if the return address is in common; it is set to 0 if the return address is in partition. Control then passes to Address-B (start address).

Branch on Service Request–Branch Variant 7

LA Must be 7.

LB Must be 0 or 9.

Operation–storing device number

Each IOC continually polls the input/output devices attached to it to see if a device has signalled a request for service. If the IOC encounters such a signal, further polling for service requests is temporarily discontinued, and the device number is held in a counter until the ACU executes "Branch on Service Request". "Branch on Service Request" causes the counter to be stored in the numeric portion of the character position pointed to by Address-A. Control then passes to Address-B. Polling resumes with the next higher device number (or 0, if the requesting device was 9).

If the IOC is holding no such request for service, "Branch on Service Request" has no effect. Execution continues with the next sequential instruction.

Condition codes

Condition codes are unchanged by the Branch instruction.

3.2.3.3 *Programming hints*

Since each instruction (with the exception of Branch) sets the condition code, it is necessary to test the condition code immediately after the performance of an operation.

3.2.4 Compare

The Compare instruction compares two fields and sets the condition code to indicate the relation between them.

3.2.4.1 *Instruction fields*

Machine operation code

F Binary 1110 (14)

Address specification

A Address of the leftmost position of Operand-A.

B Address of the leftmost position of Operand-B.

Indexing specification

IA Index register for determining effective address of Operand-A.

IB Index register for determining effective address of Operand-B.

Common/partition specification

AC If AC is 0, A is address in partition.
If AC is 1, A is address in Common.

BC If BC is 0, B is address in partition.
If BC is 1, B is address in Common.

Length specification

LA Tens position of length of both Operand-A and Operand-B.

LB Units position of length of both Operand-A and Operand-B.

3.2.4.2 *Operand fields*

Operand-A address

If IA is 0, then A is the effective address. If IA is 1, 2, or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address. If IB is 1, 2, or 3, then corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

Operand-A and Operand-B are equal in length. 10LA + LB = Lengths of operands for the Compare instruction.

If 10LA + LB = 00, 100 is the length of the operands.

3.2.4.3 *Operation*

General description

The Compare operation proceeds from left to right starting with the leftmost character of Operand-A and Operand-B. Character by character, the values of Operand-A and Operand-B are compared until a difference is found *or* the rightmost position has been compared.

When the characters differ, Condition Code 1, *or* 3 and 4 is set ON (indicating that Operand-A is smaller or larger than Operand-B) and the operation is complete.

If the characters are identical, and there are more positions to be compared, the comparison is repeated for the next position on the right.

When the characters are identical and there are no more positions to be compared, condition codes 2 and 4 are set ON.

Operand-A and Operand-B are unchanged by the compare operation.

When condition code 3 or 2 is set ON, condition code 4 is also set ON.

Condition codes

1, if Operand-A is less than Operand-B.

2 and 4, if Operand-A and Operand-B are identical.

3 and 4, if Operand-A is greater than Operand-B.

3.2.4.4 *Programming hints*

Character values

To determine which of two characters is considered to be the greater, see section 2.4.2.

Sorting

A principal use of the Compare instruction is in sorting data. The programmer is reminded that the units position of a negative numeric field is coded with zone bit-7 *ON*. (If the digit were positive, bit-7 would be OFF). Thus, in a compare operation, a negative digit is of greater value than *any* positive digit.

3.2.5 Divide

The Divide instruction computes the algebraic quotient (and remainder) of two operands.

3.2.5.1 *Instruction fields*

Machine operation code

F Binary 0101 (5)

Address specification

A Address of the leftmost position of Operand-A.

B Address of the leftmost position of Operand-B (dividend).
Address of the quotient.

Indexing specification

IA Index register for determining effective address of Operand-A.

IB Index register for determining effective address of Operand-B.

Common/partition specification

AC If AC is 0, A is an address in partition.
If AC is 1, A is an address in Common.

BC If BC is 0, B is an address in partition.
If BC is 1, B is an address in Common.

Length specification

LA Length of Operand-A (divisor).

LB Length of the quotient.

LA + LB Length of Operand-B (dividend).

3.2.5.2 *Operand fields*

Operand-A address

If IA is 0, then A is the effective address.
If IA is 1, 2, or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address.
If IB is 1, 2, or 3, the corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

If LA is 0, the length of Operand-A is 10 characters.
If LA is 1 thru 9, the length of Operand-A is 1 thru 9 characters.

If LB is 0, the length of quotient is 10 characters.
If LB is 1 thru 9, the length of quotient is 1 thru 9 characters.

LA + LB is the length of Operand-B (dividend).

3.2.5.3 *Operation*

General description

Operand-A is the divisor.

The dividend begins at the B address and contains LB + LA positions.

At the end of the operation, the quotient occupies the leftmost LB positions of the dividend field, and the remainder occupies the rightmost LA positions of the dividend field.

If the divisor and the dividend differ in sign, bit-7 of the quotient is turned ON to indicate a negative quotient. If the signs are alike, bit-7 is turned OFF to indicate a positive quotient.

Bit-5 is turned ON for all positions of the quotient: bit-7 is turned OFF for all positions except the rightmost.

Bit-7 of the rightmost position of the remainder is unchanged. It continues to show the sign of the dividend. Bit-5 is set to 1. The zone bits of the other positions in the remainder are unchanged.

Process

An internal counter is set to zero. It will count the number of times the divisor is subtracted from a subfield-of-the-dividend. The subfield length is one greater than the length of the divisor. The first subfield chosen is at the extreme left of the dividend.

The divisor is repeatedly subtracted from the subfield until the value of the subfield is less than that of the divisor. Each subtraction increments the counter. If the count exceeds 9, condition code 4 is set (indicating overflow), and the operation is abandoned. If the count does not exceed 9, the subfield value is less than the divisor, the count is stored in the leftmost position of the quotient. The counter is cleared, and the process shifts to the next subfield (one character position to the right in the dividend) to develop the second position of the quotient. After this, another shift to develop the third position, etc. The operation ends after the rightmost subfield in the dividend is processed in this fashion.

Condition codes

After completion of the Divide instruction:

1 = Negative, non-zero quotient.
2 = Zero quotient.
3 = Positive, non-zero quotient.
4 = Overflow.

3.2.5.4 *Programming hints*

Overlapped operands

In case of overlapped operands, the result is unspecified.

Division by zero

An attempt to divide by zero causes condition code 4 to be set (indicating overflow). The value of the dividend is unchanged.

Preventing overflow

Overflow will only occur if the absolute value in the leftmost LA positions of the dividend equals or exceeds the absolute value of the divisor. In cases where it is necessary to accommodate the widest possible range of data, including division by 1, the leftmost LA positions of the dividend should contain zero.

3.2.6 Edit

The Edit instruction moves a 1-100 digit numerical field into a control field so that the information is in a form suitable for printing. The control field governs the suppression of preceding zeros (including the insertion of cheque protection characters ahead of significant digits), the insertion of punctuation marks, and the indication of sign.

3.2.6.1 *Instruction fields*

Machine operation code

F Binary 1100 (12).

Address specification

A Address of the leftmost position of Operand-A.

B Address of the leftmost position of Operand-B.

Indexing specification

IA Index register for determining effective address of Operand-A.

IB Index register for determining effective address of Operand-B.

Common partition specification

AC — If AC is 0, A is an address in controlling partition.
If AC is 1, A is an address in Common.

BC — If BC is 0, B is an address in controlling partition.
If BC is 1, B is an address in Common.

Length specification

LA — Tens position of length of Operand-A.

LB — Units position of length of Operand-A.

3.2.6.2 *Operand fields*

Operand-A address

If IA is 0, then A is the effective address.
If IA is 1, 2, or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address.
If IB is 1, 2, or 3, the corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

The length of Operand-A is 10LA+LB.

If 10LA+LB=00, the length = 100.

The length of Operand-B is the sum of the following:

Operand-A length + 1

The number of punctuation characters in Operand-B.

The number of @ characters in Operand-B.

3.2.6.3 *Operation*

Operand-B, the control field

A filler character is defined as any character other than the @ sign or a punctuation mark (comma, decimal point, hyphen, slash).

Minimally, a control field consists of as many filler characters as there are digits in Operand-A plus one trailing character to show sign. In addition, the filler characters may be freely interspersed with punctuation characters (comma, period, hyphen, slash) and @ signs.

Since the Edit instruction destroys the control field, the programmer normally moves the control field to the Operand-B address before each use of the Edit instruction.

The filler characters designate the mask positions into which Operand-A digits can be moved. Significant digits from Operand-A simply replace the corresponding filler positions in the control field. Filler characters corresponding to non-significant zeros in Operand-A are not replaced, they are undisturbed. This permits the suppression of preceding zeros (that is, the filler positions are preset to contain blank characters) or the use of cheque protection characters ahead of significant digits (that is the filler positions are preset to contain a protect character, such as asterisk).

The punctuation characters are used to punctuate the significant information received from Operand-A. At the completion of the Edit instruction, any punctuation characters which find themselves embedded in the significant portion of the control field remain undisturbed by the operation and thus show the desired punctuation. Any punctuation character to the left of the significant portion of the control field will have been replaced by the neighbouring character on the left and thus wiped out. A control field should not begin with a punctuation character.

The @ sign is used to insert blank characters between filler positions. Execution of the Edit instruction replaces each @ sign in the mask with a blank character.

The rightmost position of the control field is used to show the sign of Operand-A.

Ordinarily, the programmer presets the position to contain a hyphen, or some other character to indicate minus. If Operand-A is negative, the minus character remains. If Operand-A is zero or positive, the minus character is overwritten with a blank character.

Execution of Edit instruction

The Edit instruction begins by extracting the leftmost digit of Operand-A and by finding the leftmost filler character in the control field. During the hunt for the filler character, any intervening @ sign in the control field is replaced by a blank character and any intervening punctuation mark is replaced by the neighbouring character on the left.

If the Operand-A digit is significant, the numeric portion is put into the filler position of the control field, and the zone bits of that position are set to 0/1 to ensure that the position will print as a numerical value.

If the digit is non-significant zero, but the filler character is 0, the digit is stored in the filler position as a significant zero (as are any to the right of it in Operand-A).

If the digit is non-significant, the filler character is left undisturbed.

The process is repeated using the next digit to the right in Operand-A and the next filler character in the control field. Once a significant digit has been moved from Operand-A into the control field, any punctuation mark to the right of it is allowed to stand and is not replaced by its left-hand neighbour.

The process continues until the rightmost digit in Operand-A and the rightmost filler character of the control field have been dealt with. The condition code is set. If Operand-A contains a positive value or zero, a blank character is set in the sign position of the control field (the position just to the right of the rightmost filler character).

Condition codes

After completion of the Edit instruction.

1 Negative, non-zero, Operand-A.

2 Zero Operand-A.

3 Positive, non-zero Operand-A.

An overflow condition is not possible.

3.2.6.4 *Examples*

Printing Social Security numbers

Operand-A	098144159
Operand-B	000-00-0000- before editing
Operand-B	098-14-4159 after editing

Cheque protection

Operand-A	0000001234
Operand-B	**,***,***.00- before editing
Operand-B	********12.34 after editing

Use of commas

Operand-A	1234567890
Operand-B	bb,bbb,bbb.00- before editing
Operand-B	12,345,678.90 after editing

Note that b is used here to represent a blank character.

Suppressing preceding zeros

Operand-A	0000012345
Operand-B	bb,bbb,bbb.00- before editing
Operand-B	bbbbbbb123.45 after editing

Note that b is used here to represent a blank character.

3.2.7 Exchange

The Exchange instruction interchanges the characters in two fields of equal length in main memory. Each field can comprise 1–100 characters.

3.2.7.1 *Instruction fields*

Machine operation code

F Binary 1111(15)

Address specification

A Address of the leftmost position of Operand-A.

E Address of the leftmost position of Operand-B.

Indexing specification

IA Index register for determining effective address of Operand-A.

IB Index register for determining effective address of Operand-B.

Common/partition specification

AC If AC is 0, A is an address in partition.
If AC is 1, A is an address in Common.

BC If BC is 0, B is an address in partition.
If BC is 1, B is an address in Common.

Length specification

LA Tens position of length of both Operand-A and Operand-B.

LB Units position of length of both Operand-A and Operand-B.

3.2.7.2 *Operand fields*

Operand-A address

If IA is 0, then A is the effective address.
If IA is 1, 2, or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address.
If IB is 1, 2, or 3, the corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

Operand-A and Operand-B are equal in length.

10LA + LB = Lengths of operands for Move Character instruction.

If 10LA + LB = 00, 100 is the length of the operands.

3.2.7.3 *Operation*

General description

The leftmost character of Operand-B is extracted and held temporarily in a register. The character in the leftmost position of Operand-A is moved to the leftmost position in Operand-B, and the character in the register is then stored in the leftmost position of Operand-A. This operation is repeated from left to right until the entire fields have been interchanged.

Condition code

2, after completion of the Exchange instruction.

3.2.7.4 *Programming hints*

If Operand-A and Operand-B do not overlap, a simple exchange occurs.

If Operand-A and Operand-B overlap each other, the programmer can predict the result for any particular case by mentally stepping through the operation as described in "General Description" above.

Note: Using an overlapped exchange instruction can be useful for rotating characters of a field. If Operand-A and Operand-B overlap for all but one character, then each time the exchange instruction is executed the leftmost character moves to the rightmost position, and all other characters move one position to the left.

3.2.8 Form Numeric

The Form Numeric instruction moves numeric information from a 1-10 position mixed field to a second 1-10 position field. After the operation, the second field is of the numerical form normally used for arithmetic operations.

3.2.8.1 *Instruction fields*

Machine operation code

F Binary 1101 (13).

Address specification

A Address of the leftmost position of Operand-A.

E Address of the leftmost position of Operand-B.

Indexing specification

IA Index register for determining effective address of Operand-A.

IB Index register for determining effective address of Operand-B.

Common/partition specification

AC If AC is 0, A is an address in partition.
If AC is 1, A is an address in Common.

BC If BC is 0, B is an address in partition.
If BD is 1, B is an address in Common.

Length specification

LA Length of Operand-A

LB Length of Operand-B

3.2.8.2 *Operand fields*

Operand-A address

If IA is 0, then A is the effective address.
If IA is 1, 2, or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address.
If IB is 1, 2 or 3, the corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

If LA is 0, the length of Operand-A is 10 characters.
If LA is 1 to 9, the length of Operand-A is 1 to 9 characters.

If LB is 0, the length of Operand-B is 10 characters.
If LB is 1 to 9, the length of Operand-B is 1 to 9 characters.

3.2.8.3 *OPERATION*

Execution of Form Numeric instruction

Execution begins with a right-to-left search for the rightmost digit in Operand-A and a determination of its sign:

1 If the rightmost non-blank character is a digit, it is moved unchanged into the rightmost position of Operand-B. The sign of Operand-B is positive

2 If the rightmost non-blank character is one of the characters P to Y, it is considered to be a digit with a minus sign. It is moved unchanged into the rightmost position of Operand-B. The sign of Operand-B is negative

3 If the rigntmost non-blank character is a hyphen (minus sign), the rightmost digit is converted to the corresponding character P to Y (that is, bit-7 is set ON) and is stored in the rightmost position of Operand-B. The sign of Operand-B is negative.

4 If the rightmost non-blank character is none of the above, it is skipped over and the rightmost digit is moved unchanged into the rightmost position of Operand-B. The sign of Operand-B is positive

Once the rightmost digit is selected from Operand-A and is moved into Operand-B, the process continues from right to left. The next digit to the left is found in Operand-A and is moved unchanged into the next left position of Operand-B. Intervening characters which are not digits are simply passed over and are now moved.

If a digit is moved into the leftmost position of Operand-B and there are yet unmoved digits in Operand-A, the operation is abandoned and condition code 4 is set to show the overflow condition.

When the leftmost digit of Operand-A is moved into an Operand-B position, any unfilled positions in Operand-B are set to zero and the operation is finished.

If Operand-A consists entirely of blank characters, no digits can be moved. In this case, Operand-B is set to zero in all positions.

Condition codes

After completion of the Form Numeric instruction.

1 = Negative, non-zero Operand-B.
2 = Zero Operand-B.
3 = Positive, non-zero Operand-B.
4 = Overflow

3.2.9 Move Address

The Move Address instruction transfers bits 1 to 5 of up to 100 characters from one memory location to another. All the bits of the rightmost character are transferred. Although the instruction can transfer 100 characters, four is the usual number transferred.

3.2.9.1 *Instruction fields*

Machine operation code

F Binary 0011 (3)

Address specification

A Address of the leftmost character of the field to be moved

B Address of the leftmost character of the destination field

Indexing specification

IA, IB The A and B addresses may be indexed

Indirect addressing specification

IDA, IDB If 0, the respective A or B address is indirect

Common/partition specification

AC, BC If 0, the respective address is in partition
If 1, the respective address is in Common

Length specification

LA, LB LA is the tens digit and LB is the units digit of the number of characters to be moved. Each is interpreted as a Binary-Coded Decimal digit from 0 to 9 indicating length in characters from 01 to 99 with 00 indicating 100

Condition codes

The Move Address instruction sets the following condition codes:

Code	*Meaning*
1	Result address is in Common
2	Not used
3	Result address is in partition
4	Not used

3.2.10 Move Character

The Move Character instruction moves 1-100 characters from one location in main memory to another.

3.2.10.1 *Instruction fields*

Machine operation code

F Binary 1000 (8)

Address specification

A Address of the left most position of Operand-A.

B Address of the leftmost position of Operand-B.

Indexing specification

IA Index register for determining effective address of Operand-A

IB Index register for determining effective address of Operand-B.

Common/partition specification

AC If AC is 0, A is an address in partition.
If AC is 1, A is an address in Common.

BC If BC is 0, B is an address in partition.
If BC is 1, B is an address in Common.

Length specification

LA Tens position of length of both Operand-A and Operand-B.

LB Units position of length of both Operand-A and Operand-B.

3.2.10.2 *Operand fields*

Operand-A address

If IA is 0, then A is the effective address.
If IA is 1, 2, or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address.
If B is 1, 2, or 3, the corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

Operand-A and Operand-B are equal in length. 10LA + LB Length of operands for Move Character instruction.

If 10LA + LB 00, 100 is the length of the operands.

3.2.10.3 *Operation*

General description

Operand-A is copied into Operand-B, one position at a time, from left to right, starting with the leftmost position of Operand-A and writing it into the leftmost position of Operand-B.

Condition code

2, after completion of the Move Character instruction.

3.2.10.4 *Programming hints*

Move Character and Move Numeric

The Move Character instruction is similar to the Move Numeric instruction. The Move Numeric instruction will extract and copy only the numeric portion of a character (leaving the zone bits unchanged); the Move Character instruction will copy an entire character including both numeric and zone portions.

Overlapping operands

If Operand-A and Operand-B do *not* overlap, then Operand-A is unchanged by the Move Character instruction.

To shift the Operand-A data field one or more positions to the *left* (to a lower machine address) the Move Character instruction can be used when the operands overlap if the Operand-B address is not greater than the Operand-A address. Only the unlapped positions of Operand-A will be unchanged.

To propagate a given character throughout a data field, put the character into the leftmost position of the field, and use the Move Character instruction as follows:

Operand-A address is the address of the data field.

Operand-B address is the address of the data field + 1.

Operand length must be 1 less than the data field length.

3.2.11 Move Numeric

The Move Numeric instruction moves the numeric portion of 1-100 characters from one location in main memory to another. The zone bits of both fields are unchanged.

3.2.11.1 *Instruction fields*

Machine operation code

F Binary 1001 (9)

Address specification

A — Address of the leftmost position of Operand-A.

B — Address of the leftmost position of Operand-B.

Indexing specification

IA — Index register for determining effective address of Operand-A.

IB — Index register for determining effective address of Operand-B.

Common/partition specification

AC — If AC is 0, A is an address in partition.
If AC is 1, A is an address in Common.

BC — If BC is 0, B is an address in partition.
If BC is 1, B is an address in Common.

Length specification

LA — Tens position of length of both Operand-A and Operand-B.

LB — Units position of length of both Operand-A and Operand-B.

3.2.11.2 *Operand fields*

Operand-A address

If IA is 0, then A is the effective address.
If IA is 1, 2 or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address.
If IB is 1, 2 or 3, the corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

Operand-A and Operand-B are equal in length.

10LA + LB = Length of operands for Move Numeric instruction.

If 10LA + LB = 00, 100 is the length of the operands.

3.2.11.3 *Operation*

General description

The numeric portion of Operand-A is copied into the numeric portion of Operand-B, one position at a time, from left to right, starting with the leftmost position of Operand-A and writing it into the leftmost position of Operand-B.

Condition code

2, after completion of the Move Numeric instruction.

3.2.11.4 *Programming hints*

Move Numeric and Move Character

The Move Numeric instruction is similar to the Move Character instruction. The Move Numeric instruction will extract and copy only the numeric portion of a character (leaving the zone bits unchanged); the Move Character instruction will copy an entire character including both numeric and zone portions.

Overlapping operands

If Operand-A and Operand-B do *not* overlap, then Operand-A is unchanged by the Move Numeric instruction.

To shift the Operand-A numeric field one or more positions to the *left* (to a lower machine address) the Move Numeric instruction can be used when the operands overlap if the Operand-B address is not greater than the Operand-A address. The unlapped positions of Operand-A and all zone bits in both operands will be unchanged.

To propagate a given digit throughout a data field, put the digit into the leftmost position of the field, and use the Move Numeric instruction as follows:

Operand-A address is the address of the data field.

Operand-B address is the address of the data field + 1.

Operand length must be one less than the data field length.

The Move Numeric instruction enables the programmer to change the numeric portions of instructions. It is most frequently used in address modification (A and B fields). It is also useful in varying the LA and/or LB fields.

3.2.12 Multiply

The Multiply instruction computes the algebraic product of two 1 to 10 position numeric operands.

3.2.12.1 *Instruction fields*

Machine operation code

F Binary 0110 (6)

Address specification

A	Address of the leftmost position of Operand-A.
B	Address of the leftmost position of Operand-B, and Address of the leftmost position of Product field.

Indexing specification

IA	Index register for determining effective address of Operand-A.
IB	Index register for determining effective address of Operand-B

Common/partition specification

AC	If AC is 0, A is an address in partition. If AC is 1, A is an address in Common.
BC	If BC is 0, B is an address in partition. If BC is 1, B is an address in Common.

Length specification

LA	Length of Operand-A.
LB	Length of Operand-B.
LB + LA	Length of Product field.

3.2.12.2 *Operand fields*

Operand-A address

If IA is 0, then A is the effective address.
If IA is 1, 2, or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address.
If IB is 1, 2, or 3, the corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

If LA is 0, the length of Operand-A is 10 characters.
If LA is 1 to 9, the length of Operand-A is 1 to 9 characters.

If LB is 0, the length of Operand-B is 10 characters.
If LB is 1 to 9, the length of Operand-B is 1 to 9 characters.

Product field

The product field may be thought of as the multiplier field (Operand-B) with a rightward extension of a length equal to that of the multiplicand (Operator-A); hence, the product field will be located at the Operand-B address and will have the length LB + LA.

3.2.12.3 *Operation*

General description

Operand-A is the multiplicand.

Operand-B is the multiplier.

The product is developed in the extended Operand-B field. The extension is cleared to zeros before the following computation is begun:

1. The rightmost digit of the multiplier is put into a register to govern the number of times the multiplicand will be added into the rightmost positions of the product field
2. The rightmost position of the multiplier field is cleared to provide an extra left position for the add operation
3. The multiplicand is added into the rightmost positions of the product field the number of times specified by the governing multiplier digit stored in the register
4. Steps 1, 2, and 3 are repeated with the next left digit of the multiplier acting as governing digit. The multiplicand is repeatedly added into the next left positions of the product field. The process continues until the leftmost multiplier digit has served as governing digit

Bit-7 is set OFF in all positions of the product except the rightmost position which is set to the sign of the product.

Bit-7 ON = factor signs differed.
Bit-7 OFF = factor signs were alike.
Bit-5 is set ON in all positions of the product field.

Operand-A is unchanged by the multiply operation if the fields do not overlap.

An overflow condition will never occur if the numeric portions of the numeric values are 0 thru 9.

Condition codes

After completion of the Multiply instruction:

1 = Negative, non-zero product.
2 = Zero product.
3 = Positive, non-zero product.

3.2.12.4 *Programming hints*

Overlapped operands

In case of overlapped operands, the result is unspecified.

Overflow

Overflow will never occur if all characters in the numeric portions of the operands are the digits 0 to 9. Overflow can occur if the numeric portions of the operands contain the following digits:

binary 1010 (10)
1011 (11)
1100 (12)
1101 (13)
1110 (14)
1111 (15)

3.2.13 Read

The Read instruction moves data from an input device to sequential locations in main memory.

3.2.13.1 *Instruction fields*

Machine operation code

F Binary 0000 (0)

Channel specification

LB If bit-1 is 0, reading will be routed through the FAC.
If bit-1 is 1, reading will be routed through the IOC.

Mode specification

LB If bit-4 is 0, reading will be in the *fill* mode.
If bit-4 is 1, reading will be in the *non-fill* mode.

Input device specification

LA Device address 0-9 for IOC.
Device address 0-4 and 8 for FAC.

Input address specification

A Address of input area.

B If the input device is not the disc, B is the count. If the input device is the disc, B is the indirect disc address.

The indirect disc address points to a six-character field which contains the disc address.

Indexing specification

IA Index register for determining effective address of input area.

IB Index register for determining effective indirect disc address *or* effective count.

Common/partition specification

AC If AC is 0, A is an address in partition.
If AC is 1, A is an address in Common.

BC If B is a count, then BC is ignored.

If BC is 0, B is an address in partition.
If BC is 1, B is an address in Common.

Count specification

If the disc is the input device, the count is always 100 and is not specified in the Read instruction.

If the input device is not the disc, B is the count. A count of 0000 is interpreted as 10,000.

3.2.13.2 *Operation*

IOC general operation

A Read instruction that specifies data transmission through the IOC is executed incrementally. The instruction is first decoded, and parameters are set into registers A, B and P for the partition initiating the operation. A signal is sent to the IOC to alert the input device. Control is then relinquished to the next partition. The fulfilment of the Read instruction is performed between the execution of instructions in the other partitions. Before each instruction begins, the CPU stores one character for each IOC that has a character ready. This incremental operation proceeds as follows:

1 An IOC requests a character from the input device

2 The input device gives a character to the IOC which sets a signal to inform the CPU of "character ready"

3 Between instruction executions, the CPU discovers the signal, stores the character being held by the IOC, and updates the parameter registers

4 If the number of characters already transmitted has reached the count specified in the Read instruction, no more characters are requested. If the count has not been reached, steps 1, 2, 3, and 4 are repeated

If control returns to the partition which initiated the Read instruction before the count is satisfied, control simply passes to the next partition. If the count is satisfied when control returns to the partition which initiated the Read instruction, a condition code is set (see description of individual devices), the execution continues with the next sequential instruction following the Read instruction.

FAC general operation

A Read instruction that specifies data transmission through the FAC does not relinquish control to the next partition during data transmission. Instead, the CPU is devoted exclusively to storing data provided by the FAC until the entire count is satisfied. During this period the CPU does not service any IOC. Service to the IOCs resumes at the completion of the Read instruction.

Disc access sequence

A Read instruction addressing the disc does not typically pre-empt the CPU (as described above) immediately. It is sometimes necessary to wait until the disc is free, and then to wait while the heads move to the required cylinder. During either type of wait, control passes to the neighbouring partition, and returns again in normal sequence.

A disc is *free* if it is not bound to another partition. It is *bound* to a given partition as soon as the partition institutes a seek upon it; it remains bound until data transmission is complete.

If the disc is bound to another partition when a Read instruction is attempted, control merely passes to the next partition. The Read instruction will be attempted again when control returns to the host partition.

If the disc is free when a Read instruction is attempted, a seek is automatically instituted, and the disc is then bound to the host partition. If head movement is necessary, control passes to the next partition. Transmission begins when the heads reach the proper cylinder, when control returns to the host partition, and when the desired sector rotates into place.

If the heads are already on cylinder when the seek is instituted, control remains with the host partition. Transmission begins as soon as the desired sector rotates into place.

When the disc record is entirely transmitted, a condition code is set to indicate the outcome.

The CPU services any outstanding IOC for signals, and execution continues with the next sequential instruction following the Read instruction.

Succeeding instructions in the host partition which access the same cylinder will be executed without switching partitions. The first attempt to access another cylinder, however, will free the disc and pass control to the next partition. When control again returns to the host partition, the Read/Write instruction will be subject to the entire wait process (as described above).

Fill and non-fill

A Read instruction using the IOC will terminate prematurely if the input device sends the IOC a Unit Separator character. In such a case, the Unit Separator character is not stored. Remaining positions of the input area are normally filled with blank characters. If the non-fill option was requested (bit-4 of instruction field LB), the remaining positions in the input area are left undisturbed.

Condition codes

After completion of the Read instruction.

1 = Error
2 = Normal
3 = Flag
4 = Fault

3.2.14 Subtract

The Subtract instruction computes the algebraic difference between the numeric portions of the two operands. The difference replaces the second operand (the minuend) and leaves the first operand unchanged if the fields do not overlap.

3.2.14.1 *Instruction fields*

Machine operation code

F Binary 0111 (7)

Address specification

A Address of the leftmost position of Operand-A.

B Address of the leftmost position of Operand-B

Indexing specification

IA Index register for determining effective address of Operand-A.

IB Index register for determining effective address of Operand-B.

Common/partition specification

AC If AC is 0, A is address in partition.
If AC is 1, A is address in Common.

BC If BC is 0, B is address in partition.
If BC is 1, B is address in Common.

Length specification

LA Length of Operand-A.

LB Length of Operand-B.

3.2.14.2 *Operand-A address*

If IA is 0, then A is the effective address.
If IA is 1, 2, or 3, the corresponding index register is added to A to determine the effective address of Operand-A.

If AC is 1, the effective address lies in Common.

Operand-B address

If IB is 0, then B is the effective address.
If IB is 1, 2, or 3, the corresponding index register is added to B to determine the effective address of Operand-B.

If BC is 1, the effective address lies in Common.

Operand lengths

If LA is 0, the length of Operand-A is 10 characters.
If LA is 1 to 9, the length of Operand-A is 1 to 9 characters.

If LB is 0, the length of Operand-B is 10 characters.
If LB is 1 to 9, the length of Operand-B is 1 to 9 characters.

3.2.14.3 *Operation*

General description

The subtract operation proceeds from right to left starting with the rightmost character of Operand-A and Operand-B. Character by character, the algebraic difference is developed in Operand-B.

The hardware acts as though the sign of Operand-A were reversed. In every other respect the instruction behaves like the Add instruction.

If Operand-A is shorter than Operand-B, the operation proceeds normally until Operand-A is exhausted. After that, the process continues in similar fashion except that a zero character is automatically substituted every time the logic calls for a character from Operand-A. In effect, Operand-A is given enough preceding zeros to make it the same length as Operand-B.

If Operand-A is longer than Operand-B, subtraction stops after the leftmost position in Operand-B has been subtracted. The remaining positions in Operand-A are ignored and do not affect the difference or the condition code.

The algebraic sign of the difference is placed in bit-7 of the rightmost position of Operand-B, and bit-5 is turned on. Except for the rightmost character, the other zone bits of Operand-B are unchanged. Operand-A is unchanged by the subtract operation.

If the difference exceeds the capacity of Operand-B, a carry-to-the-left from the leftmost position does not occur. Condition code 4 is set to indicate the overflow.

Condition codes

After completion of the Subtract instruction.

1 = Negative, non-zero difference.
2 = Zero difference.
3 = Positive, non-zero difference.
4 = Overflow.

3.2.14.4 *Programming hints*

Overlapped operands

In case of overlapped operands, the result is unspecified.

3.2.15 Write

The Write instruction transmits data from sequential locations in main memory to an output device. A control option enables the Write instruction to communicate control information to the input or output device.

3.2.15.2 *Instruction fields*

Machine operation code

F Binary 0001 (1).

Channel specification

LB If bit-1 is 0, writing will be routed through the FAC.
If bit-1 is 1, writing will be routed through the IOC.

Write control specification

LB If bit-2 is 0, normal Write.
If bit-2 is 1, Write Control.

Output device specification

LA — Device address 0-9 for IOC.
Device address 0-4 and 8 for FAC.

Output address specification

A — Address of output area.

B — If the output device is not the disc, B is the count.
If the output device is the disc, B is the indirect disc address.

The indirect disc address points to a six-character field which contains the disc address.

Indexing specification

IA — Index register for determining effective address of output area.

IB — Index register for determining effective indirect disc address *or* effective count.

Common/partition specification

AC — If AC is 0, A is an address in partition.
If AC is 1, A is an address in Common.

BC — If B is a count, BC is ignored.

If BC is 0, B is an address in partition.
If BC is 1, B is an address in Common.

Count specification

If the disc is the output device, the count is always 100 and is not specified in the Write instruction.

If the output device is not the disc, B is the count. A count of 0000 is interpreted as 10,000.

3.2.15.2 *Operation*

IOC general operation

A Write instruction that specifies data transmission through the IOC is executed incrementally. The instruction is first decoded, and parameters are set into registers A, B, and P for the partition initiating the operation. A signal is sent to the IOC to alert the output device. Control is then relinquished to the next partition. The transmission of characters occurs between the execution of instructions in the other partitions. Before each instruction begins, the ACU sends one character to each IOC which is ready to accept one. This incremental operation proceeds as follows:

1. The IOC sets a signal to inform the ACU that it is ready to accept a character from the output area
2. Between instruction executions, the ACU discovers the signal and checks the count balance. If the count has been reached, no more characters are sent to the IOC. If the count has not been reached, steps 3, 4, 1, and 2 are repeated, in that order
3. The ACU gives a character to the IOC and updates the parameter registers
4. As soon as it can, the output device accepts the character

If control returns to the partition which initiated the Write instruction before the count is satisfied, control simply passes to the next partition. If the count is satisfied when control returns to the partition which initiated the Write instruction, a condition code is set (see description of individual devices), and execution continues with the next sequential instruction following the Write instruction.

FAC general operation

A Write instruction that specifies data transmission through the FAC does not relinquish control to the next partition during data transmission. Instead, the ACU is devoted exclusively to feeding data to the FAC until the entire count is satisfied. During this period the ACU does not service any IOC. Service to the IOCs resumes at the completion of the Write instruction.

Disc access sequence

A Write instruction addressing the disc does not typically pre-empt the ACU (as described above) immediately. It is sometimes necessary to wait until the disc is free, and then to wait while the heads move to the required cylinder. During either type of wait, control passes to the neighbouring partition, and returns again in normal sequence.

A disc is *free* if it is not bound to another partition. It is *bound* to a given partition as soon as the partition institutes a seek upon it; it remains bound until data transmission is complete.

If the disc is bound to another partition when a Write instruction is attempted, control merely passes to the next partition. The Write instruction will be attempted again when control returns to the host partition.

If the disc is free when a Write instruction is attempted, a seek is automatically instituted, and the disc becomes bound to the host partition. If head movement is necessary, control passes to the next partition. Transmission begins when the heads reach the proper cylinder, when control returns to the host partition, and when the desired sector rotates into place.

If the heads are already on cylinder when the seek is instituted, control remains with the host partition. Transmission begins as soon as the desired sector rotates into place.

When the disc record is entirely transmitted, a condition code is set to indicate the outcome. The ACU services any outstanding IOC signals, and execution continues with the next sequential instruction following the Write instruction.

Succeeding instructions in the host partition which access the same cylinder will be executed without switching partitions. The first attempt to access another cylinder, however, will free the disc and pass control to the next partition. When control again returns to the host partition, the Write instruction will be subject to the entire wait process (as described above).

Write control mode

A Write instruction may specify the transmission of control characters to the external input/output device by having bit-2 of the LB instruction field ON. The information in the output area is sent to the external device one character at a time and exerts a controlling effect. The particular effect depends upon the information transmitted and upon the external device. As soon as the last character is accepted by the external device, program execution is free to continue even though the controlling effect is not yet realised.

Condition codes

After completion of the Write instruction:

1 = Error
2 = Normal
3 = Flag
4 = Fault

4.1 The Multiterminal Input/Output Channel II (MTIOC II)

In this section the terms MTIOC II and IOC are used interchangeably.

The MTIOC II transfers data and supervisory information between the ACU and the peripherals that can be connected to it by means of this IOC. These peripherals are the Workstation (typebar or Visual Display Unit (VDU)) and line printer. Each peripheral is connected to the IOC by means of a line unit. There may be up to ten line units and therefore up to ten peripherals. Peripherals are serviced by the IOC one at a time and a request to the IOC for data transfer can come from the ACU (under program control) or from a peripheral.

The IOC performs two fundamental functions, Input/Output and Polling, one or other of which it carries out continuously.

The Polling operation enables the IOC to detect a Service Request from any peripheral, or a Load Request from the peripheral designated as device zero.

The Input/Output operation enables characters to be transferred one by one between the ACU and a peripheral.

4.1.1 Polling

The peripheral designated device zero is the only device that may request the loading of a new program into the partition to which the IOC is attached.

The IOC interrogates each line unit (including that for device zero) in turn (whether or not a peripheral is attached to it) to determine if the peripheral has made a Service Request. Alternate interrogations are directed at device zero for a Load Request.

4.1.1.1 *Service Request*

When the IOC encounters a Service Request, control passes to the ACU which, at the next opportunity, switches past the partition and handles the data transfer by the Interrupt procedure (see section 1.3.3). That partition is not allocated processing time again until the transfer is complete.

4.1.1.2 *Load Request*

A Load Request is signalled when a Program Check or an Address Check is encountered. The IOC passes control to the ACU which suspends work in that partition and, when signalled to do so by the operator, loads ten characters (an instruction) from device zero into location zero, and executes them. This causes a program to be loaded and execution to begin.

Recovery from Program Check option

As an alternative to the above procedure, an option is available whereby (without operator intervention) the loading of ten characters is bypassed and execution continues at location zero.

This option is selected at installation and enables the program to report the failure and/or recover from it. Once the user program has been loaded and is running, a branch instruction to some error routine is inserted at location zero and then, should a check occur, control passes to the error routine which can make use of the information stored by the ACU in the Error Register 41 to 44P to report on the failure.

4.1.2 Input/Output

Each phase of the input/output procedure is undertaken by the IOC operating in successive modes:

1. Escape mode, to which the IOC returns after each I/O operation
2. Select mode, in which the IOC addresses and readies the required line unit for input or output
3. Read mode, which readies the IOC for receiving data characters or status characters from a selected line unit
4. Write mode, which readies the IOC for transmitting characters to the selected line unit

Thus, the sequence of modes through which the IOC would pass during execution of a Read instruction would be Escape, Select, Read and back to Escape.

During the Read and Write phases, supervisory instructions can be transmitted as well as data characters. These supervisory instructions are special characters, not part of the System Ten set, which can be generated by the IOC altering bit 6 of the character, to form one of the control characters shown in the first two columns of Figure 2.2. The control characters have the following meanings:

Character	*Meaning*
NUL	Null
SOH	Start of Heading (CC)
STX	Start of Text (CC)
ETC	End of Text (CC)
EOT	End of Transmission (CC)
ENQ	Enquiry (CC)
ACK	Acknowledge (CC)
BEL	Bell (audible or attention signal)
BS	Backspace (FE)
HT	Horizontal Tabulation (punched card skip) (FE)
LF	Line Feed (FE)
VT	Vertical Tabulation (FE)
FF	Form Feed (FF)
CR	Carriage Return (FE)
SO	Shift Out
SI	Shift in
DLE	Data Link Escape (CC)
DC1	Device Control 1
DC2	Device Control 2
DC3	Device Control 3
DC4	Device Control 4 (Stop)
NAK	Negative Acknowledge (CC)
SYN	Synchronous Idle (CC)
ETB	End of Transmission Block (CC)
CAN	Cancel
EM	End of Medium
SUB	Substitute
ESC	Escape
FS	File Separator (IS)
GS	Group Separator (IS)
RS	Record Separator (IS)
US	Unit Separator (IS)
DEL	Delete

Note:

(CC) Communication Control

(FE) Format Effector

(IS) Information Separator

4.2 The Digital Clock

The Digital Clock is a specialised IOC not related to input/output that provides a hardware reference for a software clock accessible to other partitions.

Included in the DMF II software is a program called CLOCK which is loaded into the partition linked to the Digital Clock IOC. The program uses the counter character generated by the IOC to maintain several fields in Common. These fields are accessible to other partitions and each holds the data and time in a different format, for example YYMMDD, YYDDD, day of the week as a number from 1 to 7, hours and decimals of hours, or minutes and seconds.

The Digital Clock produces one number every second and offers it to the ACU by indicating Service Request. This number is a single character BCD digit from 1 to 9 representing the time interval in seconds that has elapsed since the last character was transferred. Normally the Service Request is answered by a Read instruction, the digit is transferred and used to update the fields in Common. If for some reason the Service Request cannot be answered within the one second time interval, the IOC increases the offered number by one and the Service Request remains set. The offered number can reach the count of nine, but on the count of ten an error is indicated and the operator is requested to reinitialise the clock.

The Digital Clock derives its frequency standard from the AC power supply to the machine; 50 or 60 Hz is set at installation. The start time for the clock is set into Common from a workstation.

The Write and Write Control instructions are illegal for the Digital Clock IOC. Whenever either of these instructions is encountered, the ACU initiates a Load Request.

4.3 The Multi-Device IOC II

The Multi-Device IOC II (MDIOC II) provides an interface between the Model 22 processor and up to ten terminals. Each terminal is connected to the IOC by a separate twisted pair of wires of maximum length eight miles. IOC operations take place under program control.

The terminals are polled in turn by the IOC and when a terminal requests service, the IOC acknowledges the request and allows the terminal to transmit. Following receipt of a message from a terminal, the IOC transmits a reply and the terminal severs the connection.

Optionally at installation, the System Ten character set may be extended to include the full 128-character ASCII set. The following control characters (those from columns 0 and 1 of the ASCII character set diagram) can be recognised: ENQ, ETB, ETX and SOH.

Transmission is half-duplex, bit serial and asynchronous by character. Transmission speed is 1200 (for wire length up to eight miles) or 3600 bps (for wire length up to 5000 ft) from the terminal to the IOC and 120 or 1200 bps from the IOC to the terminal.

The way in which the IOC handles the program instructions Read, Write, Write Control and Branch on service request are described in the sections that follow.

4.3.1 Branch-on-service-request

The fields of the MDIOC II Branch-on-service-request instruction are interpreted as follows:

Field	*Contents*
F	1011
LA	7
LB	0 or 9
A	Location where device address is to be stored
B	Location the program will branch to

If a terminal has requested service at the time a Branch-on-service-request instruction is executed, then the four bits which form the device address replace the four low order bits of the location specified by the A address, and the program branches to the location specified by the B address. An unsuccessful Branch-on-service-request results in the next program instruction being executed.

4.3.2 Read

The fields of the MDIOC II Read instruction are interpreted as follows:

Field	*Contents*
F	0000
LA	Ignored
LB	1
A	Location where data read is to be stored in Common or partition
B	Number of characters to be transferred. Must be greater than the maximum message length (limited by software to 225 characters)
IA, IB	A and B may be indexed

A Read instruction will only be accepted after a successful Branch-on-service-request; the IOC will Check if a Read instruction occurs at any other time. After the completion of a Read, normal status is set unless one of the following transmission errors occurred:

1 Bad parity
2 No stop bit
3 Interrupt still set, indicating that the previous character has not been transferred into memory
4 Last set, indicating an incorrect character count for the instruction
5 Block Character Count (BCC) error

If a power failure occurs, the instruction will be terminated.

4.3.3 Write

The fields of the MDIOC II Write instruction are interpreted as follows:

Field	*Contents*
F	0001
LA	Ignored
LB	1
A	Location from which data is to be written, in Common or partition
B	Exact number of characters to be transferred
IA, IB	A and B may be indexed

The processor cannot initiate a call to a terminal, therefore, a Write instruction will only be accepted after a Read; the IOC will cause a Program Check if a Write is encountered at any other time.

Extended character set

Write can cause to be transmitted any character from the full ASCII set. This is achieved by preceding by @ any character other than those in the System Ten subset. The use of @ to extend the character set causes character conversions to take place as follows:

1 @ followed by a character from column 2 or 3 of the character set diagram translates that character into the equivalent one from column 6 or 7 respectively
2 @ followed by a character from column 4 or 5 translates into the equivalent one from column 0 or 1 respectively

Only the character following @ is translated and then transmitted. However, @ followed by @ results in the second @ being transmitted in normal mode. Note that the @ used to extend the character set must be included in the character count.

4.3.4 Write Control

The fields of the MDIOC II Write Control instruction are interpreted as follows:

Field	*Contents*
F	0001
LA	Ignored

Field	*Contents*
LB	3
A	Location from which data is to be written, in Common or partition
B	Exact number of characters to be transferred
IA, IB	A and B may be indexed

A Write Control instruction will only be accepted after a Read or after a successful Branch-on-service-request; the IOC will set Check if a Write Control occurs at any other time. If a Write Control is encountered after a Branch-on-service-request, the IOC ignores it and resumes polling. In this way a Service Request may be ignored.

After a Read, a Write Control is used to reply to a terminal. The reply can consist of any character from columns 0 to 3 (typically ACK or NAK) of the ASCII chart. A Write Control is always terminated with normal status.

4.4 The Synchronous Communications Adaptor

4.4.1 Introduction

The Synchronous Communications Adaptor (SCA) enables programs within a System Ten Computer to transmit data to, and receive data from, remote computers. There are two versions. One (referred to as SCA-2) has *dial out* capability, while the other (referred to as SCA-1) does not. Both versions of the SCA may receive and transmit data over telephone lines once the connection has been initiated. However, only the SCA-2 is capable of initiating the connection.

The SCA is connected to the communications adaptor of the remote computer by telephone lines and a pair of modems. The SCA replaces two consecutive, physically adjacent IOCs in the Model 22, and has one partition associated with it. Note that if installed such that it replaces IOC 19, an SCA-1 may replace only one IOC. The SCA partition has the partition number associated with the lower of the two replaced IOCs. No peripheral devices may be attached to an SCA partition. Other partitions cannot address the SCA directly. Data to be sent or received via the SCA must either be passed through Common or via an FAC device.

For the most part, an SCA partition is like any other partition. It receives and loses control of the central processor by the same rules as does any other partition, and the program which resides in it may contain any valid System Ten machine instruction. Input and output (that is, receiving and transmitting) is carried out in the normal way, by the interrupt procedure. An SCA partition differs from a normal partition in the following two respects:

1 Since the SCA partition cannot have attached to it any input devices capable of generating a load sequence, the program which is to reside in it must be loaded in a special way (the loading process is the same as for MDTS partitions)

2 The device number (LA) of input/output instructions is interpreted in an unusual manner. The device number (LA) of a Read or Write instruction, or the device number minus one (LA-1) of a Read Control instruction is taken as specifying the number of control characters to be transmitted before proceeding with the operation. Thus, besides the receiving data, a Read or Read Control instruction may also involve the transmitting of data

There are two basic configurations in which it is expected the SCA will be used: point-to-point and centralised, multi-point.

A centralised, multi-point configuration may, in one sense, be thought of as a party line. All remote stations listen to everything which the master station transmits, although each actually accepts only those messages which are specifically addressed to it. In a two-wire configuration, all stations listen to everything which is transmitted by anybody, and one remote station can transmit directly to another (in such a configuration, any of the stations could usurp the role of master). In a four-wire configuration, one remote station cannot listen in to what another transmits to the master, and one remote station *cannot* transmit directly to another.

4.2.2 Control characters

Up to nine control characters may be transmitted at the start of an SCA Read or Write operation, and up to eight may be transmitted at the start of an SCA Read Control operation. Details regarding how control characters are sent are presented under SCA INPUT/OUTPUT INSTRUCTIONS, further on page 4–7. The meaningful control characters are as follows:

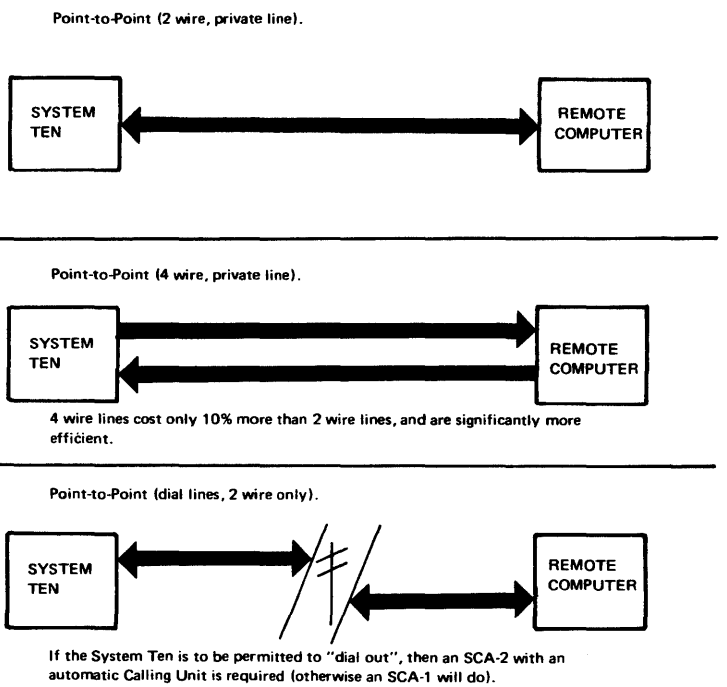


Figure 4.1 Point to point configuations

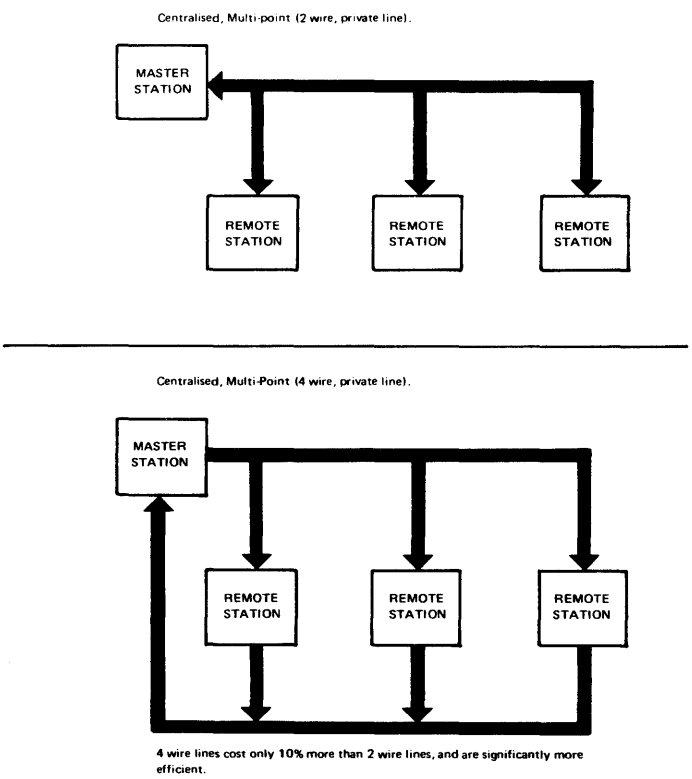


Figure 4.2 Centralised Multipoint Configurations

System Ten internal character	*External control character*	*Definition*
V	SYN	Synchronous idle
A	SOH	Start of heading block
B	STX	Start of text block
W	ETB	End of an intermediate text block within a message
C	ETX	End of the final, or only, text block in a message
F	ACK	Affirmative acknowledgement
U	NAK	Negative acknowledgement
E	ENQ	End of poll/select sequence (also used for signalling "disregard the last block")
D	EOT	End of transmission (also used by the master station in a centralised, multi-point configuration to tell all the remote stations that it is about to poll or select someone)

4.4.2.1 *DLE pairs*

In addition, there are four pairs of characters which each act as a control character:

1 The control character DLE (System Ten internal character P) immediately followed by the control character EOT (System Ten internal character D) is defined as mandatory disconnect. This is used in a dial line configuration to notify the remote station that you are hanging up

2 The other three pairs consist of the control character DLE immediately followed by the non-control character zero, one, or question mark. These pairs are defined as follows:

DLE pair	*Meaning*
DLE 0	Affirmative acknowledgement (ACK 0)
DLE 1	Affirmative acknowledgement (ACK 1)
DLE ?	Wait before transmitting (WABT)

If the SCA detects any of these four pairs while receiving data from the remote computer, it drops the DLE, passes the second character to the central processor, turns ON the Flag Condition Indicator (condition code 3) in the central processor, and terminates the Read operation. For example, if the SCA is listening for a one-character response from the remote computer, and if the remote computer sends a DLE 0, then the zero is passed to the central processor, the Flag Condition Indicator (condition code 3) in the central processor is set ON, and the Read operation is terminated.

4.4.2.2 *Terminator control characters*

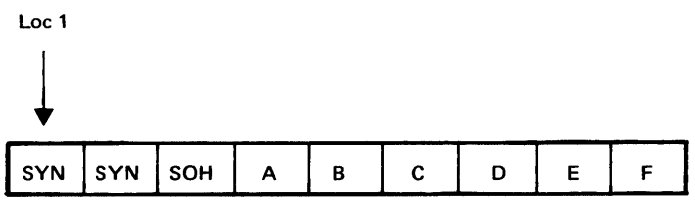
If the SCA detects any of the following control characters amongst data, it is receiving from the remote computer, then it automatically terminates the Read or Read Control operation:

ETB ETX EOT ACK NAK ENQ a DLE pair

4.4.3 SCA Input/Output instructions

4.4.3.1 *Write*

The Write instruction transmits control characters and data to the remote computer. The control characters and data characters must be accessed from a single output area. For example, if the control characters SYN, SYN, and SOH, plus the data characters A, B, C, D, E, and F are to be transmitted by one Write Instruction, then they must be accessed from a single output area, as follows (where LOC1 is the location pointed to by the A field of the instruction):



The fields of an SCA Write instruction are interpreted as follows:

F : Binary 0001 (1)

A : Address of output area

IA : Index register for A

AC : Common flag for A

LA : Number of leading control characters

B : Count (includes the number of leading control characters)

IB : Index register for B

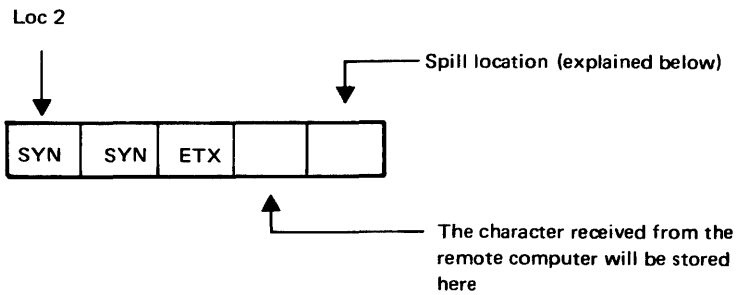
BC : Ignored

LB : Must be 1

4.4.3.2 *Read*

The Read instruction transmits control characters, and then immediately readies the SCA for characters from the remote computer. The control characters to be transmitted, and the locations reserved for receiving the characters from the remote computer must together constitute the input area. For example, if the control characters SYN, SYN, and ETX are to be transmitted, and if a one character response (for example, an ACK or a NAK) is expected from the remote computer, then the input area pointed bo by the Read instruction should be as follows (where LOC2 is the location pointed to by the A field of the instruction):

Note: The count B of the instruction is 5



It is intended that SCA Read operations normally be terminated by the receipt of a terminator control character. If an SCA Read operation is terminated by exhausting the count (B) of the Read instruction, then Error Condition Indicator (condition code 1) in the ACU is set ON. Therefore, every SCA Read instruction should normally ask for more characters than are actually expected.

Accordingly, the above sample input area contains an extra location (referred to as spill location). If, as expected, the remote computer sends an ACK or a NAK (both of which are terminator control characters), then, after the F or U has been stored in the fourth location of the input area, the operation is terminated with nothing having been stored in the spill location. If, on the other hand, the remote computer sends 25 characters of data, then, after the first two characters have been received and stored in the fourth and fifth locations of the input area, the operation is terminated, and the Error Condition Indicator (condition code 1) is set ON.

SYN control characters sent by the remote computer are neither counted (when received) nor passed on to the input area.

If, at any time during the operation, ten seconds elapse with no data (excluding SYNs) received, then the Fault Condition Indicator (condition code 4) in the central processor is set ON, and the operation is terminated. This method of terminating an SCA Read operation is referred to as *time-out*.

The fields of an SCA Read instruction are as follows:

F : Binary 0000 (0)

A : Address of input area

IA : Index register for A

AC : Common flag for A

LA : Number of leading control characters

B : Count (includes the number of leading control characters)

IB : Index register for B

BC : Ignored

LB : Must be 1

4.4.3.3 *Write Control*

The Write Control instruction is of use only in dial line configurations and four-wire, centralised, multi-point configurations. In a dial line configuration, it is used to dial out and to hang up. In a four-wire, centralised, multi-point configuration, it is used to eliminate some of the overhead from line turnround.

Dialling out

The fields of the SCA Write Control instruction for dialling out are interpreted as follows:

F : Binary 0001 (1)

A : Address of an output area which contains a 10- or 7-digit telephone number (10 if the number includes an Area Code, 7 if it does not)

IA : Index register for A

AC : Common flag for A

LA : Must be 1

B : Count (either 7 or 10)

IB : Index register for B

BC : Ignored

LB : Must be 3

When executed, this instruction initiates the connection with the specified telephone number (analagous to a person dialling a number on a telephone). It has no effect if executed in an SCA-1 partition, or in an SCA-2 partition with no Automatic Calling Unit.

If there is no answer, or if the wrong party answers, then the instruction is terminated and the Error Condition Indicator (condition code 1) in the central processor is set ON. It should be noted that when either of these events happens, the programmer need *not* execute a Write Control instruction for "hanging up".

Hanging up

The fields of an SCA Write Control instruction for "hanging up" are as follows:

F : Binary 001 (1)

A, IA, AC : Ignored

LA : Must be 2

B, IB, BC : Ignored

LB : Must be 3

This instruction (whether executed in a partition attached to an SCA-1 or an SCA-2) terminates the connection with the remote computer. It has no effect if no connection exists, or if used in something other than a dial line configuration.

Constant carrier

Line turnround means that one computer stops transmitting and starts listening, and the other computers starts transmitting. Normally, every line turnround involves approximately 200 ms. of overhead. Constant carrier eliminates this overhead. In order to be established, constant carrier must be specified at both ends of the connection.

In four-wire, point-to-point configurations, both computers will normally have constant carrier permanently wired into them.

In four-wire, centralised, multi-point configurations, the master station will normally have constant carrier permanently wired into it. To establish constant carrier between the master station and a remote station, the remote station would execute a Write Control instruction.

The fields of an SCA Write Control instruction for establishing constant carrier are interpreted as follows:

F : Binary 0001 (1)

A, IA, AC : Ignored

LA : Must be 0 (zero)

B, IB, BC : Ignored

LB : Must be 3

Once constant carrier has been established, the master station and the remote station may exchange heading blocks, text blocks, and responses without the 200 ms. of overhead at each line turnround.

Constant carrier is lost whenever either station transmits an EOT control character.

4.4.3.4 *Read control*

The Read Control instruction is meant to be used in the SCA partition of a System Ten which is serving as a remote station in a centralised, multi-point configuration. In such a configuration, communication between the master station and any of the remote stations is initiated by either polling or selecting. In polling, the master station queries a remote station to see if the remote station has anything to transmit. In selecting, the master station signals a remote station that the master has something to transmit to that remote station. A System Ten which is serving as a remote station in a centralised, multi-point configuration is assigned a one character remote station address. The primary functions of the Read Control instruction are to assign a remote station an address, and to start that station listening to the master station's transmission line for that address.

Remote station address

A System Ten's remote station address may be any System Ten internal character. The first character in the input area pointed to by the A field of the Read Control instruction is moved to the SCA's Address Compare Register. Only bits 1 to 5 of this character are used in address comparisons. Consequently, there are in fact only 32 possible remote station addresses (in assigning addresses, the characters in columns 2 and 3 of the ASCII code chart represent the same address as the corresponding characters in columns 4 and 5).

Poll/select sequence

Minimally, the poll/select sequence transmitted by the master station consists of two characters: a remote station's address followed by an ENQ control character. Optionally (as determined by the programming conventions for each individual configuration), up to nine additional characters may be included between the address and the ENQ. Bit 7 of the remote station address (that is, the first character) in a poll/select sequence is meant to be used for signifying whether the station is being polled or selected. What it means when ON or OFF is determined by the programming conventions for each individual configuration. However, if it is ON (that is, the first character of the poll/select sequence is one from columns 4 or 5 of the ASCII code chart), then the Flag Condition Indicator (condition code 3) in the remote station's central processor is set ON.

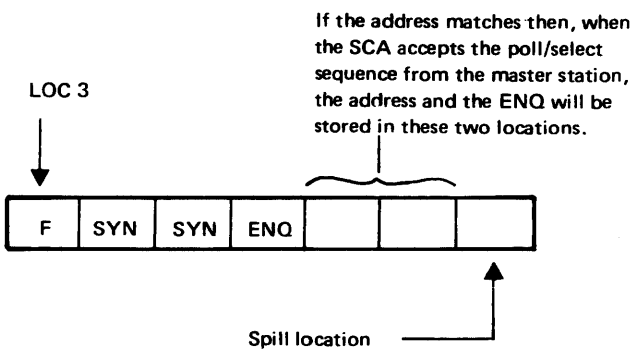
Polling and selecting

Before transmitting a poll/select sequence, the master station forces a line break by transmitting the control character EOT.

All remote stations listen (by Read Control instructions) to the master station's transmission line. Whenever a line break occurs, each station examines the next non-SYN character. If the character is a control character, then all the remote stations merely continue listening for the next line break. If the character is not a control character, then each station automatically compares bits 1 to 5 of that character with the corresponding bits in its Address Compare Register. If a station's address matches the address transmitted by the master station, then that station accepts the poll/select sequence, and the program in the remote station's SCA partition responds appropriately. Meanwhile, the other remote stations continue listening for the next line break.

Instruction format

The one character address, plus whatever control characters are to be transmitted, plus the locations reserved for accepting the poll/select sequence must together constitute the input area. For example, if the remote station's address is to be F, and if the control characters SYN, SYN, and ENQ are to be transmitted, and if the master station is transmitting the minimal poll/select sequence, then the input area pointed to by the Read Control instruction would be as follows (where LOC3 is the location pointed to by the A field of the instruction):



The fields of an SCA Read Control instruction are interpreted as follows:

F : Binary 0000 (0)

A : Address of input area

IA : Index register for A

AC : Common flag for A

LA : Number of leading control characters + 1

B : Count (includes the number of leading control characters + 1)

IB : Index register for B

BC : Ignored

LB : Must be 3

Time-out

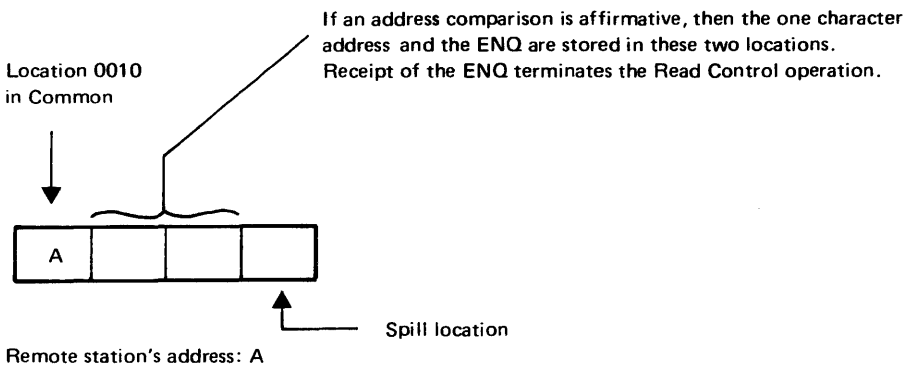
After transmitting the specified number of control characters, the SCA starts listening to the master station's transmission line. If, within ten seconds after it starts listening, the SCA does not sense an affirmative address comparison, then the Fault Condition Indicator (condition code 4) is set ON, and the Read Control operation is terminated. This method of terminating an SCA Read Control operation is referred to as time-out.

Sample instruction sequence

A typical instruction sequence executed by the program in a remote section's SCA partition might be as follows (assuming that the master station, when polling or selecting, sends a minimal poll/select sequence):

1 Execute a Read Control instruction which assigns an address, does not transmit any control characters, and which (in case of an affirmative address comparison) expects to receive two characters

A:	0010 (in Common)	B:	4
IA:	0	IB:	0
AC:	1	BC:	0
LA:	1	LB:	4



Remote station's address: A

2 Test the Error Condition Indicator (condition code 1).

If ON, then either something was wrong with the transmission, or the operation was terminated due to time-out. In either case, branch back to 1, and execute the Read Control instruction again.

If OFF, then the Read Control operation was terminated, by something other than time-out (presumably by the ENQ of a poll/select sequence), and there was nothing wrong with the transmission. Proceed with 3

3 Test the Flag Condition Indicator (condition code 3) to determine whether the station was polled or selected

ON (the station address in the poll/select sequence is a character from columns 4 or 5 of the ASCII code chart).

OFF (the station address in the poll/select sequence is a character from columns 2 or 3 of the ASCII code chart).

What this condition code means when ON or OFF is determined by the programming conventions for each individual configuration

4.4.4 Programming conventions

The following sections contain certain conventions which should be ahered to when using the SCA.

4.4.4.1 *Leading SYNs*

To establish character synchronisation between itself and the remote computer, the SCA automatically transmits a pair of SYN control characters at the start of every transmission (that is, at the start of every Read or Read Control operation which involves the sending of control characters, and at the start of every Write operation). However, under cetain circumstances this may not be sufficient.

Therefore, it is advised (though not required) that the program transmits up to half dozen SYN control characters at the start of every transmission.

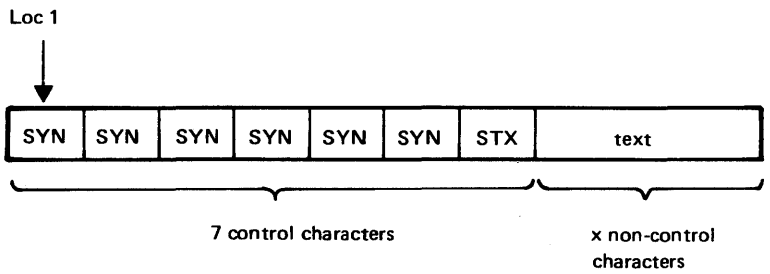
4.4.4.2 *Transmitting messages*

A transmission may consist of a single character such as an ACK or a NAK. It may also consist of one or more messages. The minimum size of a message may be one text block. It may, however, comprise a heading block followed by one or more text blocks.

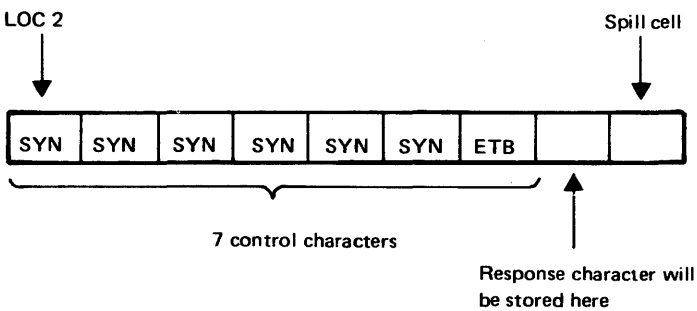
Text blocks

A text block begins with the control character STX and ends with the control character ETB or ETX (an ETB signals the end of an intermediate text block in a message, while an ETX signals the end of the final, or only, text block in a message). Since control characters can only be transmitted at the start of a Write, Read, or Read Control operation, every text block requires two instructions:

WRITE INSTRUCTION: A:LOC 1 LA: 7 B: 7+x LB: 1



READ INSTRUCTION: A: LOC 2 LA: 7 B: 9 LB: 1



The second instruction must be a Read or Read Control instruction for reasons explained below.

At the end of each text block, the SCA transmits a longitudinal parity character. In response to the ETB (or ETX) and the longitudinal parity character, the program in the remote computer should transmit one of the following:

External control character	*Internal character received*	*Meaning*
ACK	F	Reception satisfactory
DLE0 (ACK0)	0	
DLE1 (AXK1)	1	
NAK	U	Reception not satisfactory. Transmit the block again.

If the ETB or ETX which terminates a text block is sent by a Write instruction, then, after it has been sent, the SCA proceeds to send either a string of SYNs (until time-out occurs) or whatever additional characters the instruction specified be sent. In either case, because it is busy transmitting, the SCA will miss the response.

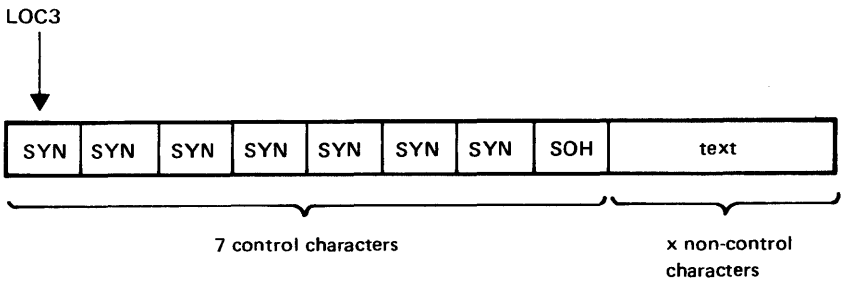
If, on the other hand, the ETB or ETX is sent by a Read or Read Control instruction, then, after it is sent, the SCA immediately begins listening for the response.

The end of transmission is signalled by the control character EOT.

Heading blocks

A heading block begins with the control character SOH and is terminated by the STX of the first text block. Thus, a heading block requires only one instruction (this instruction *must*, however, be immediately followed by the pair of instructions which transmits the first text block):

WRITE INSTRUCTION: A: LOC 3 LA: 7 B: 7 + x LB: 1



4.4.4.3 *Responses to polling and selecting*

In a centralised, multi-point configuration, communication between the master station and any of the remote stations is initiated by polling and selecting (this is discussed in detail in section 4.4.3.4).

When polled, a remote station should respond in either of two ways:

External control character	*Meaning*
EOT	The remote station has nothing to transmit to the master
SOH or STX	The remote station has a message to transmit (the SOH or STX is the initial character of the message)

When selected, a remote station should respond in one of three ways:

External control character	*Meaning*
AK	The remote station is not ready to receive the message from the master
DLE?	(WABT) The remote station is not yet ready to receive the message, but will be momentarily. The master station should wait a moment, and then select the remote station again
DLE 0	The remote station is ready to receive the message from the master station

4.4.4.4 *Answering calls in dial line configurations*

Whenever a station in a dial line configuration is called, the Service Request signal in the central processor is set ON. Thus, stations in dial line configurations should periodically test for this signal by executing the Branch-on-service-request variant of the Branch instruction. If the signal in ON, then zero is stored in the numeric portion of the location pointed to by the A field, and control passes to the location pointed to by the B field. If the signal is OFF, then execution proceeds with the next sequential instruction.

4.4.5 Condition codes after SCA operations

4.4.5.1 *Condition code 1*

Note that whenever condition code 1 (FAULT) is set ON, condition code 1 (ERROR) is also set ON automatically.

Read and Read Control

Operation terminated by exhausing the count (B) rather than by the receipt of a terminator control character.

Character received before the first of the preceding two characters could be passed on to the central processor (the SCA has a two character buffer). It is very unlikely that this will happen.

EOT detected within the text of a message.

Parity error (either vertical or longitudinal).

Loss of carrier, or receipt of 15 consecutive one bits.

Write Control

No answer, or wrong party answered.

4.4.5.2 *Condition code 2*

Read

DLE pair received

Read Control

Bit-7 of the first character of poll/select sequence is ON (signifying that the station was either polled or selected, as decided upon for each individual configuration).

Write Control

Incoming call received just as the SCA was about to initiate a call. In this case, the Write Control instruction for initiating the call is terminated immediately.

4.4.5.3 *Condition code 4*

Write

MODEM's power is off

Read

MODEM's power is off.

10 seconds elapsed without receiving any non-SYN character (time out)

Write Control

MODEM's power or Automatic Calling Unit's power is off.

Read Control

MODEM's power is off.

10 seconds elapsed without an affirmative address comparison (time out)

4.5 The Asynchronous Communications Adaptor

The Asynchronous Communications Adaptor provides an interface between the Model 22 processor and remotely located peripheral equipment, through modulator/demodulators (modems). The modems may be connected by telephone lines (direct or dialled) and must conform to the following description:

1 Data transmission mode is asynchronous (start/stop)
2 Data transmission rate is software-selectable at 150, 300, 600, 1200 and 1800 bits per second
3 Character length is 10 bits at various bit rates
4 Character set is 64-character subset of ASCII code
5 Character format is start bit, seven data bits, odd or even parity bit, and one stop bit
6 Bit sequence is least significant bit first

If dialled calls are to be originated by the Model 22 processor, an optional dial-out printed circuit card (CH 7) is available for use with an Automatic Calling Unit. The ACA can, however, answer incoming calls from dial system lines without requiring the dial-out option. An ACA without dial-out capability can be used for manually dialled calls.

The ACA is specifically designed to be compatible with three system configurations shown in Figure 4.3.

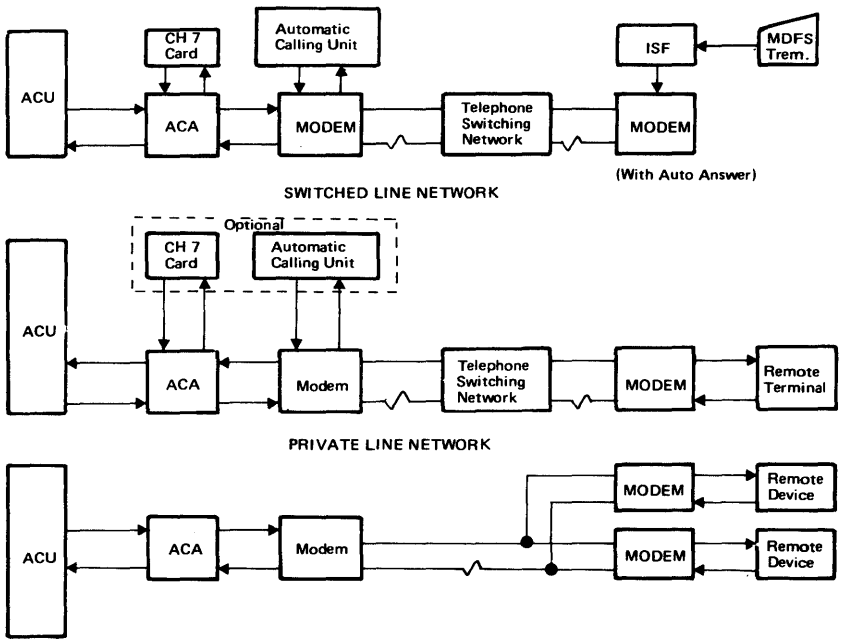


Figure 4.3 ACA system applications

In the Individual Store and Forward (ISF) configuration, the ACA, CH 7 card and an Automatic Calling Unit are used to collect point of sale information from remote terminals. ISF station numbers are individually dialled through the Automatic Calling Unit. After completing the connection, the ISF transmits accumulated data to the ACA. In this application, the ACA operates in a receive-only, simplex mode, except for reverse (back) channel control signals, also called Supervisory channel signals.

In the switched line application, the ACA is shown in a half duplex configuration using general ASCII communication procedures. Line control, error control and data messages are transmitted over the switched data channel as control character and data character sequences combined in any given message. In this application the ACA can automatically answer incoming calls. To prevent premature termination of any call due to line noise or data error, telephone connections are terminated only by software command.

In the leased line configuration, the ACA is shown in half duplex mode, using either point-to-point or multi-point centralised ASCII communication procedures.

When used in point-to-point communication, either station may initially have control (master) status, using software administration of contention problems.

In a multi-point centralised configuration, the ACA may be used either as a control (master) or tributary (slave) station. Data transfer is initiated by the control station which addresses associated tributary stations.

4.5.1 Physical description

The ACA consists of three printed circuit cards, designated AC1, AC2, and AC4. The optional dial-out card used with the ACA is designated CH7. AC1 and AC2 must occupy one input/output position in the Model 22 processor and are always placed as a pair. The partition designator number for the Asynchronous Communication Adaptor is determined by the position of the AC1 card.

4.5.1.1 *Operational characteristics*

ACA operation is fully automatic, without manual controls. The unit responds to normal processor machine instructions, and does not require any special operating procedures for the ACA itself.

Operating modes

The ACA has four general modes of operation: *start, idle, transmit* and *receive,* as shown in Figure 4.4. The modes are entered as follows:

1 *Start* When power is turned on, the ACA is in start mode, and remains in Start mode until the processor executes a Write Control command for initialisation. Any previously set operating parameters, such as bit rate (speed), odd or even parity, or ISF mode are cancelled and new parameters for current operation are established. Unless the ACA is properly initialised, all input/output instructions are terminated with FLAG status. Initialisation puts the ACA into Idle mode

2 *Idle* In Idle mode the ACA accepts all input/output instructions, keep track of the number of control characters to be transferred, and branches to Transmit or Receive mode in accordance with current instructions. If a RING indication is received from the modem while the ACA is in Idle mode, the ACA notifies the processor by setting a Service Request signal

3 *Transmit* In transmit mode, the ACA obtains characters one at a time, from the processor core memory, formats each character into a control or data character, adds the required Start, stop and parity bits, then serially shifts each character to the modem

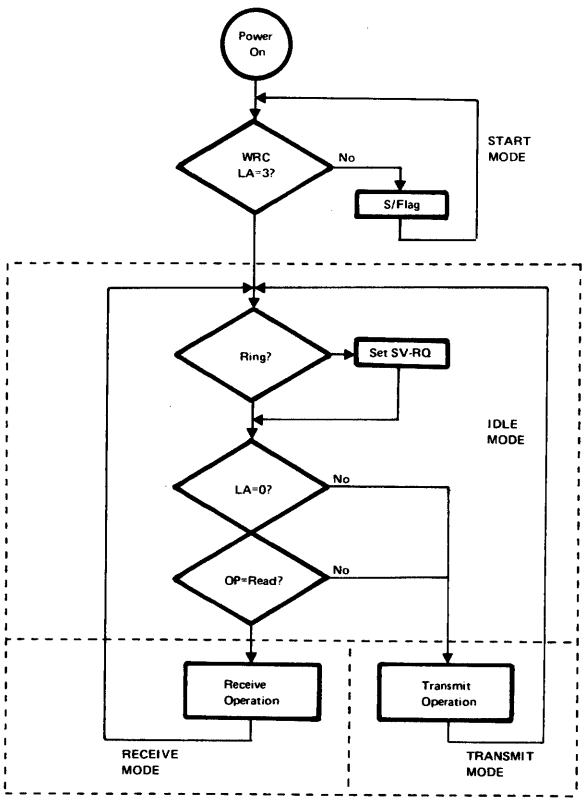


Figure 4.4 Simplified Diagram of Operating Modes

3 *Receive* In Receive mode, the ACA detects incoming start bits, samples the data bits at the selected speed, processes all control characters as required, and shifts the data bits to a data register. The ACA then transfers each data character to the storage buffer established in core memory. ACA remains in Receive mode until a terminating character is received, a malfunction is detected, or a 14-second timeout occurs. (In the event of a 14-second timeout, a new Read or Read Control command may be issued by the program to re-establish Receive mode)

4.5.2 Dial operations

When the ACA is used with modems in a switched line configuration, remote stations may be called by manual dialling, or by automatic dialling through a CH7 board and an Automatic Calling Unit.

4.5.2.1 *Manual dialling*

In manual dialling, the program initialises the ACA before the operator dials the desired number. The operator stops the program, dials the number, and upon hearing the answer tone, presses the DATA button on the modem. The operator then allows the program to continue. Under normal program control, the ACA is executing a Read or Read Control instruction before data is received.

4.5.2.2 *Automatic dialling*

Automatic dialling is initiated by a Write Control instruction with a device number (LA) of 1. The dialling digits are called from memory in the normal interrupt manner and the instruction is ended when the processor sets a LAST condition as the last digit is transferred through the CH7 card to the Automatic Calling Unit. Dialling can also be terminated by the Automatic Calling Unit because of some excessive delay in the dialling procedure. An Automatic Calling Unit timeout gives the ACA an Abandon Call and Retry (ACR) signal, resulting in ERROR status. The number is redialled by command of the software program (not an automatic procedure). ASCII communication standards require that the calling station transmit first, except when calling an ISF. Upon completing the dialling operation, therefore, the ACA should receive and commence execution of a Read or Write instruction. The Automatic Calling Unit terminates the Write Control dialling instruction and returns control to the program at the end of the answer tone. By returning control to the program at the end of the answer tone, the probability of telephone line noise and switching noise entering the ACA while it is in a Read instruction is greatly reduced. Because of normal operating time of the modem, there is enough time between the end of the answer tone and the readiness of the modems to handle data for the instruction to become active. Other timing options are available, however.

4.5.2.3 *Automatic dial options*

The Automatic Calling Unit may be required to transfer control to the program at the end of the dial number, or at the beginning of the answer tone.

End of number option

This option may be requested by the customer. In operation the following criteria apply:

1 The Write Control instruction supplies digits to the Automatic Calling Unit in which 'L' is the last digit. The dial number may or may not include the area code; examples are:

3535480L|XXXXXXXXXXXX
4153535480L|XXXXXXXXX

where X is any number

2 The WRITE CONTROL instruction is released just as the dialling is complete

Note: If the program next goes into a Read instruction, there is a great probability that noise will be received in the ACA.

Beginning of answer tone option

This option may be requested by the customer. In operation, the following criteria apply:

1 The dial number transferred to the Automatic Calling Unit during execution of the Write Control instruction must have the exact number of digits to be dialled; for example:

3535480 (no others) or
4153535480 (no others)

2 The Write Control instruction is released just as the answer tone is detected

Note: If the program next goes into a Read instruction, there is a probability that noise will be received by the ACA. Noise may be reduced by delaying the first Read instruction by 3½ seconds, by no greater time than 4½ seconds.

Telephone hang up function

The ACA does not automatically hang up when an instruction ends, either normally or in bad status. A separate software instruction (Write Control, LA=2) is required to tell the ACA to terminate a call, and disconnect.

4.5.3 Operation in ISF mode

4.5.3.1 *Introduction*

While most ACA functions are common to both Individual Store and Forward (ISF) and non-ISF operations, there are several special functions unique to the ISF mode; this mode is therefore described as a separate configuration. For all functions, the ACA must first be initialised, as described in section 4.5.3.10. When initialising in ISF mode, bit speed and parity must also be set.

4.5.3.2 *Timeout periods*

In the ISF mode, the ACA provides two timeout periods;

1 a 4½ character-time period of approximately 37½ milliseconds, and

2 a 14-second time period.

If during an active Read instruction, one or more characters are received and then 4½ character-times pass without any characters being received, the ACA times out, ending the Read instruction. If an ERROR condition exists, as for example no ETB, ETX, or EOT as the last character of the message, or less than eight characters received, a Transmission Status Character (TSC) is prepared and transferred to the Model 22 processor. After a 4½ character-time timeout, another Read instruction may be initiated by the program. The timer is reset each time a character is transferred to the processor memory.

The 4½ character-time timeout allows the ACA to recognise inter-record gaps in data transmission from the ISF. When an inter-record gap occurs in the ISF tape (250 ms. time period), the System Ten processor continues to service other partitions in the system in sequence. At the end of the inter-record gap, if the Service Request raised by the ACA has not been answered by the time the second character after the inter-record gap is received, the ACA drops the reverse channel to the ISF and sets ERROR status. When the reverse channel is dropped, the ISF rewinds to the start of the current record and waits.

The 14-second time period allows the ACA to remain in a receive mode during the time required for the ISF to start transmitting data, which may require up to 32 seconds. After each successive timeout, another Read instruction is initiated by the program a planned number of times (generally five or more). If no data is received during the repeated Read instructions, the program terminates the connection with a disconnect instruction (Write Control, LA-2).

4.5.3.3 *Manual dialling in ISF mode*

In the ISF mode, manual dialling starts by initialising the ACA:

1 Program initialises the ACA and stops (initialisation turns OFF the reverse (back) channel)

2 Operator dials the number and upon hearing the answer tone, presses the DATA button on the modem, allowing the program to resume

3 The program issues a Read instruction which turns on the reverse (back) channel

Note: It is important that the Read instruction is issued before the ISF has a chance to time out (800 milliseconds).

4.5.3.4 *Automatic dialling in ISF mode*

When dialling is used in ISF mode, the following sequence of events occurs:

1 Program initialises the ACA (Write Control, LA=3)

2 The program issues a dial instruction (Write Control, LA=1)

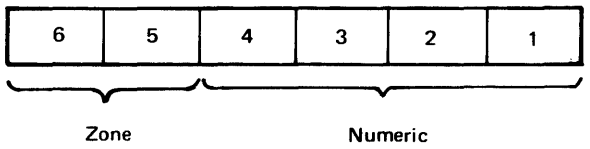
3 Upon successful completion of the dialling operation, the program issues a Read instruction which turns on the reverse (back) channel

4 The ACA is executing a Read instruction before a Start of Heading (SOH) is transmitted by the ISF

When the ISF modem is called, either by the Automatic Calling Unit or by manual dialling, and control is returned to the program, the ACA should immediately start executing a valid Read instruction. A valid Read instruction in the ISF mode is one in which the LA field is zero. When the Read instruction becomes active, the ACA turns on the Supervisory (reverse) channel, required for ISF operation

4.5.3.5 *Transmission status character*

Any error occurring while the ACA is in the ISF mode causes the ACA to generate a Transmission Status Character (TSC). This is an extra six-bit character transferred to the processor memory immediately following, and as part of, the message in error. The TSC describes the error conditions encountered while receiving the data. The meaning of each bit in the TSC character is described in Table 4.1. The meaning of various bit combinations explained in Table 4.2.



Transmission status character

Table 4.1
Interpretation of the transmission status character

Bit 1 Message Indicator (normally ON)

0 A legitimate message has not arrived, but a few characters, (less than eight) have been received.
(This condition is true provided that Bit 6 is OFF)

1 A message of proper length (eight or more characters) has been received.

Bit 2 Data Set Error Indicator (normally OFF)

0 The data set (modem) was operating properly during the message.

1 The Carrier On (CO) or Data Set Ready (DSR) lines dropped. indicating a data set error. (Noisy lines upon dial-up will cause this condition, and may be normal.)

Bit 3 Ending Indicator (normally ON)

0 A message was received but no ending character (ETB, ETX, or EOT) was included at the end of the message.

1 The ending character was present.

Bit 4 Parity Error Indicator (normally OFF)

0 The data had good parity, good START bits and good STOP bits.

1 One or more characters in the data stream had any or all of the following errors:

Parity Error
Wrong START Bit
Wrong STOP Bit

Bit 5 Character Overrun Indicator (normally OFF)

0 All characters received were transferred to the processor.

1 One or more characters were lost because the processor interrupt time was too slow. (This error is extremely serious.)

Bit 6 Record Overrun (Lockout) Indicator (normally off)

0 The programmer's Read instruction preceded data arrival.

1 The programmer's Read instruction arrived late, resulting in loss of data. Automatic retry is executed provided the data consists of eight or more characters, and provided also that the instruction is a Read, not a Read Control.

Note: Bit 6 of the TSC is the dominant bit whenever it is ON(1), indicating a data lockout condition. When Bit 6 is 1 (ON), the condition of the other bits in the character is undefined.

The individual bit indicators in the transmission status character are not mutually exclusive. However, there is a logical precedence that dictates the order of testing the individual bits.

Transmission Status Character (TSC) bit (6), the message overrun indicator, pre-empts all other indicators and, therefore, has the highest testing precedence. TSC(1), the short message indicator, is next in precedence. TSC(2, 3, 4, and 5) have no logical precedence, therefore should be tested in order of probability of occurrence, TSC(4) parity, TSC(5) character overrun, TSC(3) no ending, and finally TSC (2) data set error.

Table 4.2
Error combinations

Bit considered	*Combinations 654321*	*Automatic reverse channel*	*Probable cause comments*
1	0XXXXQ	No	Noise, or good SOH
Message indicator	1OOOOQ	Uncertain–however All legitimate messages will be retried	Program timing Random noise System timing ISF leader variations Overloaded system
2	0XXX10	No	Noise, or bad SOH
Data set error	0XXX11	Yes	Bad message
3	0XX1X0	No	ACA will never generate this combination
	0XX0X0	No	Noise, or good SOH
Ending indicator	0XX0X1	Yes	Missing ending character
4	0X1XX0	No	Noise, or bad SOH
Parity error	0X1XX1	Yes	Bad message
5	01XXX0	No	Noise, or SOH, but system is overloaded
Character overrun	01XXX1	Yes	System overloaded, notify field systems engineer
6	100000	Uncertain–however All legitimate messages will be retried	Program timing Random noise System timing ISF leader variations Overloaded system
Record overrun (lockout)	1XXXXX	(Same)	Same as above, ignore X bits

4.5.3.6 *Reverse (back) channel communication*

The reverse (back) channel provides a means of simultaneous communication from the ACA to the ISF, to facilitate certain forms of error correction (retransmission) functions. Signals are sent from the ACA to the modem as dc voltage levels, transported over the telephone lines as a 387 cps tone, and applied to the ISF as restored dc voltage levels.

The Supervisory voltage signal is a separate signal applied to the modem, similar to other control signals such as Data Set Ready (DSR) or Data Terminal Ready (DTR) signals. The modem used with the ISF must be able to receive reverse channel signals, but is not required to send any signals.

4.5.3.7 *Receiving data from ISF*

If the ISF activity counter is at one when the unit is called, indicating that it has data to be transmitted, the ISF responds to a call with a hardware-generated Start Of Heading (SOH) character. The ISF then starts the tape and performs various internal functions that may require four or five seconds, but can be as long as 32 seconds, before transmitting recorded data.

If the ISF activity counter is at zero, indicating that the ISF has no data to be transmitted, a string of EOT signals are transmitted to the ACA. (The EOT signals are remapped into Ds by the ACA.)

Start of Heading (SOH) signal

The Start of Heading (SOH) signal indicates that a proper connection has been made with the ISF and that data is available. Absence of the SOH could indicate a noisy telephone line, a malfunction in the ISF, or a malfunction in the ACA. An SOH signal, converted into an A by the ACA, results in an error condition because only one character is received; 4½ character-times later (approximately 37½ milliseconds), the ACA times out, ending the Read instruction with an ERROR status. Bit 1 of the TSC is off, indicating a short message (less than eight characters). Because only one character was received, automatic retry logic is not activated. The telephone connection and ACA initialisation are not lost when the timeout occurs, and another Read instruction should be immediately initiated by the program.

With no data coming in while the ISF prepares to transmit, and while the tape leader is passing over the ISF read head, another timeout will occur after each 14-second time period. The read instruction is re-established by the program after each timeout, for a system-planned number of repetitions (at least five or more). At some time between reception of SOH and the last programmed Read instruction repetition, the data should be received.

EOT signals

Multiple EOT signals received when an ISF is called indicate that the ISF has no data to send (activity counter is at zero). The EOT signals result in a FLAG status (code condition 3) corresponding to the received data exceeding the memory capacity specified in the READ instruction. The program should verify that at least four of the first five characters received are EOTs, then issue a disconnect instruction (Write Control, LA=2) and terminate data handling with that ISF.

Multiple EOT signals received after data has been received from an ISF indicate that all the data recorded on the ISF tape has been transmitted (activity counter has returned to a count of zero).

Receiving data

Data from the ISF is received in blocks, separated by inter-record gaps (302 milliseconds) or by large gaps (nine seconds) corresponding to end of day leaders and the tape splice. The normal size of the data storage area for ISF data blocks, as specified by the B field in a Read instruction, should be about 250 characters since the maximum ISF data block has 226 characters. Because the ISF is a unique device with a specialised format, Read instructions in the ISF mode are not terminated by an ASCII ending code, but by either 4½ character-time timeout following the end of a data block, or a 14-second timeout occurring when no data is received, or by multiple EOTs filling the data storage area. If EOT, ETB, or ETX is not received as the last character in a message, however, the fact is recorded as an error and reported by a zero as Bit 3 of the Transmission Status Character.

When called, an ISF with data transmits an SOH, then starts the tape. During the time required to start the tape and pass the first leader over the ISF read head the ACA may time out several times. After each time out, the ACA received another Read instruction from the program. While the ISF is transmitting the ACA times out at each inter-record gap and each end-of-day leader; after each timeout another Read instruction is initiated by the program. Upon completing transmission of all its data, the ISF sends a string of EOTs which fill the designated data storage area in the processor, thereby terminating the last Read instruction.

Data from multiple ISF configurations

Multiple ISF units can be connected in series to transmit data through one modem. In this configuration, the ISF connected directly to the modem is called the master and the other ISFs are called slaves. During data transmission, the master transmits all of its data until its activity counter is zero, then transfers control to the first slave in the series. If the activity counter in the master is zero when called by the remote computer, control is passed immediately to the first slave. EOTs are not transmitted in this case. Each slave transmits its data until its activity counter reaches zero, then passes control to the next slave in the series. At the end of the entire transmission the last ISF sends the EOTs to the remote computer. Each ISF with data transmits an SOH signal, then starts the tape and locates its data blocks. There is the usual four or five seconds of delay (which can extend to more than 32 seconds) during which the ACA, having received data from the previous unit, times out after 4½ character-times without data. When the ACA times out, it receives another Read instruction from the program and is ready to accept data from the ISF that is just getting started.

4.5.3.8 *Automatic error correction*

An error occurring during an ISF Read operation causes the voltage level of the Supervisory lead to be lowered for 300 milliseconds at the end of the instruction, thereby turning the reverse channel off for that period of time. This instructs the ISF to rewind the tape to the last inter-record gap and repeat the data message from that point. Automatic operation of the Supervisory lead suspends execution of another ISF Read instruction for the duration of the 300 millisecond period. The ACA drops the voltage to the reverse channel whenever either of the following conditions occur:

1 A transmission greater than eight start bits is received after the program Read instruction became active, but an ERROR status (of any nature) exists when the Read instruction ends

2 A transmission greater than eight start bits is received but the transmission started before the program Read instruction became active (Record Overrun)

The first condition (number 1 above) causing automatic error correction procedures in the ACA may be due to a lack of a START or STOP bit in a character, or a parity bit error, or reception of the end of a message without an ending character (ETB, ETX or EOT).

The second condition (number 2 above) is termed a Record Overrun, cuasing bit 6 of the TSC to be set to 1. This error can occur in the following manner. In ISF mode, the ACA continually monitors the modern Receive Data (RD) line, even without an active Read instruction. If data is received before a Read instruction, the ACA stores the first bit, and goes into a lockout condition until the transmission is completed. The first character received is then passed to the processor along with a TSC indicating the malfunction. The Read instruction is terminated in an ERROR status and the automatic reverse channel signal is sent to the ISF to retransmit the message. A Record Overrun is quite common, and could be due to any of the following normal events:

1 Random noise received by the ACA before the programmed Read instruction becomes active

2 The ACA times out between ISF leaders, just before the ISF continues sending data.

3 Noisy lines after a dial-up operation.

4 Unforeseen program delays, for example, printer top of form delay, or partition switching delays.

When an error status occurs during execution of a Read operation in the ISF mode, the automatic reverse channel signal is sent to the ISF. The ISF rewinds the tape to the last inter-record gap and repeats the portion of the message containing the error. If the error is repeated, the whole cycle of rewind and retransmit is repeated. After a programmed number of repetitions, it is assumed that this is an error actually recorded on the tape and not something due to transmission noise. Such errors are termed *hard* errors. To avoid further repetitions and to save the data, the software issues a Read Control, LA=0 instruction. This instruction is the same as a Read instruction except that the automatic reverse channel signal is inhibited. The Read Control instruction is used to accept messages which contain hard errors. The data message is accepted and stored in the processor memory where future processing may eliminate the error. Although the automatic reverse channel signal is inhibited the Transmission Status Character is sent to the processor, as in a Read operation.

Parity error indications

The ISF operates with even parity for each character. If a parity error is detected by the ACA, the character in error is transferred to the processor as ^ (circumflex), and a Transmission Status Character is generated to indicate the parity error. The ERROR status causes an automatic reverse channel signal and the transmission is repeated.

4.5.3.9 *Communicating with the ACA*

The ACA is operated with standard System Ten computer machine instructions: Read, Read Control, Write, and Write Control.

4.5.3.10 *Initialisation*

After any power-down power-up sequence, power failure, or after a system or input/output reset, the ACA requires initialising to set the parity (odd or even), bit speed, and operating mode (ISF or non-ISF). Initialisation must precede any other instruction, and should also be repeated before every telephone dialling instruction. Parameters set by initialisation remain in force throughout all following instructions until a new initialisation is made, or until initialisation is lost because of unusual line noise, power failure, or reset.

Initialisation consists of a Write Control instruction (Write Control, LA=3) followed by an initialising character taken from memory. The initialising character, as shown in Figure 4.5 sets the bit rate, parity, and operating mode.

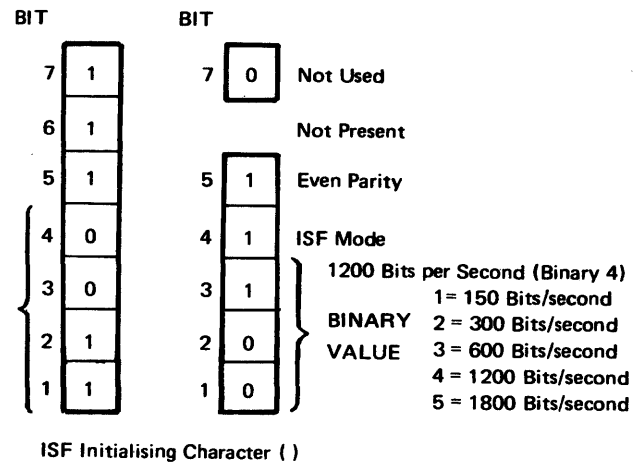


Figure 4.5 Initialisation Character.

Note: When initialising for ISF operation (receiving) the bit rate and parity must be set, as well as the ISF mode (Figure 4.5). Proper initialising for ISF operation is a Write Control, LA=3 instruction followed by a < (less than) symbol.

Since the System Ten computer may have just powered up when initialisation is first attempted it is possible to get a FAULT condition on the first attempt. It is good practice, therefore, to repeat initialisation twice. If an instruction other than Write Control, LA=3 is given to the ACA before proper initialisation, the instruction will be terminated and FLAG status (condition code) will be posted.

4.5.3.11 *ACA instructions*

Any System Ten computer input/output instruction can be executed by the ACA following proper initialisation, provided the parameters of the instruction are compatible with the bit rate, parity, and ISF, non-ISF mode specified by the initialisation instruction. The binary value of the device number (LA field) used in Read, Read Control, and Write instructions is interpreted by the ACA as the number of control characters to be transmitted to the remote device. (In ISF mode, the LA field of Read and Read Control instructions must be zero). The ACA activates an OUT REQuest line to the processor which forces a temporary Write condition in the processor logic, thereby enabling the ACA to retrieve the required control characters from memory. The ACA converts each character into control character format, adds the START and STOP bits, generates the required odd even parity, and transmits each control character to the modem. Transmission is in accordance with ASCII rules of communication. When the specified number of control characters have been transmitted, the OUT REQuest line is returned to normal and the rest of the instruction is executed in the usual way.

Read and Read Control instruction

Read and Read Control instructions executed by the ACA cause it to transmit any required control characters, then start accepting input characters from the modem, moving data from a remote device to sequential locations in the processor memory.

Read and Read Control operations are normally terminated by receipt of a terminator control character. If an ACA Read operation is terminated by exhausting the count (B field) of the Read instruction, then the Error Condition Indicator (condition code 1) in the central processor is set ON. Any further character transfers attempted by the ACA result in FLAG status (condition code 3). FLAG status disables all other status indications and, when it occurs, will be the only status indication posted, even though other errors may be detected.

When a block of data has been received, a Block Check Character (BCC) is generated in the ACA. BCC is a longitudinal redundancy check character which is compared with the received redundancy check character. If the comparison shows that no error exists, the BCC is stored with the data in memory.

When executing a Read or Read Control instruction, the program maps out a section of memory to include storage for any control characters required, storage for the data to be received, and storage for the BCC. To avoid exhausting the count of the Read instruction, every ACA Read instruction should normally specify storage for more data characters than are actually expected, as shown in Figure 4.6, a simplified diagram of a Read instruction storage area. In the diagram a message of five characters, including starting and ending characters, is expected.

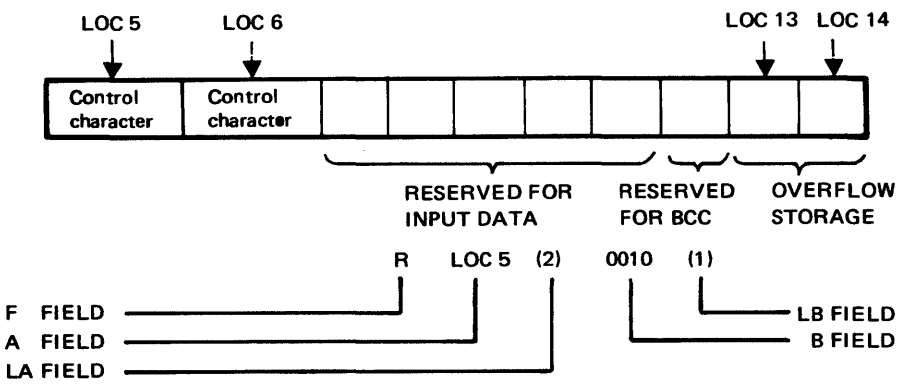


Figure 4.6 Simplified Diagram of Read Instruction Storage Area

Note: LB = 1 indicates a normal Read instruction, not a Read Control.

In the figure, the beginning address of the storage area (LOC 5) is indicated by the A field of the instruction, and the length of the storage area (10 characters) is specified in the B field. This storage area is used in the following manner:

1 Two control characters are stored in LOC5 and LOC6 by the program. These control characters are retrieved and transmitted by the ACA at the start of the Read operation

2 One storage space is reserved for the BCC

3 Although only five characters are expected in the message, a maximum of seven data characters can be read by the instruction:

(a) if six characters are received, the BCC will be placed in LOC 13

(b) if seven characters are received, the BCC will be placed in LOC 14

(c) if more than seven characters are received, the eighth character will be placed in LOC 14. The instruction will be terminated and the error condition Indicator (condition code 1) will be set ON

Normal Read and Read Control instructions are ended by receiving one of the ASCII ending codes such as ETB, ETX, ENQ, ACK, NAK, EOT, or DLE with one character (0,1,?,;) following. A Read instruction which is terminated without one of these ending characters will cause FLAG status. (A Read instruction is terminated without an ending character when received data exceeds the storage area established by the program).

A Read or Read Control instruction will remain active in the receive mode for about ten seconds without data. After 14 seconds without receiving data, the ACA times out and posts FAULT status (condition code 4). The instruction is ended. While receiving data, the 14-second timer is restarted each time a character is transferred to memory.

The fields of an ACA Read instruction are as follows:

Field	*Meaning*
F	0000
LA	0 to 9 (Number of leading control characters). This field must be zero in ISF mode.
LB	1 or 5
A	memory address of start of input data storage area
B	maximum number of characters to be transferred, including initial control characters. B must be at least one greater than the combined count of LA plus expected number of received characters in order to obtain good status at the end of the instruction.
AC	designator that A address is in the common area
BC	not used
IA, IB	index control for A and B

The condition codes for the ACA Read instruction are interpreted as follows:

Code	*Meaning*
ERROR (condition code 1)	no start bit detected; no stop bit detected; parity error; longitudinal redundancy check error; instruction terminated by Record Overrun.
FLAG (condition code 3)	ACA not initialised; instruction terminated by LAST condition
FAULT (condition code 4)	Instruction terminated by ACA time out.

The Read Control instruction is used in ISF mode to accept a message containing an uncorrectable error (see section 4.5.3). In non-ISF mode, a Read Control instruction is used to identify data received from slave stations in a multi-point centralised private line configuration. For this function, the number of control characters specified by LA are transferred from memory to the ACA, but the first character, called the station identifier code, is stored in the ACA while the remaining (LA-1) characters are transmitted to the local modem. Following this initial transmission, the ACA goes into a special receive mode in which it must match the first character received with the station identifier code stored in the ACA before it will accept further data from the modem. The operation continues until a terminating character is received or until a timeout occurs.

The fields of an ACA Read Control instruction are interpreted as follows:

Field	*Meaning*
F	0000
LA	0 This field must be zero in ISF mode
LA	1 to 9 (Number of leading control characters, including a station identifier code to be stored in the ACA)
LB	3
A	memory address of start of input data storage area
B	maximum number of characters involved in the instruction (local station identifier code, transmitted characters, and received characters).
AC	designator for A address in common area
BC	not used
IA, IB	index control of A and B

The condition codes for the ACA Read Control instruction are interpreted as follows:

Code	*Meaning*
ERROR (condition code 1)	no start bit detected; no stop bit detected; parity error; longitudinal redundancy check error; instruction terminated by Record Overrun.
FLAG (condition code 3)	ACA not initialised or operation terminated by setting LAST condition
FAULT (condition code 4)	Instruction terminated by ACA time out

Write and Write Control instructions

A Write instruction causes the ACA to transmit characters until the instruction is terminated by the System Ten processor. The ACA interprets the LA field of the instruction as specifying the number of initial characters that are to be transmitted as control characters.

The fields of an ACA Write instruction are interpreted as follows:

Field	*Meaning*
F	0001
LA	0 to 9 (Number of control characters to be transmitted)
LB	1
A	memory address of start of record to be transmitted
B	number of characters to be transmitted
AC	designator for A address in common area
BC	not used
IA, IB	index control for A and B

The condition codes for the ACA Write instruction are interpreted as follows:

Code	*Meaning*
ERROR (condition code 1)	Not used
FLAG (condition code 3)	ACA not initialised
FAULT (condition code 4)	Time out due to modem not functioning or power failure

While executing a Write instruction, the ACA generates odd or even parity (software selected) for each character, and develops a longitudinal redundancy check (BCC) character which is transmitted at the end of the message block.

A Write Control instruction is used to initialise the ACA. The instruction is followed by a character from memory which establishes bit rate, odd or even parity, and normal or ISF mode. Write Control instructions are also used to establish a DIAL OUT operation and a HANG UP operation.

The fields of an ACA Write Control instruction are interpreted as follows:

Field	*Meaning*
F	0001
LA	1 DIAL OUT sequence. Data characters in the instruction are the dial digits
LA	2 HANG UP sequence
LA	3 Required instruction to initialise the ACA before any other operation can take place
LA	4 to 9 not used
LB	3
A	memory address of record (used for dial out digits and for initialisation character).
B	number of characters in record. Always equal to 1 for initialisation
AC	designated for address in common area
IA, IB	index control for A and B

The condition codes for the ACA Write Control instruction are interpreted as follows:

Code	*Meaning*
ERROR	Set by Abandon Call and Retry (ACR) signal during dial out operation
FLAG	Set if ACA has not been initialised, or if an incoming call occurs during a dial out operation
FAULT	Timeout during a dial out operation caused by an inoperative modem or inoperative Automatic Calling Unit.

4.6 The Asynchronous Terminal Adaptor

4.6.1 General description

The Asynchronous Terminal Adaptor (ATA) provides an interface between the Model 22 processor and one operator-oriented communication terminal using asynchronous data transmission mode.

The nominal data transmission rate is 110, 150, 200 or 300 bits per second, and the full ASCII seven-bit character set can be handled. The character length is 11 bits at 110 bps and 10 bits at all other speeds, and the character format is: one START bit, seven data bits, one EVEN PARITY bit and one or two (depending on character length) STOP bits.

An ATA and a terminal can be directly connected if the distance between them does not exceed 50 feet, as shown in Figure 4.7. Remote installations require a modulator/demodulator (modem) at both the remote and the ATA end of the line. Using modems, as shown in Figure 4.7, the ATA can be used with leased lines or dial-switched lines. It is significant to note, however, that the ATA can accommodate only one terminal at a time, and is not recommended for use in multipoint configurations. Such systems should be implemented only with extreme care; application system design must include several operational procedures and unique program specifications which are described in section 4.6.5.

Using an Automatic Dial option (CH7 printed circuit card) and an automatic calling unit the ATA can be used for automatic calling. All ATA communications are in half-duplex mode, regardless of the capabilities of the attached modems and terminal.

4.6.2 Physical description

The basic ATA consists of two printed circuit cards (TA1 and TA2) which may be installed in any IOC position of a Model 22 processor. If the Automatic Dial option printed circuit card (CH7) is used, it must be installed in an odd channel slot of an adjacent input/output position in the

processor. The CH7 card may be on either side of the basic ATA, but it must occupy the first higher or lower position of the adjacent input/output channel. The ATA partition number is governed by the position of the TA1 card and is not affected by the presence or absence of an Automatic Dial option card. The ATA partition is like any other processor partition. It receives and releases control of the ACU by the same rules as any other partition, and the program which resides in it may contain any valid ATA IOC instructions.

ATA circuit cards are fitted with five manually-set jumpers used to specify the partition size allocated to the ATA and to inhibit or enable ATA partition access to the Privileged area of Common memory. These jumpers are set at the time of installation.

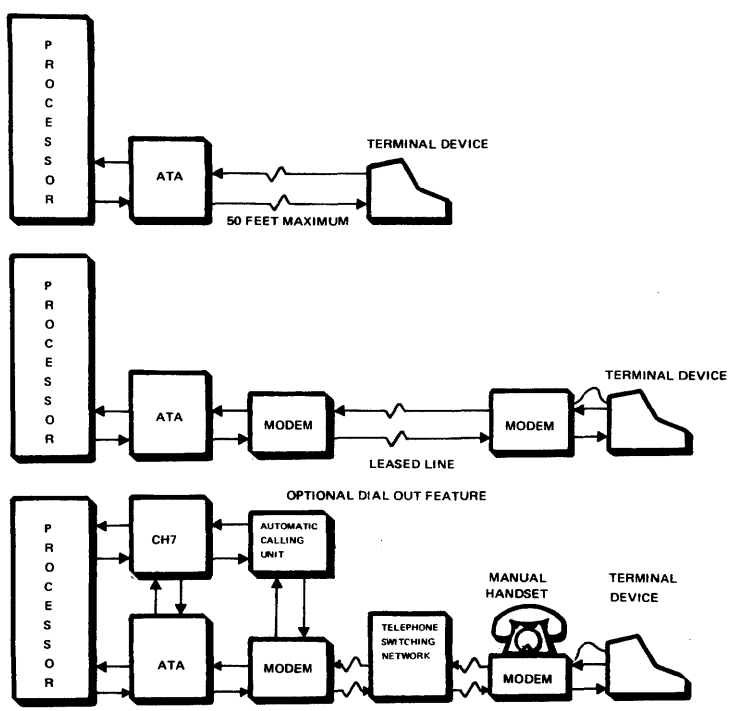


Figure 4.7 ATA System Application

There are five DPDT (Double-Pole, Double-Throw) switches mounted on the ATA circuit card. Four of the switches are used to set various optional functions: CONTROL CHARACTER INPUT (S1), BIT RATE (S2 and S3), and WRITE TIMEOUT INTERVAL (S4). Switch S5 is a manual reset switch used during maintenance.

4.6.3 Functional description

4.6.3.1 Selection switches

The ATA circuit cards include the five switches described in Table 4.3.

Switch name	*Switch*	*State*	*Function*
Control character input	S1	OFF	Characters in columns 0 and 1 of the ASCII chart (the control characters) are not transferred to memory during read instructions.
		ON	Control characters are transferred to memory during read instructions. Control characters will not be decoded to perform internal functions in this case.
Bit rate	S2, S3	S2 S3 OFF OFF OFF ON ON OFF ON ON	 Bit rate = 110 bps Bit rate = 150 bps Bit rate = 300 bps Bit rate = 200 bps
Write timeout interval	S4	OFF	Write timeout interval = 1.8 sec minimum
		ON	Write timeout interval = 1.0 sec minimum
Manual reset (momentary)	S5	OFF	ATA operates normally
		ON	ATA is initialised to an idle state

4.6.3.2 *Control character input*

Switch S1, which affects control character input, is used only for diagnostic routines. The switch is normally off. Control characters (characters with bit 6 and bit 7 = 0) are decoded and appropriate action taken, but the characters are not entered in memory. This feature allows data to be received that contains form feed, line space and other mechanical action control characters which are omitted from the record stored in memory.

When Switch S1 is on (for diagnostic routines only), control characters are read into memory. In this mode, control characters do not cause any special action within the ATA such as terminating a Read operation.

Independently of Switch S1, the four characters occurring in column 7, rows 12, 13, 14 and 15 of the ASCII code chart are never transferred to memory. This feature is included so that DEL codes can be automatically rejected from the paper tape data stream during input.

Bit rate

The ATA has a selectable bit rate of 110, 150, 200, or 300 bits per second (10, 15, 20, or 30 characters per second). The bit rate is selected by the position of Switches S2 and S3, as described in Table 4.3.

Write timeout interval

With Switch S4 in the off position, a 1.8 second delay is provided after transfer of print control characters (HT, LF, BS, CR, and FF) to inhibit further transmissions until mechanical actions are complete.

When Switch S4 is on, the time delay is 1.0 seconds. This is greater than the maximum time required when operating with a teletypewriter.

Clock generator

All ATA internal timing is generated by counting down the IOC High Frequency clock (450 KHz) and the IOC Medium Frequency clock (300 KHz) signals from the ACU. Switches S2 and S3 select the bit rate from the four frequencies available.

Character generator

Some ASCII characters are generated by the ATA to control the associated terminal. This is necessary because each particular device in the terminal (keyboard printer, paper tape reader, and paper tape punch) may be individually selected. It is also necessary to provide cues to an operator to inform him of the state of the terminal. The following is a list of the characters automatically generated and transmitted to the terminal by the ATA:

Character	*Meaning*
S1	Keyboard printer on
S0	Keyboard printer off
DC1	Paper tape reader on
DC2	Paper tape punch on

A READER-OFF code (DC3) is not generated since most paper tape readers respond to a STOP code only if it is punched in the paper tape. This restriction is a result of the fact that communication between the ATA and the terminal is half-duplex.

Character	*Meaning*
DC4	Paper tape punch off
E	Enter data cue for operator
L	Load cue for operator
(	Off-line cue for operator
)	On-line cue for operator

Character detector

The Character Detector circuit in the ATA determines if any critical control characters are received. When receiving data, critical characters include US, EM, ESC, SUB and CAN. In transmit operations, eight characters are detected: HT, BS, CR, FF, VT, LF, S0, and S1.

Operation codes

The ATA contains a register which stores operation codes or instruction information which it receives from the processor ACU at the start of an input/output operation. The following is a list of the operation codes that are interpreted by the ATA:

1 READ FROM KEYBOARD (RDKB) The ATA sends PRINTER-ON(S1), sends operator cue E, reads data from keyboard, sends PRINTER-OFF (S0)

2 READ FROM READER (RDR) The ATA sends READER-ON(DC1) and reads data from the paper tape. Assuming the terminal has an automatic tape read feature, the operator must have the tape properly mounted before the instruction starts. The tape reader stops only if a READER-OFF code (DC3) is present in the tape or if the operator presses the STOP READER key

3 READ CONTROL FROM KEYBOARD (RECKB) The ATA sends PRINTER-ON (S1), sends operator cue L, reads a 10-character program-loading instruction (bootstrap) from keyboard, sends PRINTER-OFF (S0)

4 WRITE TO KEYBOARD PRINTER (WRKB) The ATA sends PRINTER-ON (S1) transmits data to keyboard, sends PRINTER-OFF (S0)

5 WRITE TO PUNCH (WRP) The ATA sends PUNCH-ON (DC2), transmits data to the paper tape punch, sends PUNCH-OFF (DC4)

6 WRITE CONTROL TO KEYBOARD PRINTER (WRCKB) The ATA sends PRINTER-ON (S1), transmits control characters from the ACU to the printer, sends PRINTER-OFF (S0)

7 WRITE CONTROL TO PUNCH (WRCP) The ATA sends PUNCH-ON (DC2), transmits control characters from the ACU to the paper tape punch, sends PUNCH-OFF (DC4)

8 WRITE CONTROL HANG-UP (WRCH) The ATA executes hang-up sequence

In addition, a WRITE CONTROL DIAL operation is interpreted if a CH7 Automatic Dialling option card is present. During operation, dial digits are transmitted from the processor to the Automatic Calling Unit through the CH7 card.

4.6.4 Operation

There are no manual controls or indicators accessible to the operator. The ATA channel operates only under software control. Operation of the various terminals that may interface with the ATA varies considerably. General operation characteristics and procedures required to operate with a Model 22 processor are given below.

4.6.4.1 *Break*

The ATA logic operation allows the terminal operator to temporarily break from the computer data exchange to a stand-alone local condition. In the local mode the terminal can be used without affecting the processor or the software program. For example, a Model 7102 Communication Terminal can be used in the local mode to punch a paper tape which contains data or a software program for use with another system or at another time. The operator is in full control of the local/on-line status of the terminal and can change from one to the other at any time by depressing the BREAK key for a short time (two character-times or longer). At each transition from local to on-line or from on-line to local, an operator cue is generated by the ATA and printed by the keyboard printer. A (indicates that the terminal is in local mode; a) means the terminal is on-line. Any software instructions executed in the ATA partition while the terminal is off-line terminate with FAULT status (Condition Code 4).

4.6.4.2 *Service Request*

Service Request (SVR) is used by the operator to alert the ATA program of the operator's desire to do something. In most software programs the operator has the option of branching to a new place in the program or back to the beginning of the program by setting a Service Request condition. Service Request is set in the ATA by first placing the terminal in local mode, then entering a SUB code as the first character in local. The terminal can be immediately returned to the on-line mode, at the operator's option. The operator cannot cancel or reset the Service Request once it is set. The first operation handled by the ATA program after returning on-line should be the Service Request set while in the local mode.

4.6.4.3 *Load Request*

A Load Request is used by the operator to initiate loading a program into the processor. The terminal is put in local mode and an ESC code entered as the first character in local. When the terminal is returned on-line, the ATA immediately acknowledges the Load Request by generating and transmitting an L code to the keyboard printer. The L cue indicates to the operator that a 10-character load instruction or bootstrap is required.

The operator will normally load a Read instruction. For example: Read 20 characters from device 0 on the ATA, starting at location 0010. In this example, the next 20 characters read by the ATA could be two instructions:

1 Read 5K characters from peripheral 2 (I/O), starting at 0100

2 Branch to 0100

The first instruction actually loads the program and the second allows execution to begin.

The operator may respond to the L cue by loading ten zeros. The hardware interprets the instruction 0000000000 to mean: Read from the disc (100 characters) into location 0000. The disc address is at 0000.

The disc address is then: Drive 0, Surface 0, Cylinder 000, Sector 00.

If a Unit Separator (US) or End of Media (EM) code is given to the ATA as a first character after L is printed for Load Request, the ATA will attempt to fill the field with blanks, which the ACU will convert to zeros, because the Load Request is set.

Load Request is automatically set by the processor if a program instruction error is found. The L cue is printed in the same manner as an operator-initiated load request, and a bootstrap load instruction must be loaded in the same manner, by the operator. A Load Request automatically terminates any program executing in the ATA partition.

4.6.4.4 *Repeat function*

REPEAT is a retry of a Read or Read Control instruction. The REPEAT function is set by keying in a cancel (CAN) code after the instruction has been started. The operator can force any RDR instruction to repeat by keying in a CAN code after first stopping the paper tape reader; a CAN code in the paper tape has a similar effect. REPEAT does not affect the processor or the

program, which only recognises that the Read or Read Control instruction is still active in the ATA.

4.6.5 Programming information

4.6.5.1 *Introduction*

The ATA is a special type of input/output channel which allows only one external device (communication terminal) to be connected to it, either directly or through a modem. Input and output (receiving and transmitting), however, are carried out in the same way as for any other input/output channel. The only significant difference between a standard Multi-Terminal Input/Output Channel (MTIOC) and an ATA is the fact that a standard MTIOC uses a device number (LA field of the instruction) to select one of ten units that might be connected to the IOC, and the ATA has only three devices contained in a single communication terminal (paper tape punch, paper tape reader, and keyboard/keyboard printer). The ATA can therefore interpret the device number code to generate additional operating instructions, for example, DIAL and WRITE CONTROL HANG-UP (WRCH).

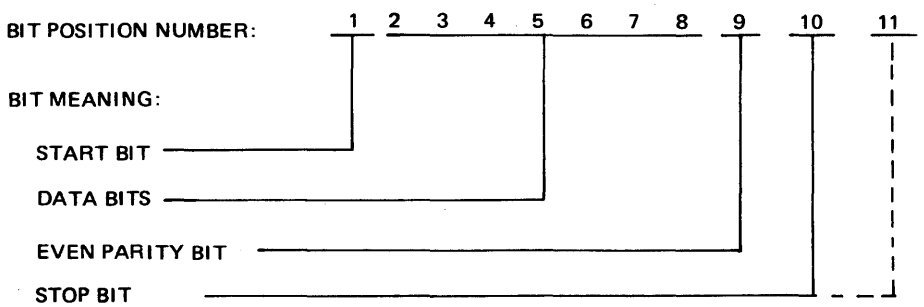
The ATA is operated with standard System Ten processor machine instructions: Read, Read Control, Write, and Write Control. Instructions are received in a register in the ATA and decoded to generate operating instructions. If an improper instruction (for example, an instruction with the wrong device number) is received, the instruction is merely discarded; neither LOAD REQUEST nor a condition code is set by an improper instruction. The program, recognising that the ATA is not busy, executes the next instruction.

4.6.5.2 *Status indications*

The ATA reports status at the close of each instruction. Although not available to the operator, the status indications are interpreted by the processor to show unusual conditions that developed, if any, during the term of the instruction. Table 4.4 shows the logic conditions that are reported by the status indicators in the ATA and the resultant condition codes.

4.6.5.3 *Communication format*

Communications between the ATA and the terminal are conducted in half-duplex mode, using 10-bit serial ASCII Asynchronous Code format. The character format, shown below, is a START bit, seven data bits, EVEN PARITY bit, and STOP bit. When operating in the 10-characters-per-second mode, an eleventh bit, also a STOP bit, is included in each character:



Code conversion

The input/output devices that can be connected to the ATA are compatible with the 7-bit ASCII character set. The Model 22 processor, however, uses the 6-bit character set occurring in columns 2, 3, 4, and 5 of the ASCII code chart in Figure 2.2. The difference in character length requires the processor to perform an output code conversion that changes an alphabetic character into its corresponding control character (bits 6 and 7 both = 0) during WRITE CONTROL KEYBOARD and WRITE CONTROL PUNCH instructions. An input conversion is performed by the ATA to ensure compatibility with the output conversion. In the input conversion, the lower case alphabetic codes (columns 6 and 7) and the control character codes (columns 0 and 1) of the ASCII code chart are converted into equivalent upper case alphabetic codes (columns 4 and 5) of the ASCII chart. Characters in columns 2, 3, 4, and 5 do not require conversion. This means, for example, the "DC1", "q", and "Q" are all converted to the System Ten processor "Q" equivalent, and an ASCII "1" is the only character which converts to a System Ten processor "1" equivalent.

Table 4.4
Status Indicators

Status	*Instruction*	*Cause*
ERROR (Condition Code 1)	Read	Receiving a parity error, no START bit, no STOP bit, or a data overwrite.
	Read Control	Same as Read.
	Write	Not used.
	Write Control	Set by the ACR (Abandon Call and Retry) signal during DIAL.
FLAG (Condition Code 3)	Read	Receiving an EM code.
	Read Control	Same as Read.
	Write	Not used.
	Write Control	Set by an incoming call during DIAL.
FAULT (Condition Code 4)	Read	Loss of DSR (Data Set Ready), CARRIER, or CTS (Clear to Send). Or if the instruction is active while the terminal is off-line.
	Read Control	Same as Read.
	Write	Same as Read.
	Write Control	Same as Read except that the CTS signal is not required during DIAL. During DIAL, a loss of power to the automatic calling unit or modem, or a disconnected cable will provide FAULT status.

4.6.5.4 *Read*

When a Read instruction is activated, the program establishes a buffer area (Field) in memory to hold the expected number of data characters. Characters are stored in memory starting at the address given in the A field of the instruction and continuing sequentially through the number of locations indicated by the character count given in the B field of the instruction. As each character is received, the address at A is incremented by one, and the character count at B is decremented by one.

If the ATA has been instructed to BLANK-FILL (LB = 1 in the instruction) and a US or EM is received, ending the Read instruction before the expected character count has been received, the ATA generates and transfers null characters into the buffer area until the character count is complete.

If the ATA has been instructed to NON-FILL (LB = 5 in the instruction) and the instruction is terminated before the character count is complete, the unused portion of the storage area is not filled with blanks; the B register in low Common memory is left at the count where the Read terminated and indicates the number of characters remaining to be received. The programmer must plan to subtract the count in the B register from the total expected character count to determine the number of good data characters received.

Terminating Read instructions

When a Read instruction is activated, the B register contains a number representing one less than the number of characters the processor expects to receive. As each character is received, the count in the B register is decremented by one. When the count reaches zero, the processor sets a LAST condition in the ATA, indicating that the character being transferred to storage is the last character expected. If no more data is received, the Read instruction is terminated with condition code 2 (normal). If more data is received after the LAST condition is set, however, the ATA waits until

the input data line becomes quiescent or a Unit Separator (US) code is received before terminating the Read instruction. Any Read instruction is terminated if the ATA receives a Unit Separator (US) or End of Media (EM) code. If an RDKB or RDCKB instruction is active, the operator may terminate the instruction by keying in a US or an EM. An RDR instruction may be terminated in this manner if the STOP READER key is pressed before keying in the terminating character or if an EM or US exists in the paper tape being read. Instructions terminated by EM end with FLAG status (condition code 3). A Read instruction can also be terminated by taking the terminal off-line. The instruction terminates with FAULT status (condition code 4).

OVERWRITE condition

While executing a Read instruction, the ATA sets INTERRUPT each time it is ready to transfer a character to memory. If the processor does not accept the data character before the ATA starts receiving the next character, an OVERWRITE condition occurs, causing the ATA to set ERROR status (condition code 1).

Device numbers in READ instructions

A Read instruction with device number (LA) 0 or 2 designates a keyboard device (RDKB) and a device number 9 indicates a paper tape reader (RDR). A Read Control instruction with device number 0 or 2 also designates a keyboard device (RDCKB).

Read instruction format

The following specifies the fields of the Read Keyboard (RDKB) and Read Reader (RDR) instructions:

Field	*Contents*
F	0000
LA	0, 2 for RDKB
	9 for RDR
LB	1 for blank fill
	5 for non-fill
A	memory address of start of record
B	maximum number of characters that can be received
AC	mark that A address is in Common area
BC	not used
1A, 1B	index control for A and B

The meaning of the status bits for the RDKB and RDR instructions is as follows:

1 ERROR
Set by reception of a parity error, no START bit, no STOP bit, or data overwrite

2 FLAG
Set by reception of an End of Media (EM) code

3 FAULT
Set by loss of Data Set Ready (DSR), loss of carrier, loss of Clear to Send (CTS), or if the operation is activated while the terminal is off-line

Read Control instruction format

The following specifies the fields of the Read Control (RDCKB) instruction:

Field	*Contents*
F	0000
LA	0, 2
LB	3
A	memory address of start of record

Field	*Contents*
B	number of characters to be transferred (10 for hardware) (LOAD REQUEST)
AC	mark that A address is in Common area
BC	not used
1A, 1B	index control for A and B

The meaning of the status bits for the RDCKB instructions is as follows:

1 ERROR

Set by reception of a parity error, no START bit, no STOP bit, or data overwrite

2 FLAG

Set by reception of an EM code

3 FAULT

Set by loss of Data Set Ready (DSR), loss of carrier, loss of Clear to Send (CTS), or if the operation is activated while the terminal is off-line

4.6.5.5 *Write*

A Write instruction to device number (LA) 0 or 2 designates a keyboard printer and to device number 8 designates a paper tape punch. A Write instruction to a read-type device is not accepted by the ATA. The processor then executes the next instruction in the program sequence.

A Write Control instruction with device number 0 designates the keyboard printer. Device number 1 indicates a DIAL operation (CH7) and device number 2 indicates a HANG-UP operation. Device number 8 designates a punch. A time delay is provided for all Write Control instructions to the printer to allow completion of any mechanical action such as a Carriage Return.

Terminating a Write instruction

When a Write instruction to the punch or printer is being executed, operator action is not normally required. However, if the operator decides that manual intervention is necessary, the terminal can be placed OFF-LINE, thereby terminating the operation.

Write Instruction Format

The following specifies the fields of the Write Keyboard (WRKB) and Write Printer (WRP) instruction.

Field	*Contents*
F	0001
LA	0, 2 for the keyboard
	8 for the punch
LB	1
A	memory address for start of record
B	number of characters to be transmitted
AC	mark that A address is in the Common area
BC	not used
1A, 1B	index control for A and B

The meaning of the status bits for the Write Keyboard (WRKB) and Write Printer (WRP) instructions is as follows:

1 ERROR

Not used

2 FLAG

Not used

3 FAULT

Set by the loss of Data Set Ready (DSR), loss of carrier, loss of Clear to Send (CTS), or if the operation becomes active while off-line

Write Control Instruction Format

The following specifies the fields of the Write Control Keyboard (WRCKB), Write Control Printer (WRCP), DIAL, and Write Control Hang-up (WRCH) instructions.

Field	*Contents*
F	0001
LA	0 for Write Control Keyboard (WRCKB)
	1 for DIAL
	2 for Write Control Hang-up (WRCH)
	8 for Write Control Printer (WRCP)
LB	3
A	memory address for start of record (except for WRCH)
	any valid address for WRCH
B	number of characters to be transmitted (except for WRCH)
	any number 0000 to 9999 but preferably 0000 for WRCH
AC	note that A address is in the Common area
BC	not used
1A, 1B	index control for A and B

The meaning of the status bits for the Write Control instructions WRCB, DIAL, WRCH, and WRCP is as follows:

1 ERROR

Set by Abandon Call and Retry (ACR) during DIAL; otherwise unused

2 FLAG

Set by Incoming Call during DIAL; otherwise, unused

3 FAULT

Set by loss of Data Set Ready (DSR), carrier, or Clear to Send (CTS) during DIAL. Also set if the operation is active while off-line

4.6.6 ATA application design considerations

The Asynchronous Terminal Adaptor (ATA) is a console adaptor designed to provide an interface in both local (less than 50 feet) and remote (using modems) applications. The ATA may also be used to attach leased or dial network communications terminals to a System Ten processor; however, such systems should be implemented only with extreme care. There are several potential problems when the ATA is used in general data communications systems. Application system design, therefore, must include several operational procedures and unique program specifications in order to control the ATA-terminal communication environment.

4.6.6.1 *Load*

Any communications terminal meeting the required interface specifications may cause a LOAD condition in the ATA partition if the terminal operator depresses the BREAK key followed by the Escape key (ESC) or if the noise on the communication line simulates an escape code immediately after the BREAK key has been pressed. To prevent a noise-generated LOAD condition when taking the terminal off-line for any reason other than LOAD, always press some character key other than Escape immediately after pressing BREAK.

The following protections should be considered:

1 Never put an ATA in Partition Zero unless the intent is to allow the terminal operator to control the entire processor

2 Place appropriate coding in a user's application program (UAP), residing in another partition, to periodically check the ATA partition to see if it is in a LOAD condition. This may be accomplished by looking for "99Y1" in the P register in low Common and an IOC I/O READ CONTROL instruction active in the ATA partition (indicated by a "2" in character 1 of the B register in low Common). (The first character in the P register is the partition size in thousands minus 1.)

3 Prepare a recovery procedure to cause the ATA program to be reloaded. This may be accomplished automatically if the UAP or a user monitor program is running in Partition Zero and check point-restart procedures are built into the program. The recovery procedure can also be handled by notifying the processor operator to manually reload the partition. The remote terminal operator must be warned about the LOAD problem and instructed about procedures to follow if an "L" appears on the terminal printout (indicative of a LOAD condition).

4.6.6.2 *Timeout*

The ATA does not have a timeout feature. If, for example, the ATA partition program executes a Read instruction and the keyboard operator does not reply, the ATA will wait indefinitely for a response, and the program in the ATA partition will have no control whatsoever over the delay. The application programmer, therefore, must give special consideration to potential timeout problems. The user can protect against unwarranted time delays by monitoring the activity of the ATA partition (within reason) for the particular application. Activity can be checked by a buffer flag in Common memory versus machine time cycles or by a variety of other programming techniques.

4.6.6.3 *On-line/local*

When a terminal operator presses the BREAK key, a local flip-flop in the ATA is toggled, changing state each time the key is pressed. Mode changes may also be caused by noise or momentary interruptions on the communications channel. For this reason, the ATA should not be used in leased line multipoint systems. A BREAK originated at any one of the several terminals will cause the entire network to go into local or on-line mode. Operators of dial systems should be advised to always hang up and try again if random and characters appear on the console printout. This indicates that a noisy line is causing the ATA to toggle back and forth between on-line and local modes.

Control character printing

Control codes are generated by the ATA and printed on the terminal to inform the operator of the status of the partition, and other codes are used by the operator to control the ATA. When using pre-printed forms such as order forms and payroll checks, some terminals can be fitted with an option which prevents printing control characters. A similar option is available for teletype (TTY) terminals. Alternately, provisions can be made in the design of the forms to hide control characters under a pre-printed block.

4.6.6.4 *Write delay timer*

When an ASR teletype terminal TTY33, 35, or 37 is connected to the ATA, request that the teletype machine have the carriage return option installed and that the Write Delay Timer in the ATA be set for 1 second.

4.6.6.5 *Device start/stop codes*

The ATA automatically generates device start and stop codes (delimiting the data from a single Write instruction) to control the keyboard printer or paper tape punch on the terminal. Some teletype compatible terminals have different device characteristics (such as a CRT) that may not work with the given codes. Requirements of each installation must be carefully checked.

Special programming considerations **Chapter 5**

This chapter to be issued later.

Index

Amendment record

Should any additions or amendments to this Manual be necessary, they will be issued in the form of Amendment Lists.
When such changes have been incorporated, an appropriate entry should be made in the table below and the pink instruction sheet(s) should be filed behind this page.

List number	Item number
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to

Model 22 Processor
ICL System Ten
Technical Publication 7529

System Ten

Publications notice

ICL

Technical Publication no.	Title (Notice no.)	Date
7529	*Model 22 Processor (1)*	31/7/78

Page 1-2

Section 1.2.1.2. Paragraph 3 should be amended to read:

Memory can be allocated to a partition in one of two ways:

1 From 1K to 10K in modules of 1K and then from 10K to the maximum of 80K in modules of 10K

2 From 1K to a maximum of 19K in modules of 1K

Distributed by UK Software & Literature Distribution Centre, 60 Portman Road, Reading, Berkshire RG3 1NR

Form 1/401/3 (6.76)