User Hardware Handbook
INTERFACES

The information presented herein is, to the best of our knowledge, true and accurate. No warranty or guarantee, expressed or implied, is made regarding the accuracy of information supplied or capacity, performance or suitability of any product or service since the manner of use is beyond our control.

You are advised that you should ensure that the information contained herein has not been superseded.

All our products, materials and services are sold subject to our Conditions of Sale, available on request.

GEC COMPUTERS LIMITED
Elstree Way, Borehamwood, Hertfordshire.
Telephone No. 01-953-2030

Holding Company — the General Electric Company Limited of England

April, 1977

INTERFACES

CONTENTS

Page

TABLES

IN-TEXT FIGURES

1 INTRODUCTION

A computer system can contain a number of interfaces of various types across which data can be transferred. These interfaces are defined in this manual to enable a user to select a suitable interface or interfaces for the connection of his own equipment and to design the means of connection. It is intended that this manual provide all the information required; however, any designer with a query is invited to contact GEC Computers Limited.

The interfaces are divided into two categories:-

(1) *Internal*

An internal interface is one within a unit at logic level. The user connects to an internal interface by a plug in board mechanically identical to the standard logic board (See figure 8).

(2) *External*

An external interface is a cable connected to an interface board mounted within a unit.

1.1 TYPES OF INTERFACE

(a) *Normal Interface*

The Normal Interface is the interface between a multiplexer channel (such as that situated in the CPU rack known as the Basic Multiplexer Channel) and the peripheral controllers. There are internal and external versions of this interface.

The External Multiplexer Channel (EMC) also provides a Normal Interface and may be used when additional I/O throughput is required to that provided by the BMC.

(b) *Store Interface*

The Store Interface is the interface between a store and any processor (which may be the CPU or an Input/Output processor). There are internal and external versions of this interface, of which, the latter is described in this manual.

(c) *Asynchronous and Synchronous Serial Interfaces*

These external interfaces are used in data communication links often involving the use of Post Office communication facilities.

(d) *Command Interface*

This external interface is used by the Central Processor Unit (CPU) to control input/output processors.

(e) *The Digital Input/Output Interface*

The Digital Input/Output Interface is a specially designed simple interface enabling a user to more easily design and attach his own peripheral controller or controllers than if, for example, the Normal Interface was used.

2. SYSTEM ASPECTS

2.1 INTERNAL INTERFACES

The Normal Interface is available on the backplane of the CPU. Peripheral controller boards are bus connected to the Normal Interface by plugging the boards into particular slots in the CPU. As the number of slots fed by the bus in the CPU is limited, the Normal Interface can be extended by adding Extension Units. An Extension Unit is connected to the CPU by a Normal Interface link and provides 11 slots into which peripheral controller boards can be plugged for bus connection to the extension (secondary) Normal Interface. A secondary Normal Interface can be extended further by the use of a second Extension Unit connected to the first Extension Unit by another Normal Interface link. However, each extension reduces the maximum rate of data transfer that the interface can achieve. Consequently, a better way of providing extension Normal Interface slots is to link each Extension Unit directly to the CPU.

The Normal Interface as implemented by the BMC can operate at up to 800K bytes per second. An Input/Output Processor IOP, can operate the Normal Interface at up to 2M bytes per second. For the majority of connections which a user may wish to make to a system, the Normal Interface is adequate. For some applications, it may be necessary to connect directly to the Store Interface. This provides a high data rate of up to 2.5M bytes. per second (3.6 Mb - 4080) dependent on the peripheral controller design.

2.2 EXTERNAL INTERFACES

The external interfaces available for the connection of user supplied peripherals are:-

Modem Interface Asynchronous

The Asynchronous Modem Interface controller enables Modems or peripherals with the Modem interface to be connected to the CPU via the Normal Interface. Maximum data rate is 9600 baud. Refer to peripheral section of handbook for full details of operation.

Modem Interface Synchronous

This enables a higher data rate (48K baud) than the Asynchronous Modem Interface. Refer to peripheral section of handbook for full details of operation.

Digital Input/Output

The Digital I/O controller enables single peripherals to be connected to the CPU via the Normal Interface. It enables easy connection of simple peripherals. The maximum data rate is 200Kbytes/sec, the actual achievable rate depends on the peripheral controller design. Refer to peripheral section of handbook for full details of operation.

External Link – Normal Interface

A user may connect directly to the Normal Interface via an External Link. The Channel Adaptor board plugs into a CPU or Extension Unit to pick-up the logic level Normal Interface. The user may connect directly to EC4 and EC6 edge connectors on the board. The maximum cable length is 30 metres. Data rate is dependent on channel and peripheral controller design and can be up to 2Mbyte per second.

External Store Interface

The connection to a store from a CPU is via a store link which consists of a store port board (fitted in the store), cable and store link board. The external interface is a cable connection to the store link. The cable may be up to 4 metres long. Maximum data rate with 550ns store is 3.6M bytes per second, and with 800ns store is 2.5M bytes per second.

The external store interface enables the user to provide a special Input Output Processor (IOP) operating directly into store. It needs to be supplemented by the Command Interface to provide program control, interrupts and interlocking of store operations associated with the IOP.

3 NORMAL INTERFACE

3.1 GENERAL INFORMATION

The Normal Interface (NI) is the standard input/output interface. The low speed external interfaces such as the V24 interfaces are controlled by peripheral controllers (e.g. asynchronous modem interfaces) connected to a Normal Interface. The device controlling the NI is called the channel controller. The CPU internal channel controller is called the Basic Multiplexer Channel (BMC).

Generally data transfers are performed autonomously, that is, they proceed independently of the processor, after the program has specified the starting address of the appropriate store data area and the length of the transfer required. Once initiated, an autonomous transfer is controlled by the peripheral controller. Each controller connected to the Normal Interface is allocated a Ready line which determines the controller's priority when requesting service from the BMC. Normally at the end of a data transfer the peripheral initiates a Status Break cycle which transfers peripheral and BMC status information to the store. Then the CPU is interrupted to advise it that the data transfer is completed.

Data may also be transferred to or from a peripheral by program cycles. Data transferred to a peripheral is often used for control e.g. for instructing a peripheral to initiate on autonomous cycle.

Figure 1 shows possible interconnections between a processor, BMC, IOP and two store modules. In practice the Input/Output load might be handled by the BMC alone.

3.2 INTERFACE OPERATION

Types of Cycle

The channel controller conducts a dialogue known as a cycle with one peripheral controller at a time. There are two main types of cycle:

(a) Program cycles. These occur at the instigation of the central processor,

(b) Autonomous cycles. These occur on demand from a peripheral controller, as indicated by an individual READY signal from that controller.

Cycles of either type are allocated by the channel controller, each cycle being divided into subcycles. During each subcycle a subword of 8 or 16 bits is transferred, for peripheral selection or data.

Thus a transfer is the conveyance of data over one or more cycles, each of which is a prescribed sequence of subcycles, with each subcycle providing for the conveyance of one subword.

Autonomous Data Transfers

An autonomous transfer is concerned with the sequential output or input of a data string, from or to an area of store. For this the channel controller has access to a number of Way Control Blocks which contain the addresses and sizes of this data string, and other necessary parameters. The Way Control Block is set up by the Nucleus before the autonomous transfer commences. Usually a program cycle is required to cause the peripheral controller to start the autonomous transfer.

The autonomous transfer of the data string normally occupies many autonomous interface cycles, with either a single data subword or a short burst of data subwords being transferred on each cycle. The channel controller may maintain a large number of autonomous transfers in progress simultaneously, allocating cycles to each peripheral as required. Similarly, a particular peripheral controller itself may carry on more than one autonomous transfer simultaneously.

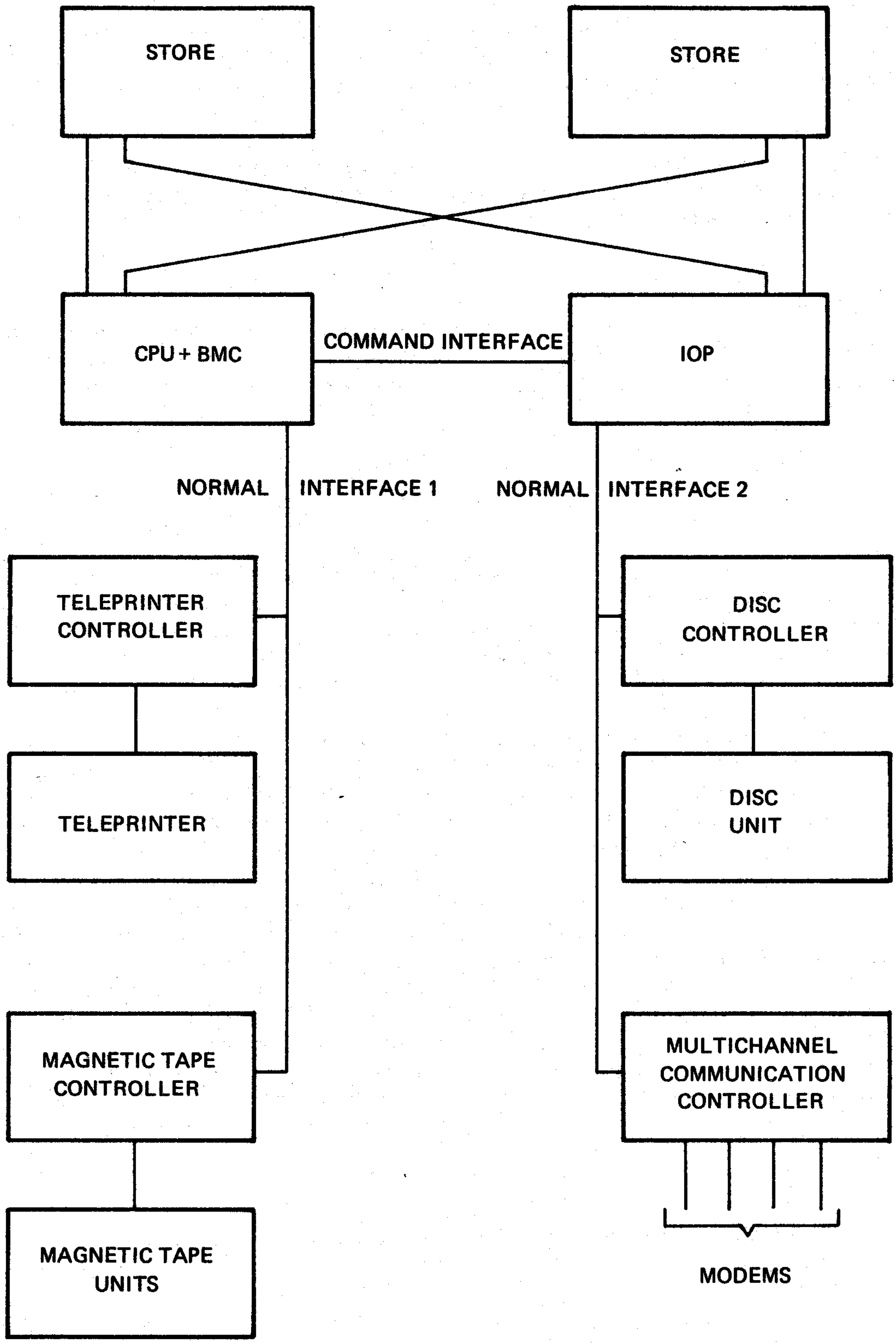


Figure 1: A POSSIBLE SYSTEM INTERCONNECTION

Autonomous Status Breaks

An autonomous Status Break transfer is used whenever a peripheral controller wishes to report a significant change of operating circumstances to its controlling software. A change of program may be associated with this interface cycle, which inputs a single operand subword. This is identified with a particular data string and describes the status of the peripheral controller.

Usually Status Breaks are generated as the result of a peripheral controller completing an autonmous data string transfer. Other uses of Status Break are to report erroneous operation, or to indicate the need for operator (manual) action.

Program Cycles

A program cycle is initiated by a program instruction specifying the direction of transfer and the Way Number. Normally it is used to command or supervise the execution of an autonomous transfer, as noted above. Also it is used explicitly for data transfer, particularly where the peripheral controller is so simple as not to justify the use of the autonomous transfer facility.

Addressing System

Two independent addressing systems are used in the Normal Interface.

The main address is known as the Way Number and lies in the range 0-255. A Way Number corresponds to a separately identifiable source of input data, or destination for output data. A single peripheral controller may use one or more Way Numbers. For example, a paper tape reader uses one Way Number only, whereas a controller for a group of telecommunications lines uses one Way Number per line. A complex peripheral may use one Way Number for data and another for supervisory information.

In program cycles, the software provides a Way Number, which is transmitted by the channel during the first, or select, subcycle. Thus the software uses the Way Number to identify the peripheral concerned. In autonomous cycles the peripheral controller is granted a cycle by the channel and then the peripheral controller itself specifies the Way Number to be used, transmitting this to the channel during the response subcycle of the autonomous cycle. The channel then uses the Way Number to access the appropriate Way Control Block.

The other address is the Ready Number. Each peripheral controller has a unique Ready Number and a corresponding star-connected READY signal. Ready Numbers lie in the range 0-15. When the channel allocates an autonomous cycle, in response to a READY signal raised by a peripheral controller, the channel transmits the corresponding Ready Number in the select subcycle, so initiating a cycle with the peripheral concerned. The peripheral responds with its Way Number, as described above. The Ready Number and READY system are concerned solely with the allocation of autonomous cycles to peripheral controllers in an orderly manner. The subsequent progress of these cycles is determined by the Way Number input by the peripheral, and the Ready Number takes no further part. The Ready Number is not used by, nor is it made available to, the computer program.

Cycle Structure

Normally an interface cycle is made up as described below, though a number of variations can occur in special circumstances:

(i) A select subcycle is transmitted by the channel controller. This subcycle also specifies the type of transfer required. The subword carries a Way Number in the case of program transfers or a Ready Number in the case of autonomous transfer cycles,

(ii) A response subcycle is transmitted by the engaged peripheral controller. This confirms engagement. For autonomous transfer cycles, the subword carries the Way Number,

(iii) An operand-in subcycle is transmitted from the peripheral controller to the channel controller,

(iv) An operand-out subcycle is transmitted from the channel controller to the peripheral controller.

The actual operand to be transferred is carried by the subwords of subcycles (iii) and (iv), accompanied by control information on the mode lines. In fact, the interface cycles described do not transfer useful operands in both directions, and one of the two subwords carries meaningless data; however the mode lines are needed for control purposes in both subcycles so the subcycle cannot be omitted from the cycle.

The cycle may be extended by repeating subcycles (iii) and (iv) one or more times. This is known as burst mode operation. Burst mode operation increases the useful throughput of the interface, at the expense of increasing the length of the cycles concerned. System limitations impose an upper limit on the cycle length.

Interface Signals

Definitions

Interface signals are binary. Their two states are usually referred to as 'true' and 'false'.

All signals are unidirectional. Outgoing signals are generated by the channel controller and inspected by one or more peripheral controllers. Incoming signals are generated by the peripheral controller and inspected by the channel controller.

Star connection describes the situation where an outgoing signal is inspected by, or an incoming signal is generated by, a single particular peripheral controller. Such a signal may be physically present at the connector to other peripheral controllers.

Bus connection describes the situation where an outgoing signal is inspected by many peripheral controllers, or where an incoming signal is the logical 'or' of signals generated by many peripheral controllers. The majority of Normal Interface signals are bus connected, and the electrical specification of internal interfaces allows a distributed 'or' function.

The use of bus connection on incoming lines makes it mandatory that only the peripheral engaged by the select subcycle should place data on these lines.

And-bus connection describes the situation where an incoming signal is the logical 'and' of signals generated by many controllers. One special-purpose signal is connected in this manner. It necessarily uses special transmitter and receiver circuits.

Signal List

Type	Name	Number	Direction	Connection
Timing	OUTTIME	1	Outgoing	bus
Timing	ENGAGED	1	Incoming	bus
Timing	PROCEED	1	Outgoing	bus
Timing	INTIME	1	Incoming	bus
Information	INFO-OUT	16	Outgoing	bus
Mode	MODE-OUT	2	Outgoing	bus
Information	INFO-IN	16	Incoming	bus
Mode	MODE-IN	2	Incoming	bus
Timing	READY	16	Incoming	star
Timing	PERMIT	1	Outgoing	bus
Auxiliary	ACTIVE	1	Outgoing	bus
Auxiliary	INTERLOCK	1	Incoming	And-bus

Interface Expansion

The Normal Interface at logic backplane level as provided directly by a channel controller is called a primary Normal Interface.

A limited number of subunits may be directly connected to the primary Normal Interface, using common power supplies. Each subunit consists of a single logic board or a small number of interconnected logic boards. Simple peripheral controllers are constructed as subunits.

To expand the interface and so accommodate more peripheral controllers, the primary Normal Interface is extended with a Link and an extension unit. An extension unit provides housing and power for peripheral controllers and reproduces the Normal Interface at logic level. A link is in three parts.

(1) A channel adaptor that connects to the primary Normal Interface.

(2) A cable.

(3) A peripheral adaptor that connects to the secondary Normal Interface.

The secondary Normal Interface operates in step with the primary Normal Interface to which it is connected. The timing and logical system used on the secondary interface is identical to that used on the primary interface, except for the degradation in certain fixed time intervals as follows:

i.e. The minimum time periods for the select subcycles, and other signals with a minimum time duration, are less on the secondary interface than are specified for the primary interface.

The effect on the system timing of connecting a link to a primary interface is to cause those controllers on the secondary interface to increase their contribution to the Normal Interface loading by about 10%, Interface loading rules are defined in the System Description manual.

Multiplexing Links

A link provides an independent set of READY signals for its secondary interface. It combines these READY signals and demands cycles from the channel on one or more primary READY lines. When the channel allocates a cycle to such a primary READY line the link replaces the Ready Number by the appropriate secondary Ready Number for the secondary interface.

Timing and Cycle Control

General

The several subcycles which make up an interface cycle are timed and controlled by the four timing signals :

OUTTIME
PROCEED } outgoing generated by channel controller.

ENGAGED
INTIME } incoming, generated by peripheral controller.

Information is transferred over the INFO-OUT, MODE-OUT, INFO-IN and MODE-IN lines. For the purpose of this section INFO-OUT and MODE-OUT are grouped together and called DOUT, and INFO-IN and MODE-IN are called DIN.

The DOUT lines may be either valid (i.e. carrying meaningful data) or invalid when their meaning is not defined.

The DIN lines may be valid or invalid similarly. There is a further state for the DIN lines, that is false when a peripheral is disengaged, so that there shall be no interference with the operation of other peripherals.

Timing

The peripheral transmits its data over the DIN lines accompanied by the two timing signals ENGAGED and INTIME. Whenever the peripheral controller makes the DIN lines valid, it indicates this by a change in a timing signal. Then the peripheral holds DIN, ENGAGED, and INTIME steady until the channel controller acknowledges receipt by changing its timing lines. An exactly similar system is used by the channel controller to indicate when the DOUT lines are valid.

In practice, signals in a parallel group change at marginally different times. This is called 'skew' Such time skew becomes greater whenever the signals are passed through logic stages or over interconnections.

Normal Interface times are measured at the reference connection, where the peripheral connects to the bussed signals. The reference connection is the socket that connects a peripheral controller to the Normal Interface logic level backplane. Signals are so generated that, after allowing for skew, it can be guaranteed, at the reference point that the data is valid at the moment the timing signal changes. Thus any controller attached to a reference connection must allow for skew introduced in its own logic, but not for skew introduced by other equipment.

It is a design feature of the Normal Interface that allowance for skew may be made simply by delaying the timing signals. This may be done up to a limit determined by the channel controller timing.

Quiescent State

When no cycle is in progress the interface is maintained in a quiescent state. The channel controller holds OUTTIME and PROCEED false, and all peripherals are disengaged. Disengaged peripherals must hold DIN false, and INTIME and ENGAGED false.

It is required that a peripheral becomes disengaged immediately if the channel controller makes both OUTTIME and PROCEED false, even where this sequence is not compatible with normal operations. This ensures that the interface may be made quiescent even after a transient failure.

The channel controller does not start a cycle until INTIME and ENGAGED are false.

Cycle Sequence

This description of the sequence of events during an interface cycle covers program, autonomous data, and autonomous status cycles, with or without burst operation. To improve clarity, it concentrates on the essentials. Further details are given in the cycle sequence notes A, B, C, D, E following Table 1.

The state of two timing signals is given at each step in Table 1 (see also Figs. 4 to 7); The interface is designed so that only one of the two signals changes state at any significant time, the other remaining unchanged. Thus no problem of skew between timing signals can arise.

The operative timing signal is bracketted, e.g. (OUTTIME).

Thus the notation: '(OUTTIME) true PROCEED false with DOUT valid' indicates that DOUT is made valid, and that OUTTIME then becomes true allowing for skew as previously described.

Step	Name	Action	Notes
0	-	Channel quiescent, all timing signals false	
1	Select	Channel initiates cycle, outputs select subword. (OUTTIME) true PROCEED false with DOUT valid.	A
2	Response	Peripheral recognises selection, inputs response subword. (ENGAGED) true INTIME false with DIN valid.	
3a		DOUT may change	B
3b	Handshake A	Channel acknowledges responses OK. OUTTIME true (PROCEED) true.	C
4	Operand-in	Peripheral inputs operand. ENGAGED true (INTIME) true with DIN valid	D
5	Operand-out	Channel outputs operand. (OUTTIME) false PROCEED true with DOUT valid	D
6	Handshake B	Peripheral acknowledges operand-out. ENGAGED true (INTIME) false DIN may change	
IF THE CYCLE IS COMPLETE, GO TO STEP 8, OTHERWISE CONTINUE WITH STEP 7.			
7	Handshake A	Channel acknowledges Handshake B. (OUTTIME) true PROCEED true DOUT may change	
CONDITIONS ARE NOW AS FOR STEP 3 CYCLE REPEATS FROM STEP 4.			
8	Close B	Peripheral disengages. (ENGAGED) false INTIME false with DIN zero	E
9	Close A	Channel disengages OUTTIME false (PROCEED) false DOUT may change	E
ABNORMAL TERMINATE (see Abnormal Sequence and Figure 6).			
10	Reject	Channel rejects response 2. (OUTTIME) false PROCEED false DOUT may change.	
11	Close B	Peripheral disengages after reject. (ENGAGED) false INTIME false with DIN zero.	

Table 1: CYCLE SEQUENCE

Table 1 Notes

A The combination OUTTIME true, PROCEED false uniquely identifies a select subcycle, and occurs at no other time.

B Step 3a does not occur until at least 300 nanoseconds after step 1 occurs. Channel controllers must be such that periods shorter than this are not presented to the more distant peripherals, even when a nearby peripheral responds very quickly, and OUTTIME is delayed to allow for skew.

C Following step 3b DIN is invalid for a period as the response subword is removed, and then becomes valid again with the operand subword when INTIME changes (step 4). Peripherals may alter DIN as a result of step 3a (i.e. following change in DOUT). This manner of operation is explicitly permitted and is accommodated by channel controllers.

D In the case of output operand cycles, the INFO-IN lines carry no useful information in step 4, but the MODE-IN lines carry useful information. The unrequired and unspecified data on INFO-IN is nevertheless made valid and timed exactly as though it were indeed useful. Similarly, for input cycles, INFO-OUT lines carry no useful information in step 5, but are made valid and timed correctly.

E Steps 3 and 9 do not occur in sequence, but both may occur arbitrarily soon after step 6.

Note: if step 8 is very close to step 6, then ENGAGED false may actually procede INTIME false as a result of skew. This is of no importance.

Abnormal Sequence

The sequence described under 'Cycle Sequence' is varied if abnormalities occur :

(i) *Unacceptable Response*

The response subword submitted in step 2 may be unacceptable to the channel for any of a number of reasons, e.g.

Autonomous Data requested is not allowed for the Way Number or the data string is exhausted or is not set up.

Status Break is requested, with PERMIT false.

Peripheral rejects program cycle.

In these cases the channel replies to step 2 by step 10 instead of step 3b, following step 3a. The peripheral must detect this and disengage (step 11).

(ii) *No Reponse*

If there is no response step 2 occurs, the channel controller time-outs (see * below) and goes to step 10 to end the cycle. Since no response has occurred, the conditions of step 11 already obtain. A belated response could possibly still arise, though it is suppressed by step 10 and therefore is transient. The channel allows a delay of at least 3 microseconds to accommodate this transient before a new cycle.

* For the exact value of the time-out delay refer to the channel controller specifications. The permitted delay from OUTTIME true to ENGAGED true is not less than 30 microseconds with Basic and External Multiplexer Channels.

(iii) *Other Time-Out Limits*

Some channel controllers may in addition apply time-outs of not less than 3 microseconds to the following timing periods:

PROCEED going true to INTIME going true (step 3 to 4)

OUTTIME going false to INTIME going false (step 5 to 6)

OUTTIME going true (step 7 to 4)

As a check that peripheral controllers do not remain engaged for excessive periods, when in burst mode for example, the following overall time-out period of not less than 75 microseconds is used: PROCEED going true to ENGAGED going false (step 3 to 8).

In all cases, if the channel detects a time-out error, it deletes any subword which may be on the interface by making both OUTTIME and PROCEED false. In such circumstances, the peripheral controller which is engaged should likewise delete any subword which is on the interface by making both INTIME and ENGAGED false and dropping INFO-IN and MODE-IN to logic zero. This usually entails the corruption of the data being transferred. In all cases the channel allows a delay of at least 3 microseconds before commencing a further cycle.

READY and PERMIT Signals

READY Signal

If a peripheral controller requires an autonomous cycle it may make its particular READY true at any time, subject to conditions described under PERMIT heading below. The channel controller, in due course, allocates an autonomous cycle to this peripheral in accordance with an inbuilt priority system. The delay between making READY true and a cycle being allocated depends on the relative priority of the peripheral controller and the amount of activity on the channel.

The peripheral controller normally makes READY false when it makes ENGAGED true, but may again set READY, or allow it to remain true, if it requires a further cycle. Such a cycle is allocated in accordance with the priority system in the usual way.

In general, if a peripheral resets a true READY to false at an arbitrary time, it is uncertain whether or not the channel subsequently will allocate an interface cycle with the corresponding Ready Number. To obtain defined results READY should be made false only:

(a) before ENGAGED is made true during any cycle, addressing the controller concerned,

(b) within 2 microseconds of PERMIT becoming false, (see under PERMIT signal below),

(c) on ACTIVE becoming false (see ACTIVE and INTERLOCK Signal, next page).

PERMIT Signal

Autonomous cycles are of two types – data cycles and status cycles. At certain times the channel controller may be unable to process status cycles. The channel controller indicates these times by means of the PERMIT signal which is held true when status cycles are acceptable, false at other times. The PERMIT signal may change state at any time.

When PERMIT is false, no peripheral should raise READY to request a status cycle. Any peripheral with READY true for this purpose when PERMIT goes false, should make its READY false forthwith. The channel controller does not allocate any autonomous cycle during a period of at least 2 microseconds after PERMIT goes false, so allowing time for the READY signals to be removed.

Note that PERMIT is used only to control READY. A peripheral controller which has already started a cycle should not very its procedure if PERMIT changes.

ACTIVE and INTERLOCK Signals

General

The INTERLOCK system is used to protect systems from the effects of erroneous interface signals generated during power sequencing.

A computer system can consist of a number of independently powered units.

When the power supply for a unit is up sequencing or down sequencing, the unit may transmit arbitrary signals over any interfaces connected to it. It is necessary that other units should be prevented from acting upon these signals.

In small systems, loss of mains power to any unit causes the complete system to halt; conversely, sytem operation does not start unless all units have completed up sequencing. In larger systems the system may be partitioned, so tht some units may down sequence or remain off without causing the entire system to cease operation.

Unit Interconnection (see Figure 2)

The interface is hierarchical, and in each case may be said to connect a superior (channel-like) unit to one or more inferior (peripheral-like) units. The superior unit is that which controls the interface, i.e. the unit 'nearer' the processor in simple systems.

A channel controller generally has no unit superior to it. Most ordinary units have one immediately superior unit. It is possible to envisage units which are connected to several immediately superior units, over separate interfaces. A unit with no superior units is referred to in the following sections as a master unit. All other units are slave units.

In the following paragraphs it is assumed, unless otherwise stated, that the system being considered is not partitioned and has only one master unit.

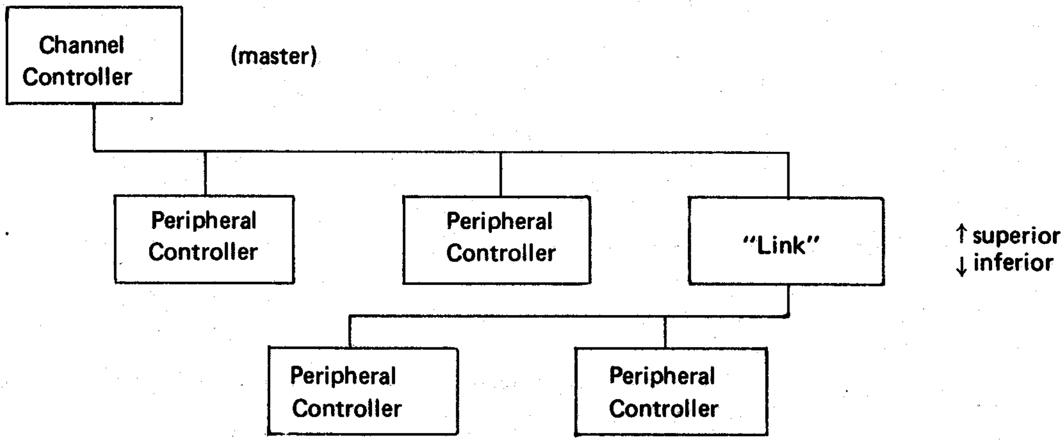


Figure 2: UNIT HIERARCHY

INTERLOCK Signals

Each independently powered peripheral unit generates an INTERLOCK signal, which is true only if the power supply for the unit is established. In the event of loss of mains, or intentional down sequencing, the unit makes INTERLOCK false at least 1 millisecond before the unit begins to malfunction or to send erroneous interface signals. INTERLOCK itself must be transmitted between independently powered units by a 'power-tolerant' circuit which can maintain a 'false' signal in the absence of power and also during power sequencing.

INTERLOCK must also be made false by any unit which receives a false INTERLOCK from an inferior unit. Where several inferior units are connected via a 'bus', the drive system must generate the logical 'and' of the INTERLOCK signals.

The master unit can form an overall INTERLOCK signal which is true only when the complete system is operational and becomes false at least 1 millisecond before malfunctioning occurs as a result of down sequencing.

When peripheral controllers share a common power supply with the channel controller, or with link hardware via which they are connected, there is no need for them to use the INTERLOCK line as the necessary action in case of supply failure is taken by the channel controller or link hardware as appropriate.

ACTIVE Signals

The master unit generates a signal called ACTIVE, which it transmits to its inferior units.

Any unit receiving ACTIVE as false must:

(i) Transmit ACTIVE false to any inferiors.

(ii) Cease operation and enter the reset state.

(iii) Ignore other signals from its superior, since these may have been incorrectly transmitted.

The ACTIVE signals are transmitted by power-tolerant circuits which can continue to transmit false during sequencing or with no power.

The master normally makes ACTIVE false as soon as it detects overall INTERLOCK false. During switch-on, ACTIVE becomes true.

A master unit may, in response to manual control, make the ACTIVE signals false to cause system reset to occur, even though no power sequencing is involved.

When ACTIVE is made false for any reason it remains in that state for at least 5μs.

Information and Mode Formats

General

Tables 2 and 3 which follow show in detail how the 8 or 16 INFO signals and 2 MODE signals in each direction are used, during the various types of interface cycle.

During the SELECT subcycle, MODE-OUT is used to specify the type of cycle allocated by the channel controller as follows:

MODE - OUT	SELECT Subcycle Type
00	Autonomous
01	Not used (Spare)
10	Program input
11	Program output

Table 2 shows the continuation of the autonomous cycle, and table 3 deals with the program cycle, Notations are explained in the accompanying notes. The symbol * is used where INFO-IN signals are ignored by the channel, or where the INFO-OUT signals may take unspecified values. In general operands transmitted via the INFO-IN and INFO-OUT signals may take any values; exceptions are listed below.

Cycle Class	Subcycle	M INFO-MS M	L INFO-LS L	MODE		Notes
		0 7	0 7	0	1	
Autonomous (Data in)	Select	*	Ready Number	0	0	
	Response	width mark	Way Number	1	1	AD
	Op-in	optional	operand	x	x	B
	Op-out	*	*	y	y	B
Autonomous (Data out)	Select	*	Ready Number	0	0	
	Response	width mark	Way Number	1	0	AD
	Op-in	*	*	x	x	B
	Op-out	optional	operand	y	y	B
Autonomous (Status break)	Select	*	Ready Number	0	0	
	Response	*	Way Number	0	1	D
	Op-in	optional	status	0	0	C
	Op-out	*	*	0	0	

Table 2: AUTONOMOUS CYCLES

Table 2 Notes:

A The width mark consists of eight bits and may take the values 00000000 and 10000000 only (i.e. zero or 128). The zero value indicates that the peripheral requires one byte from the data string to be transferred in each subsequent operand subcycle. This occurs by default in the case of 8-bit peripheral controllers. The value 128 indicates that two bytes are to be transferred from the data string in each operand subcycle.

B The mode bits xx and yy determine whether further cycles or subcycles are possible (see Operand Subcycle MODE Meanings later). The optional m.s. byte of the operand is determined as above.

C The width of status information is determined by the channel, irrespective of the code transmitted on the m.s. INFO byte during the response subcycle. The l.s. byte of the status information may not be zero (i.e. 00000000).

D The MODE-IN code is as follows:

MODE-IN	Cycle Type
00	Not used
01	Autonomous Status Break
10	Autonomous Data Out
11	Autonomous Data In

Cycle Class	Subcycle	M 0 INFO-MS M 7	L 0 INFO-LS L 7	MODE 0	MODE 1	Notes
Program Input (Accepted)	Select	Subaddress	Way Number	1	0	E
	Response	*	*	1	1	
	Op-in	optional	operand	x	x	EG
	Op-out	*	*	y	y	G
Program Input (Rejected)	Select	Subaddress	Way Number	1	0	
	Response	*	*	z	z	F
	SUBCYLES SUPPRESSED					
Program Output (Accepted)	Select	Subaddress	Way Number	1	1	E
	Response	*	*	1	1	
	Op-in	*	*	x	x	G
	Op-out	optional	operand	y	y	EG
Program Output (Rejected)	Select	Subaddress	Way Number	1	1	
	Response	*	*	z	z	F
	SUBCYCLES SUPPRESSED					

Table 3: PROGRAM CYCLES

Table 3 Notes

E If the peripheral controller is of 16-bits width then all 16 bits are transmitted, and if the m.s. byte is not needed it must be removed by software.

If the peripheral controller is of 8-bits width, the m.s. byte is input as zero.

F If the response mode is not 11 the cycle is abandoned by the channel controller as explained under 'Abnormal Sequences'. The MODE-IN code zz is as follows:

MODE-IN zz	Response to Program Cycle
00	Reject - underfined fault
01	Reject - invalid function
10	Reject - peripheral controller busy
11	Accept

Some channel controllers (e.g. BMC) make zz available to software. Reference should be made to the appropriate channel controller manual for details.

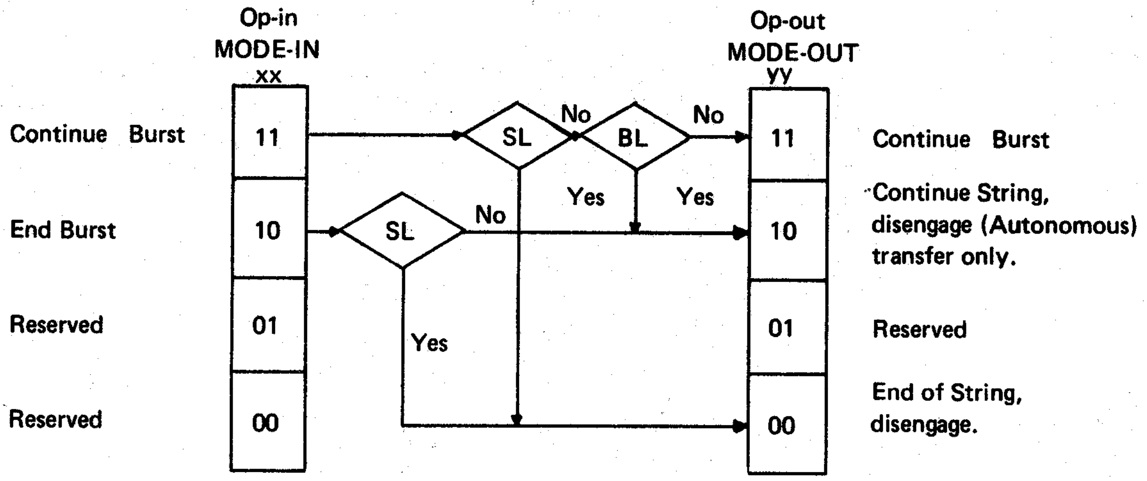
G The Mode bits xx and yy determine whether further cycles are possible, see under 'Operand Subcycle MODE Meanings' which follows: the BMC can transfer only one word in a program cycle, hence it ignores xx and makes yy zero.

Operand Subcycle MODE Meanings

The peripheral controller uses MODE-IN during the operand input cycle to indicate its requirement for further subcycles during the present burst.

The channel controller uses MODE-OUT during the operand output cycle to indicate whether further subcycles are to be performed, and in autonomous transfers to indicate when the data string is exhausted.

The codes used are given in Figure 3, which also shows how the channel controller's decision is related to the peripheral controller's request. The channel controller can if necessary end the string or the burst even if the peripheral has requested continuation. Not all channel controllers impose a burst limit other than as an overall time-out.



SL = String Limit - End of data, or Program Transfer Limit

BL = Burst Limit (Autonomous Transfers only)

Figure 3: OPERAND SUBCYCLE MODE MEANINGS

Timing Diagrams

The following figures 4, 5, 6 and 7 illustrate the sequences of events which occur during interface operations. The figures are idealised as regards the detail of signal transitions; they are not to any defined or fixed scale on either axis.

A positive logic convention is used; this does not imply that positive logic is in fact used on the reference version or any other version of the interface.

The following conventions are used:

═ Indicates when a signal or set of signals is valid and is to be maintained unchanged in one or other state.

▨ Indicates when a signal or set of signals is invalid and therefore may be in either state or in transition.

_┌ ✱ Indicates the definitive timing signal concerned when more than one signal changes simultaneously; in other words the signal which is in practice delayed to allow for 'skew' (see under 'Timing')

Step numbers shown in the sequences correspond to those listed under 'Cycle Sequence'.

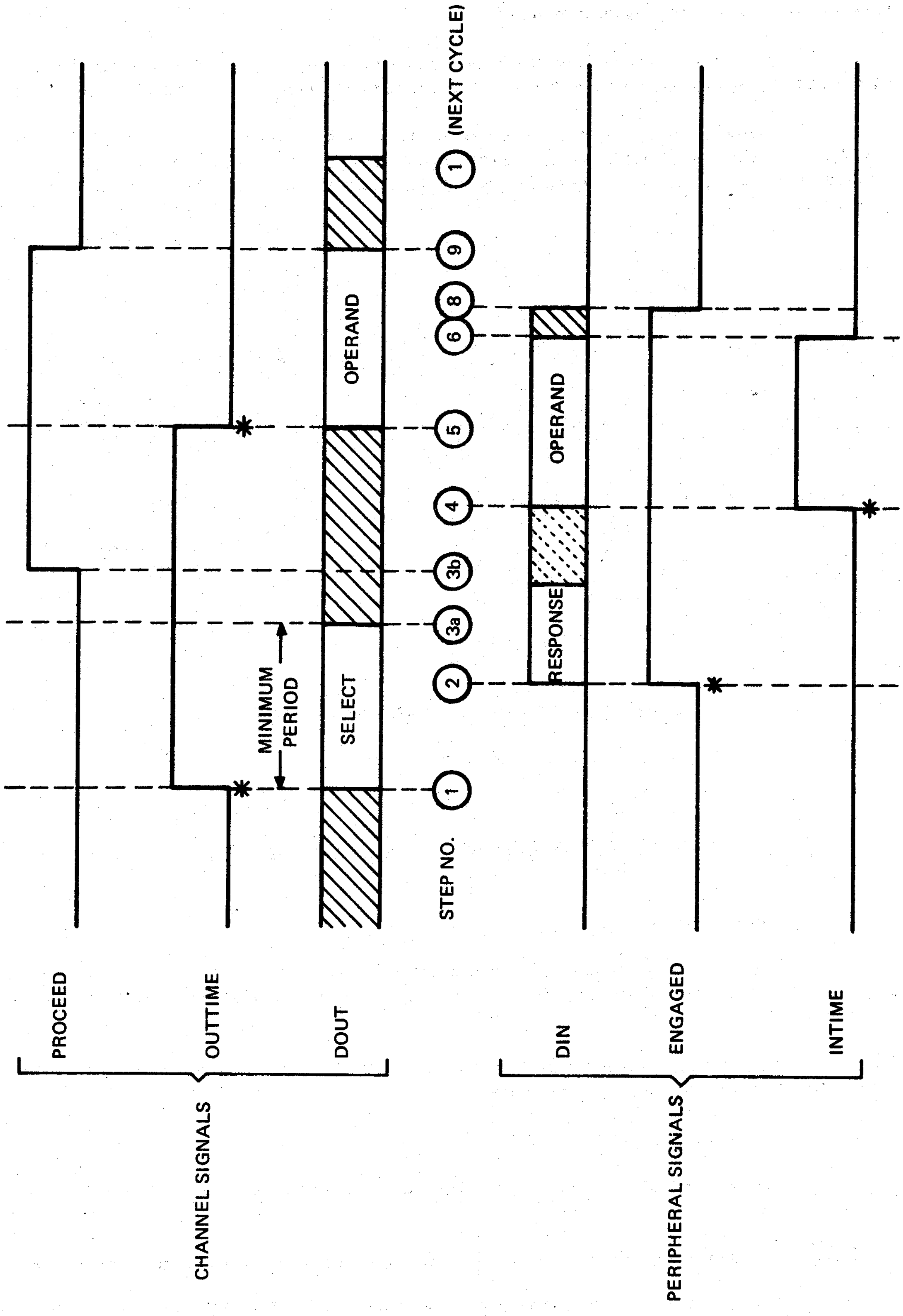


Figure 4: AUTONOMOUS/PROGRAM CYCLE TIMING - SINGLE OPERAND MODE

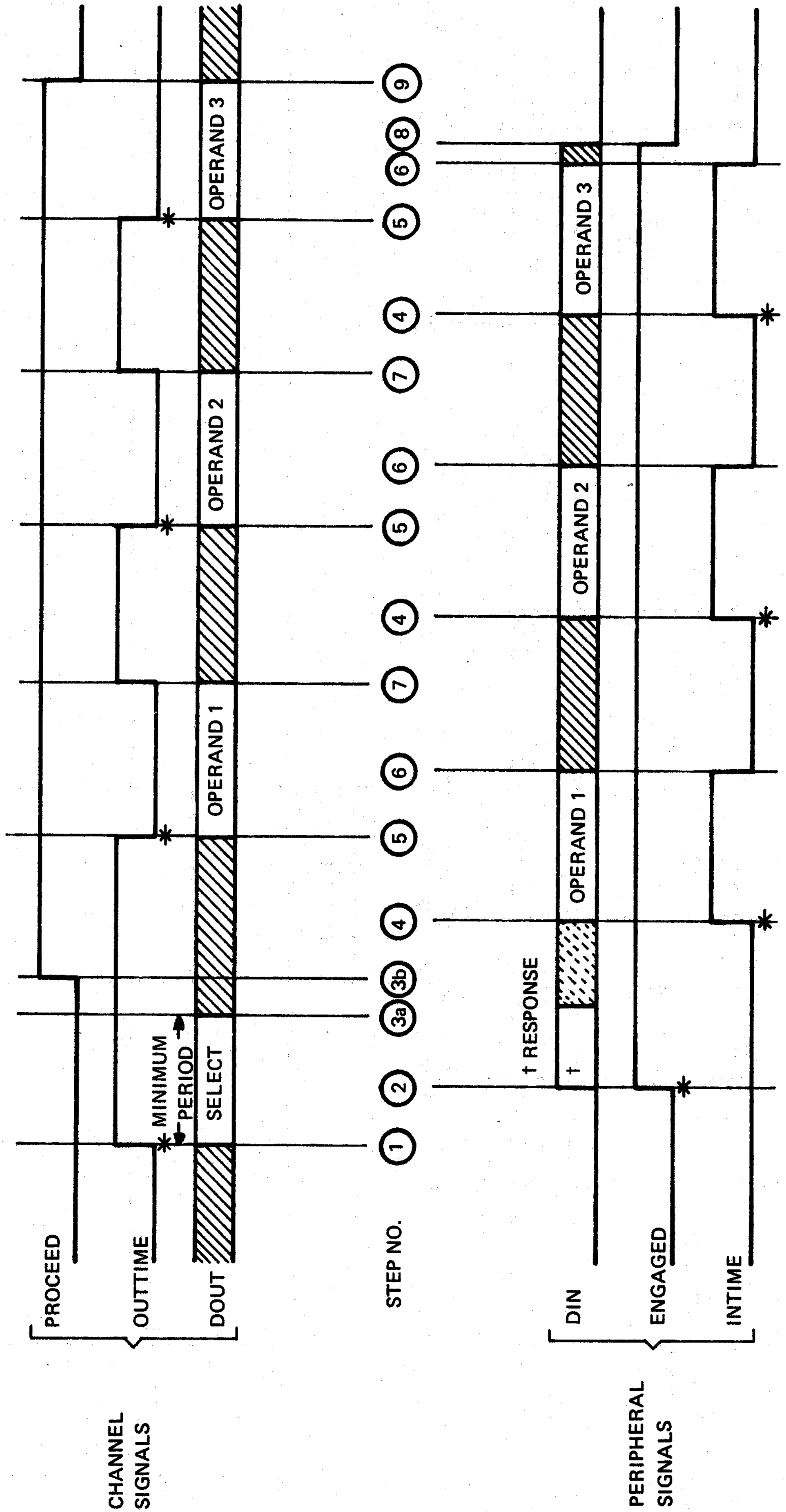


Figure 5: AUTONOMOUS/PROGRAM CYCLE TIMING - BURST OPERAND MODE

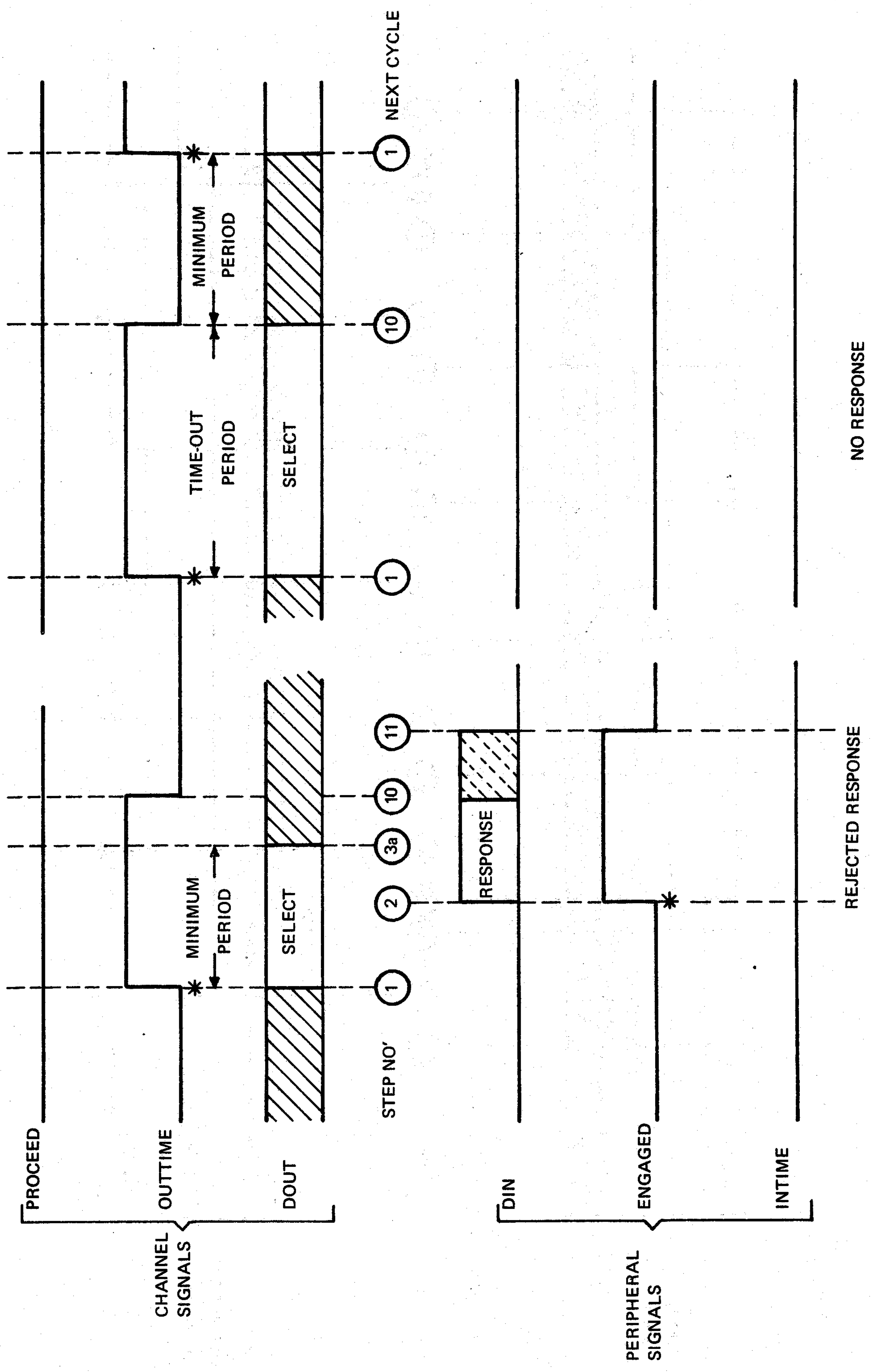


Figure 6: AUTONOMOUS/PROGRAM CYCLE TIMING – ABNORMAL CONDITIONS

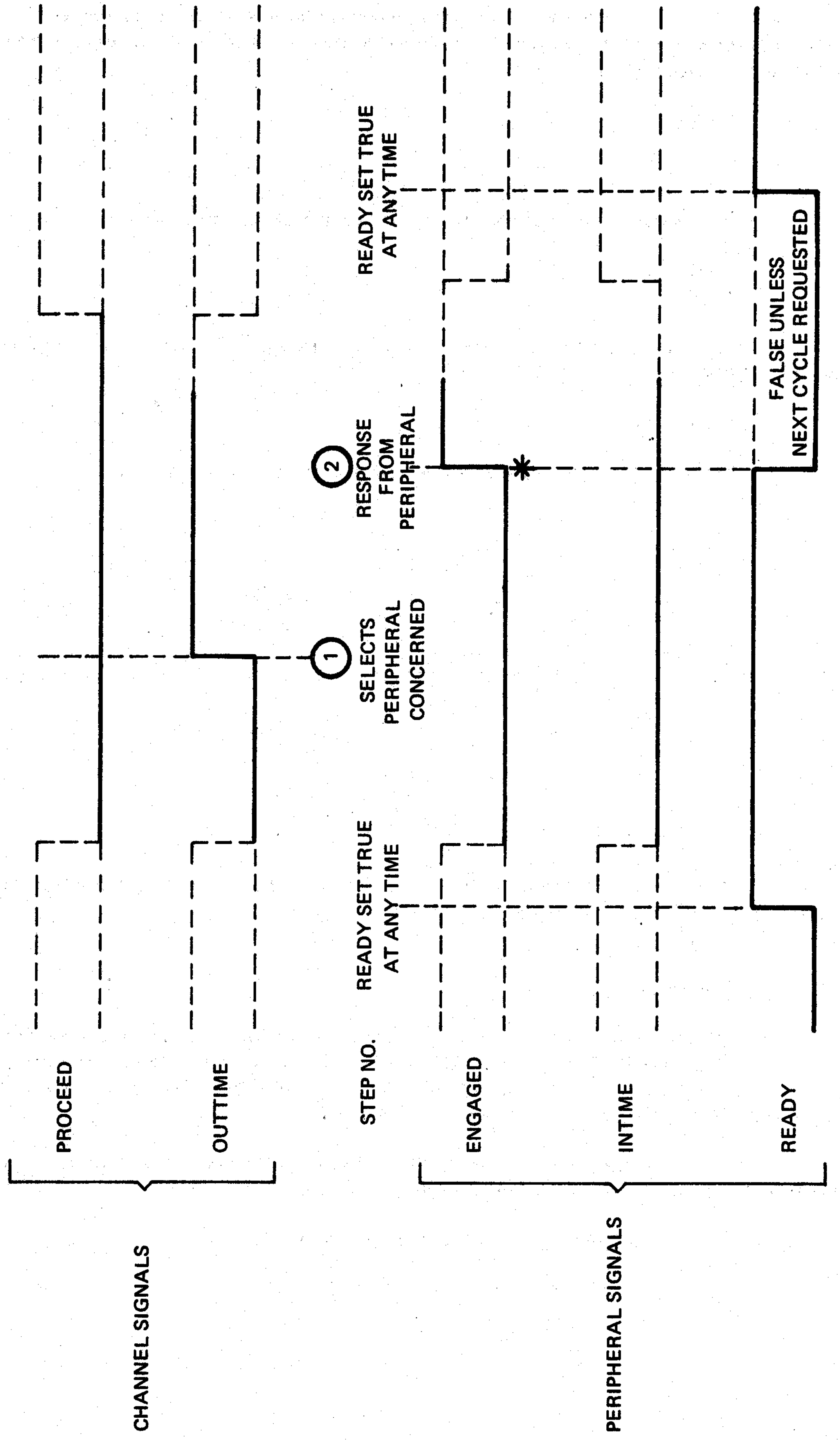

CHANNEL SIGNALS
PROCEED
OUTTIME
STEP NO.
PERIPHERAL SIGNALS
ENGAGED
INTIME
READY
READY SET TRUE AT ANY TIME
1
SELECTS PERIPHERAL CONCERNED
2
RESPONSE FROM PERIPHERAL
*
READY SET TRUE AT ANY TIME
FALSE UNLESS NEXT CYCLE REQUESTED

Figure 7: AUTONOMOUS CYCLE REQUEST

3.3 PHYSICAL INTERFACE CONNECTIONS AND SIGNALS

This section describes the basic rules applicable to all peripheral units to be connected to the 16 bit logic level Normal Interface. If the 8 bit version of the interface is being used the references to the most significant byte should be ignored.

Circuits

Transmitters

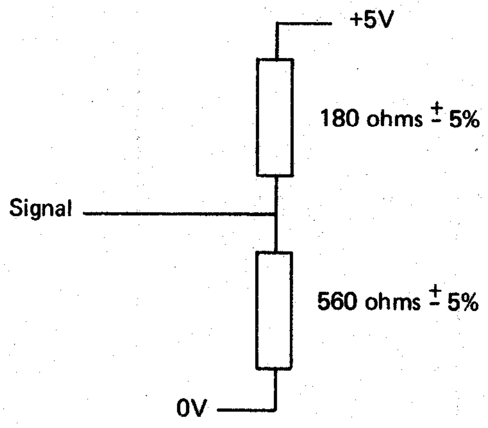
The open collector driver SN7438 quad two input NAND gate is used generally as the peripheral logic transmitter.

Receivers

The standard peripheral logic level receiver is a single logic load input 74N series gate with integral input clamp diodes.

Termination

The signal termination network below is used to terminate each signal line:



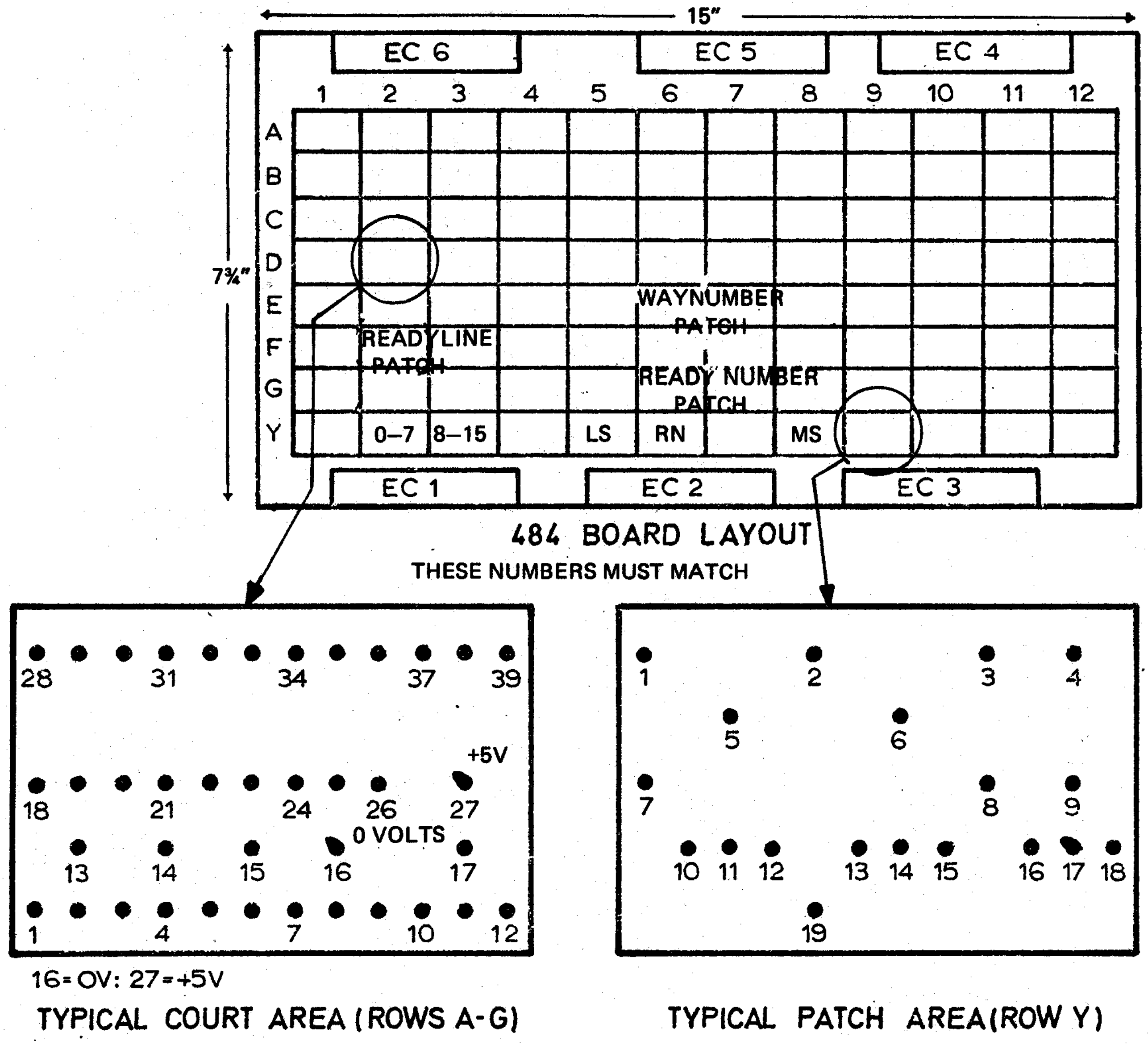
The termination networks are placed on a termination board at one end of the logic rack for outgoing lines, and on the channel receiver board for incoming lines.

Backplane

The interconnecting backplane consists of a fibre-glass printed circuit board of which two layers are used for the logic level NI. The layer on the surface of the fibre-glass nearest to the logic boards is a signal layer on which the various interface signals are tracked, with earth guard tracks between critical signals. The signal layer is separated by 0.030 inch from a buried earth plane.

The edge connector is a 70 way double sided Type BICC Burndy PB44DD 35-A-02-02 and is designed to fit the standard logic board (see figure 8).

The allocation of NI to the connector pins is shown in Table 4



COMPONENT SIDE VIEW

Figure 8: STANDARD LOGIC BOARD

	Connector EC1		Connector EC3	
	B	A	B	A
1	+5V P	+5V P	0V	INFO - OUT L0 £
2	-15V P	-15V P	INFO - OUT L1 £	INFO - OUT L2 £
3	-5V P	-5V P	INFO - OUT L3 £	INFO - OUT L4 £
4	+15V P	+15V P	INFO - OUT L5 £	INFO - OUT L6 £
5	0V P	0V P	INFO - OUT L7 £	MODE - OUT 0 £
6			MODE - OUT 1 £	0V
7	0V		INFO - OUT M0 £	INFO - OUT M1 £
8	READY 0 £	0V	INFO - OUT M2 £	INFO - OUT M3 £
9	0V	READY 1 £	INFO - OUT M4 £	INFO - OUT M5 £
10	READY 2 £	0V	INFO - OUT M6 £	INFO - OUT M7 £
11	0V	READY 3 £	0V	
12	READY 4 £	0V		
13	0V	READY 5 £		
14	READY 6 £	0V		
15	0V	READY 7 £		
16	READY 8 £	0V		
17	0V	READY 9 £	0V	INTERLOCK
18				
19	0V	READY 10 £		
20	READY 11 £	0V		
21	0V	READY 12 £		
22	READY 13 £	0V		
23	0V	READY 14 £		
24	READY 15 £	0V	OUTTIME £	0V
25	0V	0V	0V	ENGAGED £
26	INFO - IN L0 £	INFO - IN L1 £	PROCEED £	0V
27	INFO - IN L2 £	INFO - IN L3 £	0V	INTIME £
28	INFO - IN L4 £	INFO - IN L5 £	PERMIT £	0V
29	INFO - IN L6 £	INFO - IN L7 £	0V	ACTIVE £
30	INFO - IN L8 £	INFO - IN L9 £		0V
31	0V	INFO - IN M0 £	0V P	0V P
32	INFO - IN M1 £	INFO - IN M2 £	+15V P	+15V P
33	INFO - IN M3 £	INFO - IN M4 £	-5V P	-5V P
34	INFO - IN M5 £	INFO - IN M6 £	-15V P	-15V P
35	INFO - IN M7 £	0V	+5V P	+5V P

Edge connector pin numbers

P = Power Rail

Table 4: NI LOGIC LEVEL CONNECTIONS

Signal Polarity

Negative logic convention is used on the backplane NI:

Logic FALSE (0)	> + 2.5V	i.e. HIGH voltage level
Logic TRUE (1)	< + 0.5V	i.e. LOW voltage level

Indication is by the "£" sign after the signal name. One signal, INTERLOCK, is an exception to the negative logic convention.

Engineering Restrictions

The board track length of the signals should not exceed 3 inches and should be guarded by an earth track running parallel to and separated by 0.050 inch from the signal track for as much of its length as possible. The earth track should be connected to the logic board earth close to the relevant transmitter or receiver package.

Parallel running of any information or mode signal to a timing or auxiliary signal should be avoided.

Power

The power pin connections to a peripheral logic board are standard.

There are four voltage rails available, +5V, +15V, -15V, and -5V. The +5V and -5V rails have a maximum variation of 0.2V, the +15V and -15V rails have a maximum variation of 2V.

The ± 5V rails are established before and fall after the ± 15V rails.

4 EXTERNAL LINK – NORMAL INTERFACE

4.1 GENERAL INFORMATION

The External Link (EL) provides a version of the Normal Interface to which a user can connect a peripheral controller or controllers without being constrained by the GEC Computers Ltd. mechanical standards. It is a cable, as opposed to a backplane, defined Normal Interface.

The Peripheral Link consists of a channel adaptor board, a cable, and a peripheral adaptor board. The latter produces a secondary logic level interface up to 100 ft. from the channel or primary Normal Interface. The EL is supplied less the peripheral adaptor board with a choice of cable lengths from 10 ft to 100 ft.

READY lines are multiplexed onto a single line within the peripheral adaptor board. This single-line is patched on the channel adaptor board connected to the Normal Interface. Hence there is only a single READY line on the cable defined EL.

4.2 LOGICAL OPERATION

The logical operation is defined in the Normal Interface section of this manual. There are some differences, however, which are described below:

FAST OUTTIME

This is an additional signal required for the peripheral adaptor board and is used to statisise the READY lines on the secondary Normal Interface. It should not be used in place of OUTTIME when connecting directly to the interface.

OUTTIME, INTIME and ENGAGED

These control signals are delayed by the channel adaptor to allow for their inherent transmission delay differentials in the Link.

ME

This signal is generated by the channel adaptor when the patched encoded Ready Number corresponds with the data output lines. It is used by the peripheral adaptor during the select phase of an autonomous cycle. This signal is not required if the encoded Ready Number is recognised within the user's peripheral controller.

LINK ENABLE AND DISENABLE

The channel adaptor contains logic which enables a user's peripheral to isolate itself from the host system. This facility can be used in three ways:

(a) To isolate the user's peripheral or peripherals from the host system,

(b) To allow either of two sets of user peripherals to be connected to the host system,

(c) To permit a user's peripheral to be switched between two or more processors.

NOTE: (b) and (c) above imply the use of multiple interface links.

Normally the ENABLE signal is held true. When it goes false the channel adaptor prevents the incoming signals from the external interface reaching the logic level NI. When the user peripheral sets ENABLE true, the EL is not connected to the NI until at least 5μs have elapsed after the logic level NI has finished its correct cycle.

When the ELis used without the peripheral adaptor board, the logic on the channel adaptor board allowing faster data transfers in burst mode must be disabled by the fitting of a wire link.

INTERLOCK

The channel adaptor board can be patched to prevent INTERLOCK from the EI reaching the NI. When this is done a permanently true INTERLOCK is transmitted to the NI.

Data Latches

The state of the 16 INFO-OUT lines and the 2 MODE-OUT lines from the NI is maintained on the channel adaptor board while OUTTIME is true.

4.3 IMPLEMENTATION

The channel adaptor board is a GEC Computers Ltd. standard logic board with transmitters, receivers etc. mounted on the board in dual-in-line packages. The 8 bit Normal Interface requires one cable connected to the board. The 16 bit version requires two cables.

Each cable contains 37 twisted pair conductors (34 of which are used) with overall screening. Each end is terminated with a standard shrouded B.I.C.C. Burndy PB4DD 35-02-02 connector. The cable length is specified by the customer up to 100 ft.

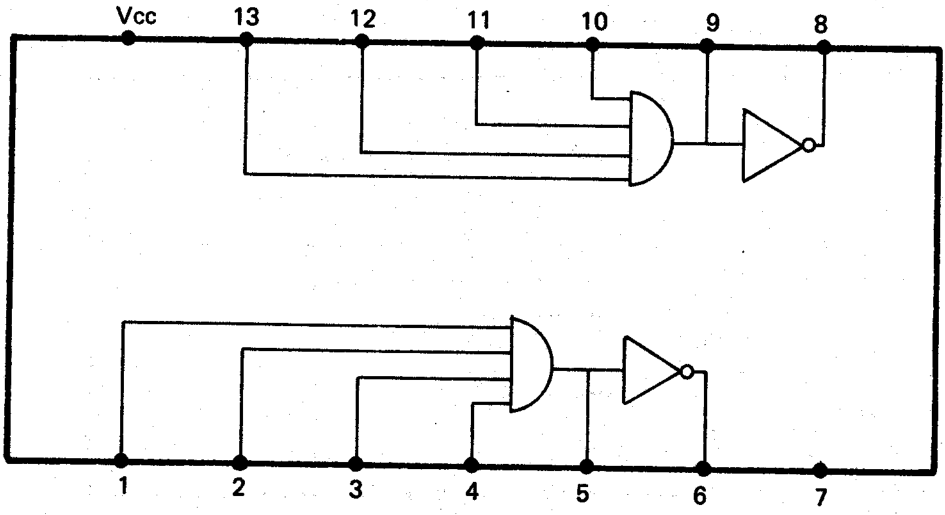
A balanced transmitter/receiver system is used which gives a common mode noise immunity of 10 volts.

Transmitter

This is a Texas SN75183 or a National Semiconductor DM8830N with the following specifications:

Package	:	14 pin Dual - in - line
Type	:	Dual differential line driver
Logic	:	Dual 4 input NAND or AND depending on the output conditions
Power Supply per each driver:		+5V, 36 mA internal consumption 80 mA external drive.
Input Load	:	3 Normal TTL loads (4.8 mA max.)

The package functional logic diagram and package connections are shown below:

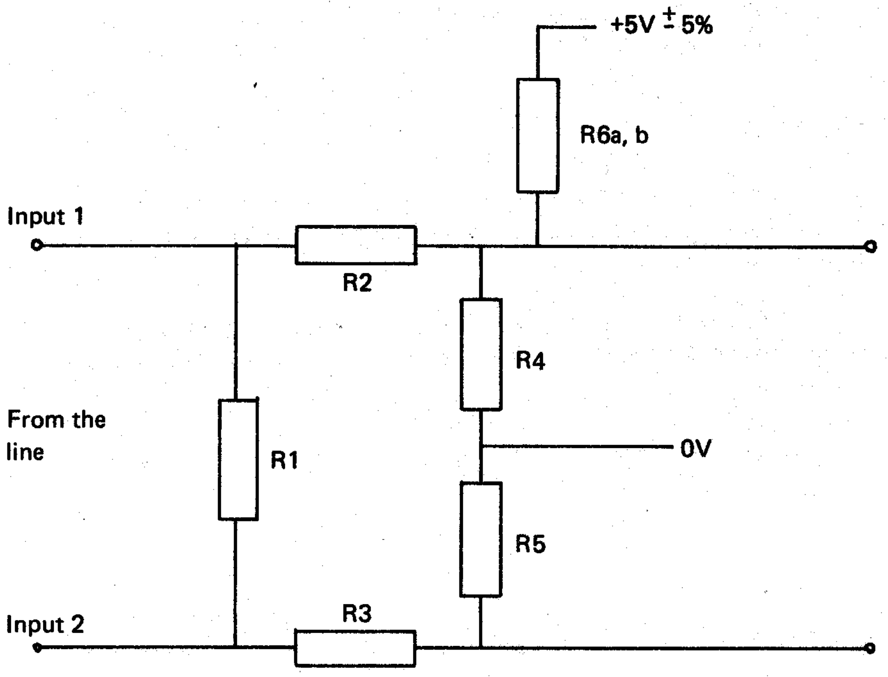


H.F. Decoupling

Each transmitter package requires 0.01 μF decoupling capacitor from Vcc to ground, located in the area occupied by the transmitter.

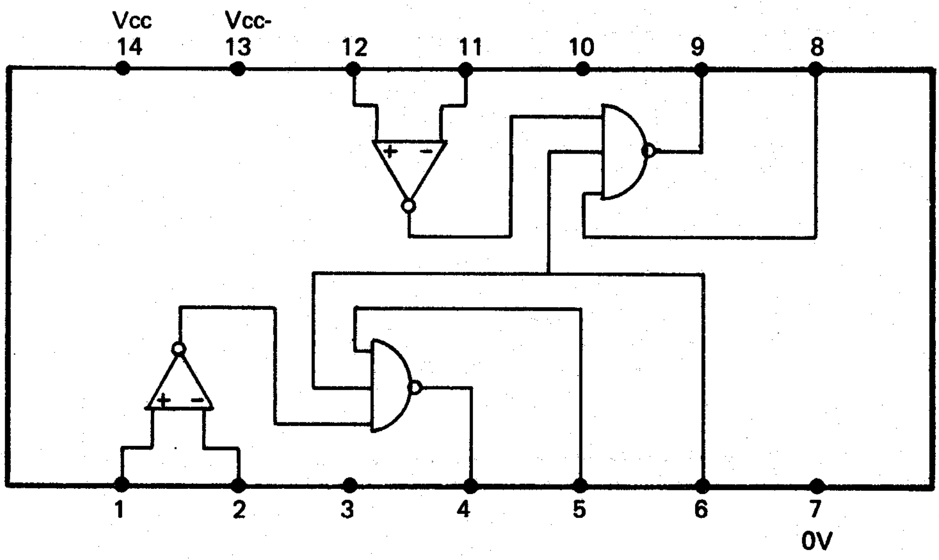
Receiver

The differential line receiver is used with a termination network on its input. In addition to terminating the lines, the network provides an input attenuation of 4.6:1 and fail safe biassing of the receiver. The termination network is shown below:



R1	150 ohms, ¼W ± 5%	R4, R5	330 ohms, 1/10W ± 1%
R2	1.2K ohms, 1/10W ± 1%	R6a	27K ohms, 1/8W ± 5%
R3	1.2K ohms, 1/10W ± 1%	R6b	10K ohms, 1/8W ± 5% (High Integrity Signal INTERLOCK£ only)

To keep circuit delays to a minimum, the network should be fitted as close as possible to the receiver with a tracking length of less than 2 in. The receiver function logic diagram is shown below:



H.F. Coupling. Each receiver package requires a 0.01 μF decoupling capacitor from Vcc to ground and a 0.005μF decoupling capacitor from -Vcc to ground, both located in the area occupied by the receiver package.

Receiver type no.	:	Texas SN75107 or SN75108 (open collector)
Type	:	Dual differental line receiver
Logic	:	Dual 3 input NAND (one input common)
Power requirements	:	+5V, 30 mA max. per dual circuit -5V, -15 mA max. per dual circuit
Output Load	:	10 Normal TTL loads

Link Delays from Input to Transmitter to Output of Receiver

	Min.(ns)	Typical (ns)	Max. (ns)	
Transmitter SN75183	6	12	16	
100 ft. cable	143	150	157	(A)
Attenuator to Receiver-threshold.	8	9	10	
Rise time allowance	0	–	8	(B)
Receiver SN75107	10	17	25	
Total (ns)	167	188	216	

(A) The cable allowance is 1.5 ns/ft. delay plus
0.15 ns/ft skew

(B) Due to 1.5V differential noise immunity

Signal Timing

At their respective transmitter inputs on the channel adaptor board:

(a) INFO-OUT and MODE-OUT are established a minimum of 68 ns before OUTTIME goes true,

(b) INFO-OUT and MODE-OUT are maintained for a minimum of 58 ns after OUTTIME goes false,

(c) ME is established a minimum of 36ns before OUTTIME goes true.

At their respective outputs on the channel adaptor board:

(a) INFO-IN and MODE-IN must be established a maximum of 46 ns after ENGAGED goes true or INTIME goes true.

Screening and Earthing

The interface cable incorporates an overall copper braid screen with provision for earthing at both ends. 30in. long flying leads are connected to the copper braid so that they can be earthed to the 'Safety Earth' studs of the cabinets at each end of the cable. The cable screen and the cabinets provide a continuous screen around the signal transmission paths.

Several conductor pairs in each interface cable are allocated for logic 0V connection.

Cable Connections

One cable, connected to EC4 on the channel adaptor board, carries all the signals for the 8 bit Normal Interface. For the 16 bit Normal Interface an additional, mechanically similar, cable is connected to EC6. The allocation of signals to the edge connectors is shown in Table 5

Signal Levels

On the cable a signal is true when it is greater than 1.8 volts positive relative to its inverted signal (labelled with a £ sign).

i.e. OUTTIME is true if:

$$(V_{OUTTIME} - V_{OUTTIME\,£}) > +\,1.8 \text{ volts}$$

As the transmitter/receiver system is a balanced one, OUTTIME is false if:

$$(V_{OUTTIME\,£} - V_{OUTTIME}) > +\,1.8 \text{ volts}$$

	Cable connected to EC4		Cable connected to EC6	
	B	A	B	A
1	0V	0V		
2	INTERLOCK £	INTERLOCK		
3	READY M £	READY M	INFO - OUT M7 £	INFO - OUT M7
4	0V	0V	INFO - OUT M6 £	INFO - OUT M6
5	INTIME	INTIME £	INFO - OUT M5 £	INFO - OUT M5
6	ENGAGED £	ENGAGED	INFO - OUT M4 £	INFO - OUT M4
7	ACTIVE £	ACTIVE	INFO - OUT M3 £	INFO - OUT M3
8	PERMIT	PERMIT £	INFO - OUT M2 £	INFO - OUT M2
9	PROCEED	PROCEED £	INFO - OUT M1 £	INFO - OUT M1
10	OUTTIME	OUTTIME £	INFO - OUT M0 £	INFO - OUT M0
11	ENABLE £	ENABLE	INFO - IN M7	INFO - IN M7 £
12	FAST O/T	FAST O/T £	INFO - IN M6	INFO - IN M6 £
13	ME £	ME	INFO - IN M5	INFO - IN M5 £
14			INFO - IN M4	INFO - IN M4 £
15	MODE - OUT 1 £	MODE - OUT 1	INFO - IN M3	INFO - IN M3 £
16	MODE - OUT 0 £	MODE - OUT 0	INFO - IN M2	INFO - IN M2 £
17	INFO - OUT L7 £	INFO - OUT L7	INFO - IN M1	INFO - IN M1 £
18	KEY	WAY	KEY	WAY
19	INFO - OUT L6 £	INFO - OUT L6	INFO - IN M0	INFO - IN M0 £
20	INFO - OUT L5 £	INFO - OUT L5		
21	INFO - OUT L4 £	INFO - OUT L4		
22	INFO - OUT L3 £	INFO - OUT L3		
23	INFO - OUT L2 £	INFO - OUT L2		
24	INFO - OUT L1 £	INFO - OUT L1		
25	INFO - OUT L0 £	INFO - OUT L0		
26	MODE - IN 1	MODE - IN 1 £		
27	MODE - IN 0	MODE - IN 0 £		
28	INFO - IN L7	INFO - IN L6 £		
29	INFO - IN L6	INFO - IN L6 £		
30	INFO - IN L5	INFO - IN L5 £		
31	INFO - IN L4	INFO - IN L4 £		
32	INFO - IN L3	INFO - IN L3 £		
33	INFO - IN L2	INFO - IN L2 £		
34	INFO - IN L1	INFO - IN L1 £		
35	INFO - IN L0	INFO - IN L0 £		

CONNECTION PIN NUMBER £ = inverted signal

TABLE 5: CHANNEL ADAPTOR BOARD CABLE CONNECTIONS

5 EXTERNAL STORE INTERFACE

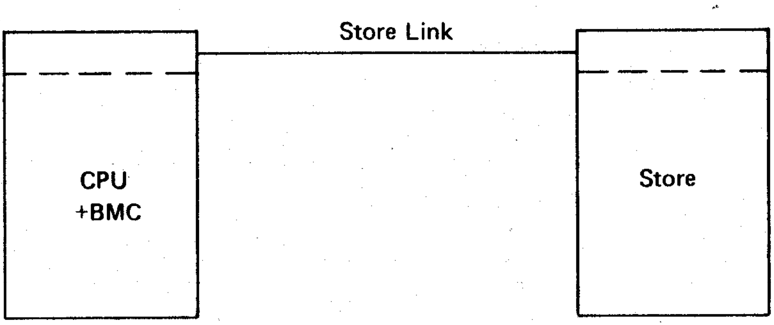
This section of the manual defines the logical operation of the Store Interface as seen from a processor and its implementation.

5.1 GENERAL INFORMATION

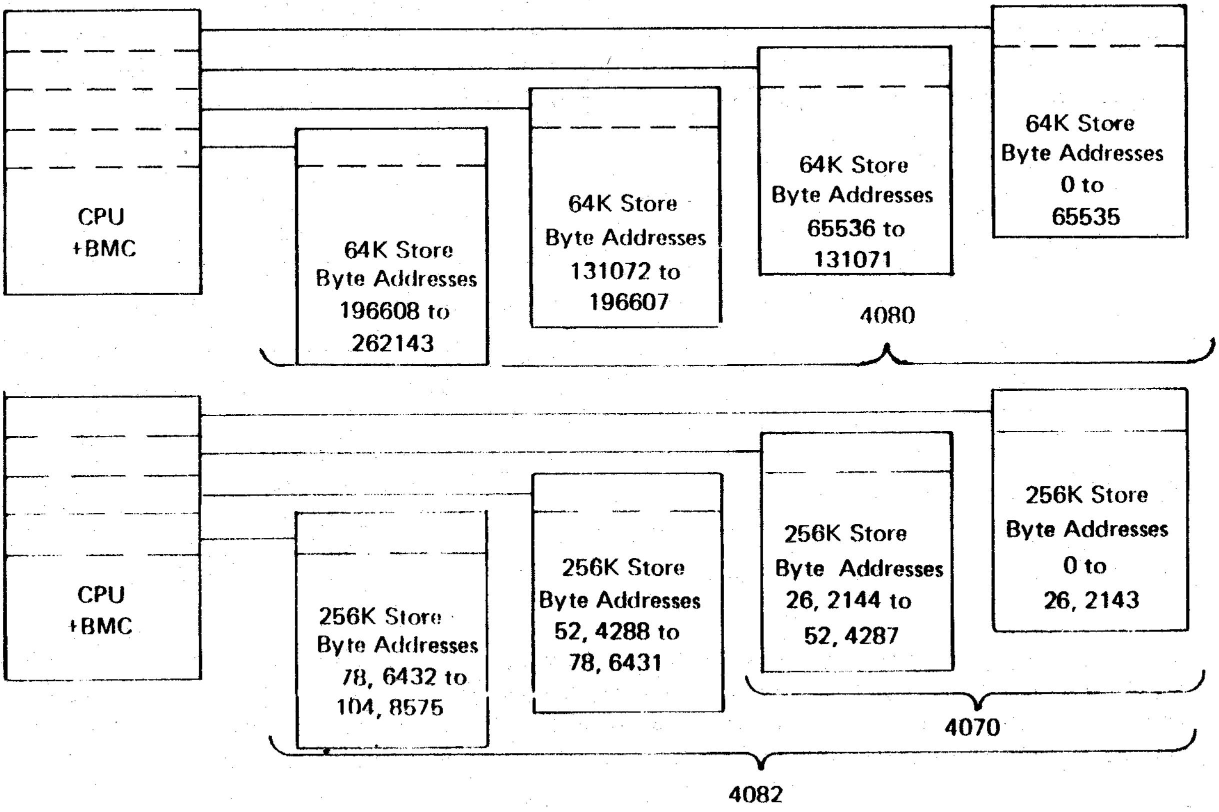
4080 store units have a capacity of either 32K or 64K bytes. The 4070 store unit can be fitted with any multiple of 32K bytes up to a maximum of 256K bytes. The capacity of the 4082 store unit is 128K or 256K bytes. Data is read or written in either bytes (8 bits) or halfwords (16 bits). Each byte has an associated parity bit which is added to the byte on writing and has its validity checked on reading. This is carried out within the store unit. A 4080 unit may be accessed by up to 4 processors on a fixed priority basis; while a 4070 or 4082 store unit may be accessed by up to 6.

5.2 SYSTEM ASPECTS

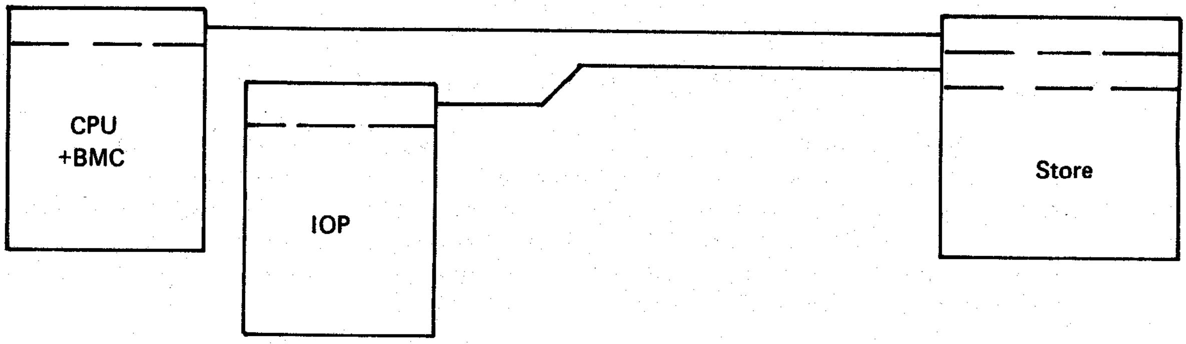
The simplest possible store configuration is shown below:



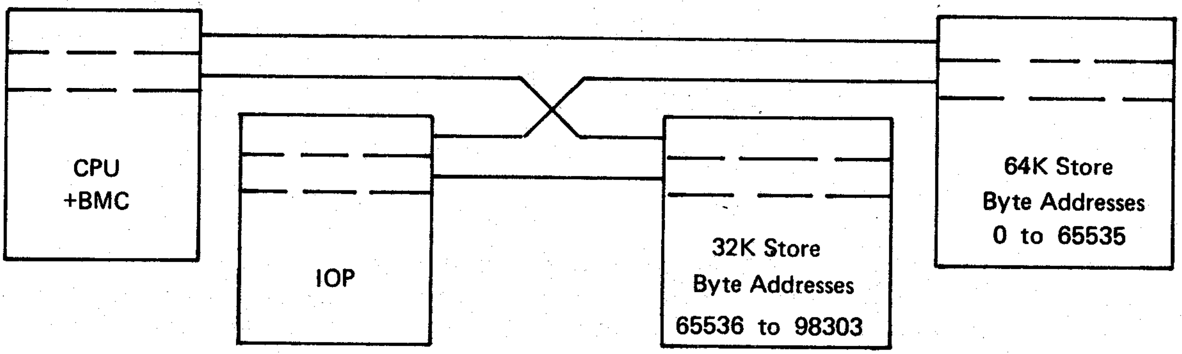
The Store Link plugs into the CPU backplane at one end and the store unit backplane at the other end. It consists of 2 logic boards and interconnecting cables. The processor contains store slots, and therefore it can be connected to store units to give access to the maximum core store size as shown below:



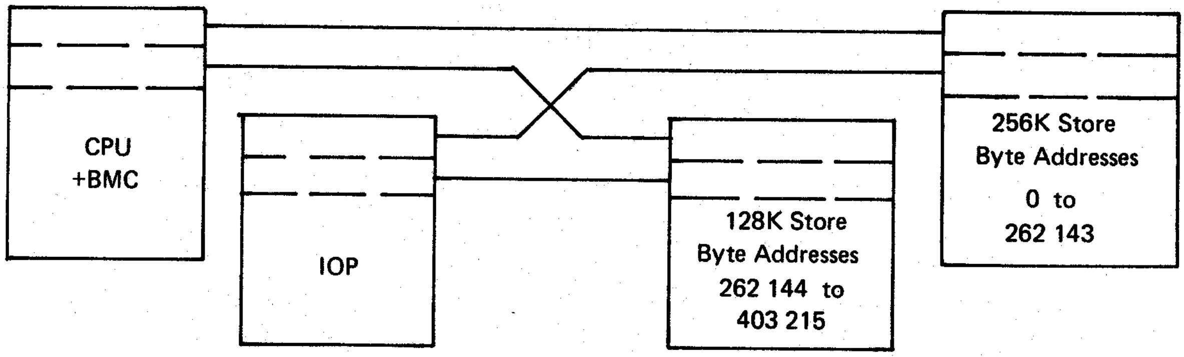
If a second processor or Input/Output Processor (IOP) is included in a system then the configuration shown below is needed.



If the system is expanded to include two stores, as shown below, both the CPU/BMC and the IOP access the store simultaneously, provided that they do not wish to access the same store unit.

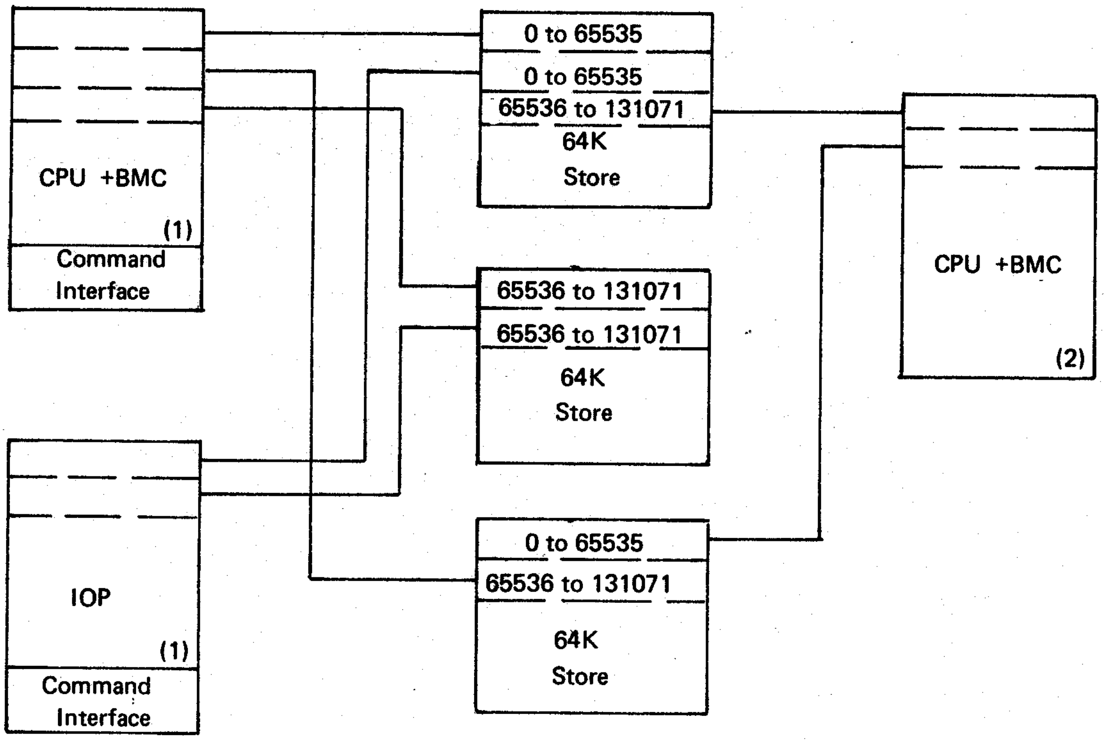


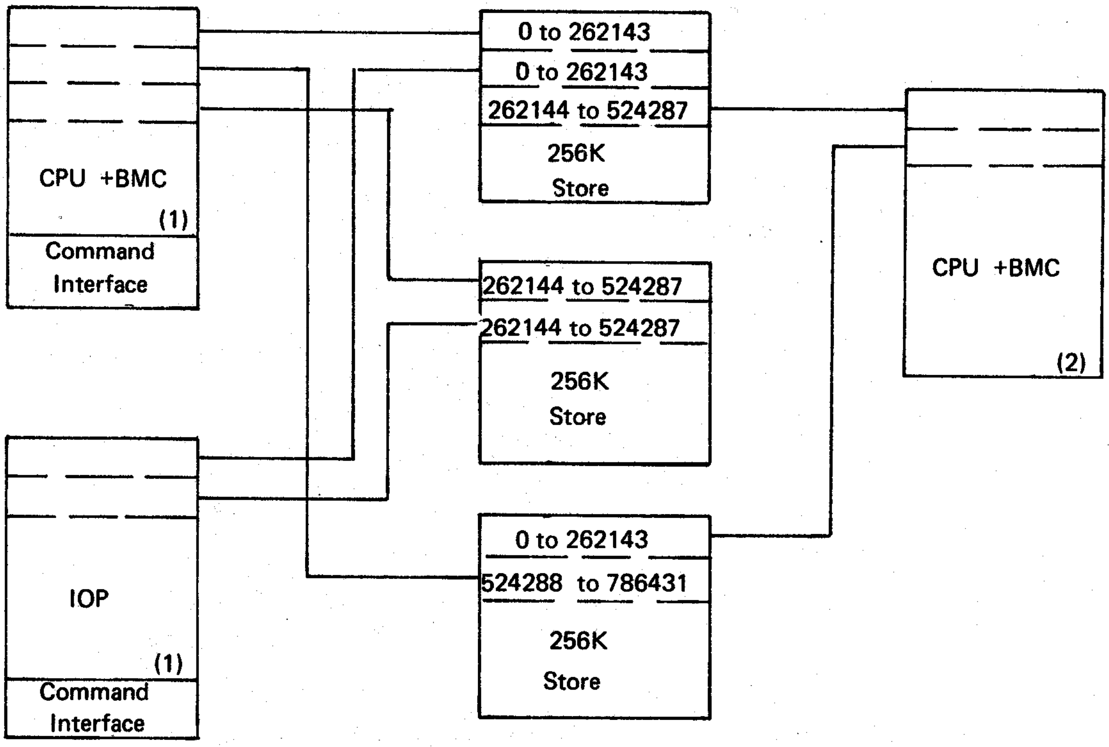
In the configuration shown above the CPU/BMC can be accessing the 64K byte store unit while the IOP is accessing the 32K byte store unit. In theory this permits a doubling of the effective store data transfer rate. In practice, since both processors must access the 64K byte store unit from time to time to access the Way Control Blocks the achievable data transfer rate is always less than the theoretical maximum.



In the configuration shown above the CPU/BMC can be accessing the 256K byte store unit while the IOP is accessing the 128K byte store unit. In theory this permits a doubling of the effective store data transfer rate. In practice, since both processors must access the 256K byte store unit from time to time to access the Way Control Blocks the achievable data transfer rate is always less than theoretical maximum.

Since the store address patching is carried out on the board at the store end of the Store Link there is no necessity for each processor to address the various store units in the same manner, as shown below: If two processor-like devices (e.g. CPU and IOP) make simultaneous store requests then, normally, the IOP is given priority over the CPU. Priority is determined by the slot position in the store unit, into which the interface link is plugged - Slot 7 (or 4) has the highest priority, slot 2 (or 1) has the lowest priority. Store starting addresses are patchable in increments of 32KB.





NOTES: The number range refers to the byte address patched for the particular store interface.

5.3 INTERFACE OPERATION

Data is transferred between the processor and the store over two highways. Each highway is 2 bytes wide. Data is transferred in either bytes or halfwords (two bytes).

Transfers from the processor to store are called output transfers, and the data is transferred over the output data highway. The two bytes are identified as DOUTM and DOUTL. Byte transfers are carried out on DOUTL. For halfword transfers DOUTL is the less significant byte.

Transfers from the store to the processor are called input transfers. The two bytes transferred over the input data highway are identified as DINM and DINL.

For both highways bit 0 is the most significant bit. The 4080 processor can address up to 256K bytes of store. To do this there are 18 address bits. These are called XADD/14,15 and ADDR/0 to 15. The 4070 processor can address up to 512K bytes of store with 19 address bits. These are called XADD13 to 15 and ADDR/0 to 15. And the 4082 processor can addresses up to 1M byte of store with 20 address bits. These are called XADD 12 to 15 and ADDR/0 to 15. For halfword transfers ADDR/15 is always zero. For byte transfers ADDR/15 determines which byte in the halfword is selected. When true the least significant byte is addressed.

Store Cycles

Each store cycle consists of two part-cycles, the address part and the data part. Each part-cycle has a separate and independent handshake of timing signals. The CPU initiates a transfer by setting REQUEST (REQ) true. This is acknowledged by the store generating ACCEPT (ACC). Similarly the data part-cycle is controlled by the DOUTC and DINC handshake. It is not necessary to complete one data part-cycle before starting the next address part-cycle. Consequently it is necessary to distinguish between consecutive part-cycles. This is done by the REQSEQ and DOUTSEQ lines. These have defined values, known as the sequence number, at times related to REQ and DOUTC and are used to associated each data-cycle with the appropriate address part-cycle.

The interface also contains various signals related to error detection, power sequencing and store interface monopolisation.

Address Part-cycle

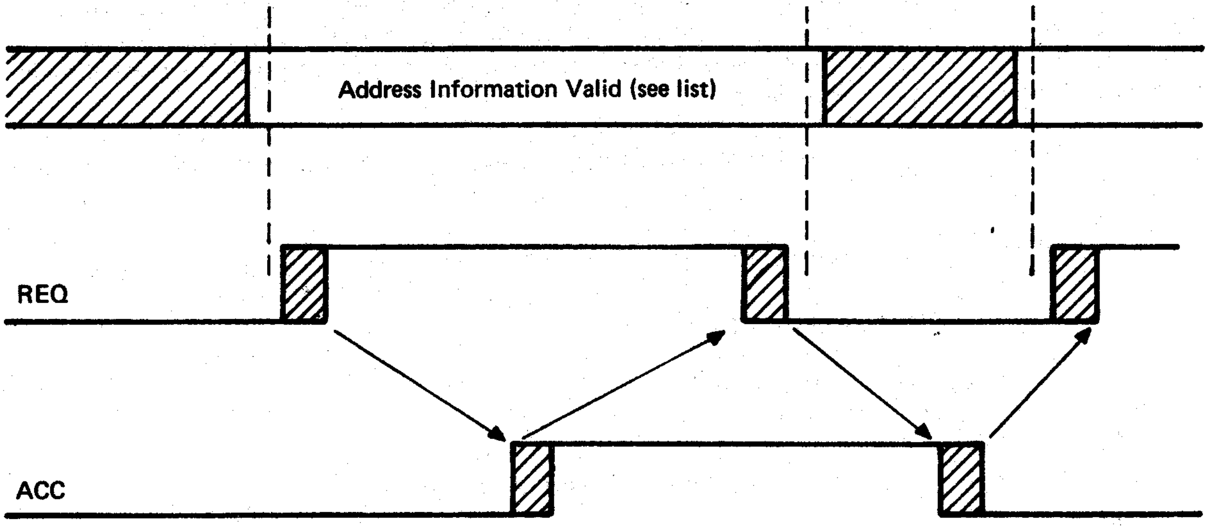
Before the processor initiates a store cycle it establishes the XADDR, ADDR, MODE, WIDTH and REQSEQ lines, thus specifying the required store location, the type of transfer (input or output) the width of the transfer (byte or half-word) and the sequence number (0 or 1). The address part-cycle is timed by REQ and ACC as shown in figure 9. REQ and ACC form a strict handshake, the following sequence being always observed:

(i) REQ becomes true
(ii) ACC becomes true
(iii) REQ becomes false
(iv) ACC becomes false

The outgoing information transmitted from the processor to the store during the address part -cycle is:

4080	4070	4082	
XADD 14, 15	XADD 13, 14, 15	XADD, 12, 13, 14, 15	Address
ADDR 0 to 15	ADDR 0 to 15	ADDR 0 to 15	
Width	Width	Width	0 for byte, 1 for halfword
MODE	MODE	MODE	0 for read, 1 for write
REQSEQ	REQSEQ	REQSEQ	Alternately 0 and 1 on consecutive cycles

This information is held valid by the processor while REQ is true.



	4080	4070	4082
Address Information is:	ADDR/0–15	ADDR/0–15	ADDR/0–15
	XADD/14,15	XADD/13–15	XADD/12–15
	WIDTH	WIDTH	WIDTH
	MODE	MODE	MODE
	REQSEQ	REQSEQ	REQSEQ

Figure 9 : ADDRESS PART-CYCLE TIMING

Data Part-cycle

The control signal handshakes are identical for both the input and output data transfers, although there are differences in the time relationship between the signals in the two cases. The transfer of data is controlled by the DOUTC/DINC handshake. Essentially data output to the store is valid between the leading edge of DOUTC and the leading edge of DINC. Data input from the store is valid between the leading edge of DINC and the trailing edge of DOUTC.

Logic within a store unit generates a separate parity bit for each byte before the data is stored, and checks the validity of this parity bit when the data is read. If a parity error is detected the store sets the Module Parity Fail (MPF) line true. The processor acknowledges this signal and causes the store to cancel the MPF by making Received Parity Error (RPE) line true.

The relationship between the various signals associated with the data part-cycle is shown in figure 10.

DOUTC and DINC form a strict handshake and the sequence is always as follows:

DOUTC	becomes true
DINC	becomes true
DOUTC	becomes false
DINC	becomes false

The following signals are transmitted during the data part-cycle:

To the store

DOUT	L0 to L7	Not defined for
DOUT	M0 to M7	input transfers.
DOUTSEQ		Corresponds to REQSEQ
RPE		Only if MPF is set true.

From the store

DIN	L0 to L7	Not defined for
DIN	M0 to M7	output transfers.
MPF		Only if parity error detected.

Association of Part-cycles

The address and data part-cycles have been described as if they are entirely independent; Using two independent handshakes enables a processor to overlap the start of one cycle with the completion of the previous cycle. This is necessary to prevent various delays in logic and cables degrading the inherent store cycle time.

To ensure that the two part-cycles do not get out of step and that data is associated with the correct address, sequence controls are provided for address and data. The store remembers the value of the request sequence (REQSEQ) at the initiation of the cycle and only initiates the data part-cycle if the data out control sequence (DOUTSEQ) has the same value.

REQSEQ and DOUTSEQ are set to zero when the interface is reset and switch from 0 to 1 at the end of an even numbered cycle and 1 to 0 at the end of an odd numbered cycle. Figure 11 that follows shows the time relationship between the two part-cycles. For clarity the figure shows REQ and DOUTC with the appropriate sequence number rather than as separate waveforms. At the processor REQSEQ is changed when REQ is reset and DOUTSEQ is changed when DOUTC is reset.

At the interface cable level the sequence signals are combined with control signals producing:

REQ/0 and REQ/1
and DOUTC/0 and DOUTC/1

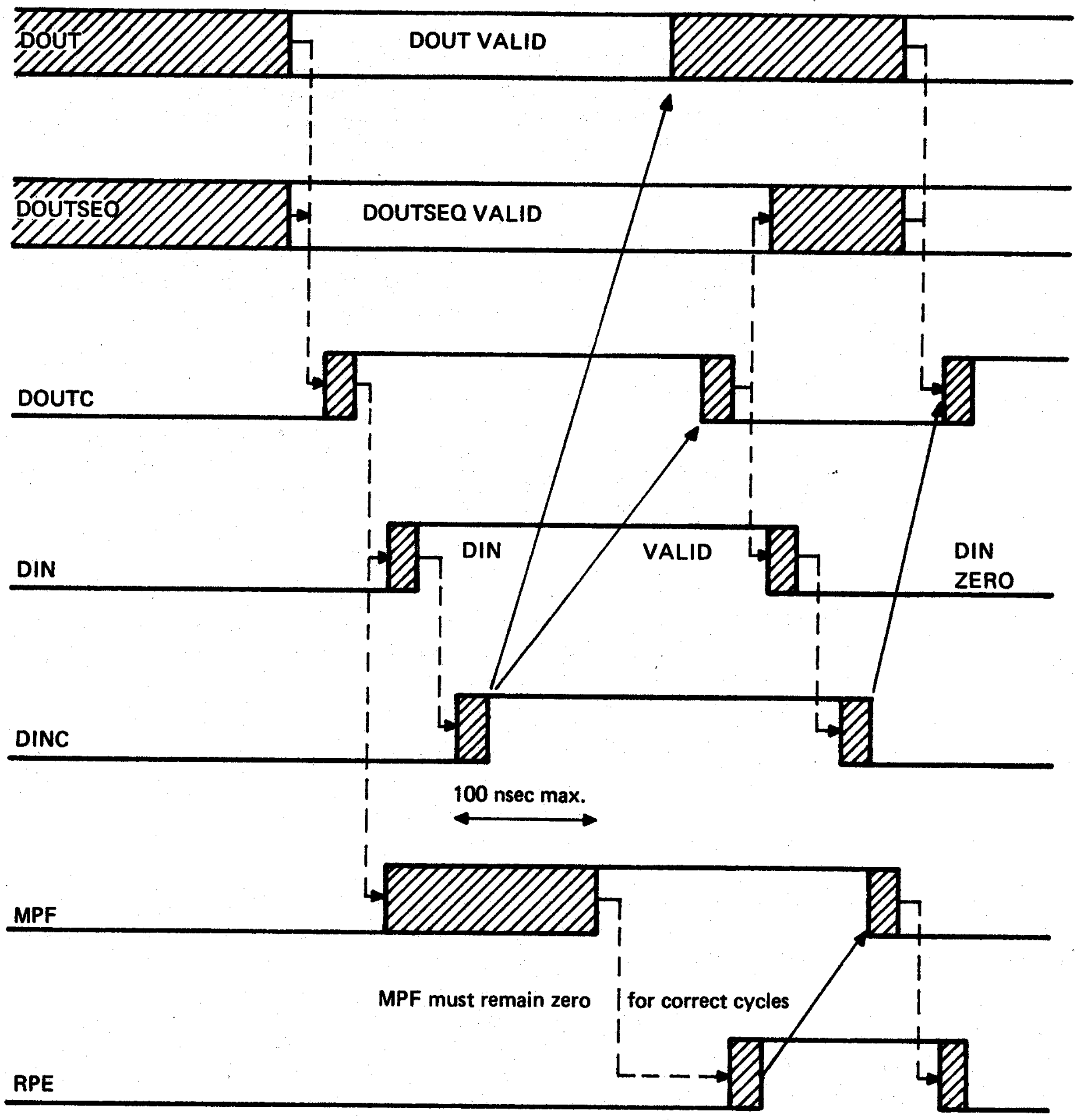


Figure 10 DATA PART-CYCLE TIMING

Relative Timing of Address and Data Part-cycles (see figure 11)

Note that all timings are defined at logic level, i.e. at the transmitter inputs and the receiver outputs in the processor.

(i) A – B

Normally REQ goes true before DOUTC because the selection of a store module and defining the type of cycle takes longer than the data part-cycle. The CPU has been implemented with a minimum of 100 ns between REQ and DOUTC. The store cycle time is increased if A to B is greater than 300 ns.

(ii) C – D

In receiving ACC, the processor takes REQ false. Address data may change 25 ns after REQ goes false. ACC goes false 85 - 160 ns after REQ goes false. The address part-cycle is now complete. REQSEQ is changed for the next cycle.

(iii) E – F

Provided DOUTC has gone true for the previous cycle, the processor may initiate another cycle. The processor must wait for DOUTC to prevent the address part-cycle from getting too far ahead of the data part-cycle. This interlock is not required for the first store cycle after a reset.

(iv) G – H

The data handshake initiated at B completes the first cycle and DOUTSEQ is changed. DINC going false enables DOUTC to be set true for the second cycle which in turn allows REQ to go true for the third cycle (H -J)

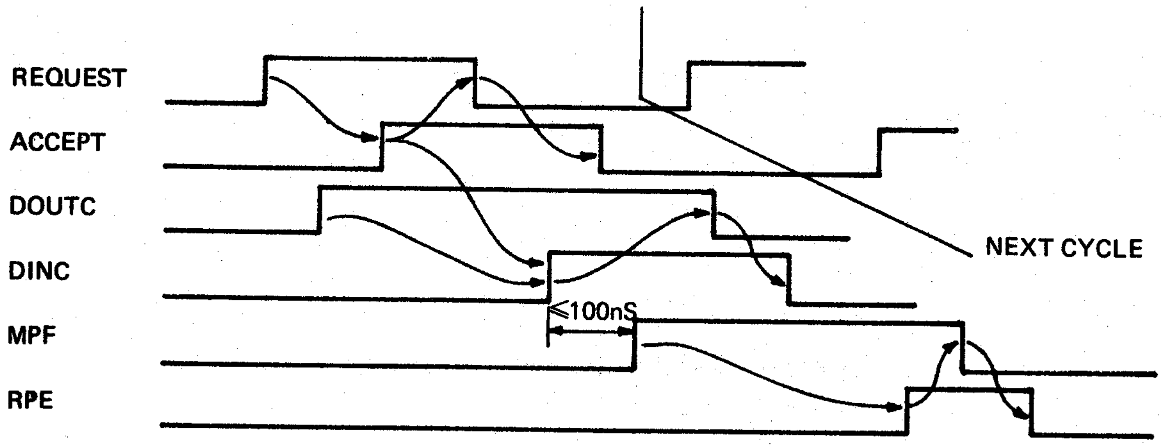
If it is acceptable to run the store slower than its nominal cycle time, it is possible to ignore the sequence signals. Then it is necessary to prevent REQ going true until DOUTC has gone false from the previous cycle (i.e. operating the part-cycles without any overlap).

Error Handling

Parity check failures detected during READ operations are signalled to the processor accessing the store by means of the MPF signals, and are acknowledged by the processor using the RPE signal. The timing of those signals is illustrated in the figure below.

Each store unit signals parity failure to the processor which was accessing the store at the time the error was detected by setting its MPF true within 100 nsecs of its setting DINC true to indicate the availability to the processor of the data read from the store. MPF is held true until acknowledged by the processor setting RPE true. At this time, the store unit resets its MPF signal (to false) and the processor resets RPE (to false).

In the CPU the MPF signals from the store units are star connected so that there can be an indication as to which unit is signalling.



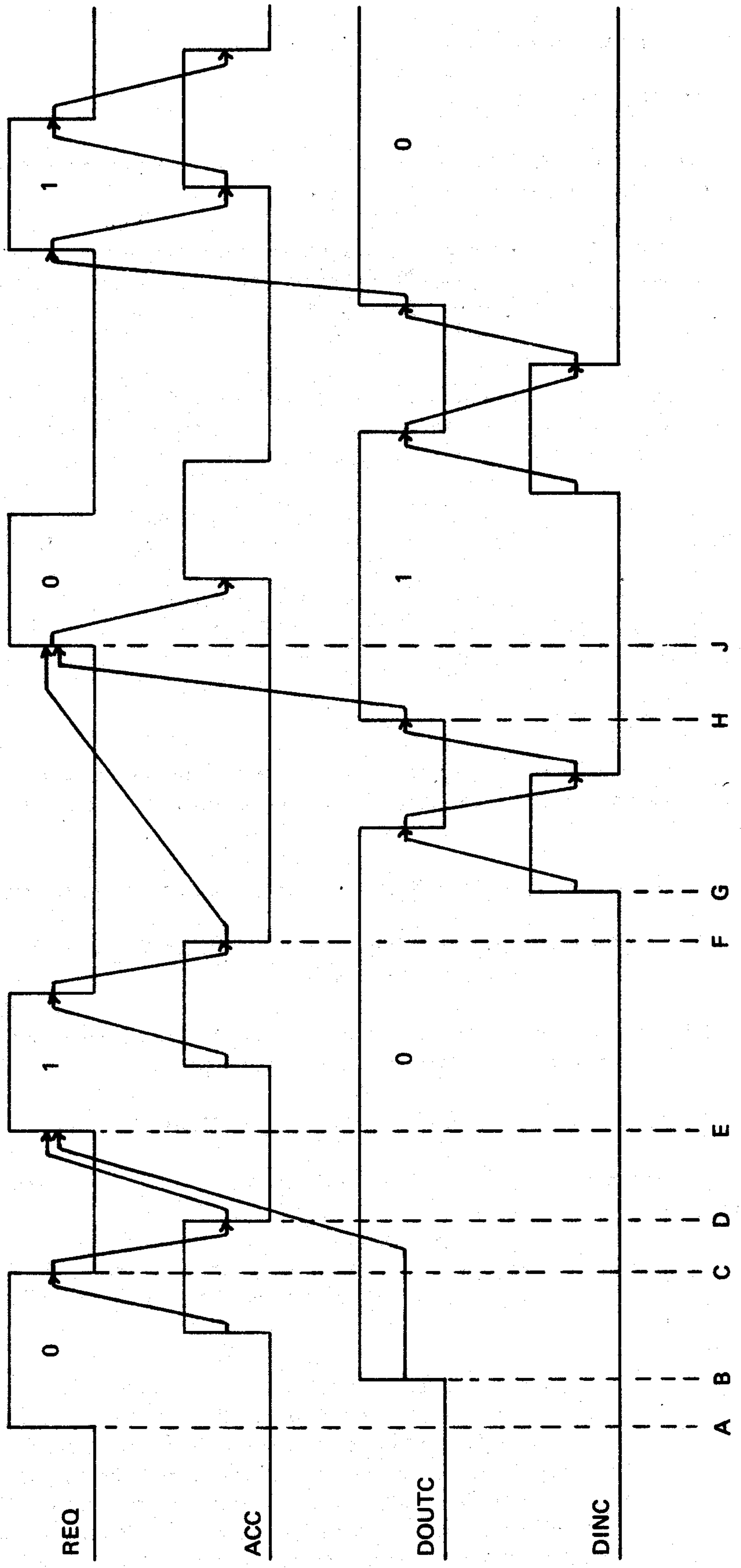


Figure 11: RELATIVE TIMING OF ADDRESS & DATA PART-CYCLES

Interlock System

INTERLOCK

INTERLOCK is a signal transmitted by a store unit to all processors connected to it and is part of the standard power interlock system.

INTERLOCK is held falseby a store unit if its power rails are not established and consequently interface signals generated by it and its response to interface signals generated by a processor are unpredictable. It is held true by a store unit if its power rails are established and correct operation under all circumstances can be guaranteed.

During power up sequencing following the switch on of a store unit INTERLOCK is held false until all power rails are established and correct operation can be guaranteed. At such time INTERLOCK goes true, and stays true until the power down sequence occurs.

During the power down sequencing of a store unit, it takes INTERLOCK false at least 1 ms before its internal power rails have deteriorated to the extent that correct operation cannot be guaranteed.

The method of transmitting INTERLOCK to the processor is such that the processor receives INTERLOCK asfalsewhile the store unit generating the signal is being up or down sequenced. It is also required that the INTERLOCK signal from an unpowered store unit is received by a processor as false.

STORE ACTIVE

The STORE ACTIVE signal transmitted to the store unit by each processor connected to it fulfils a dual function; it is part of the standard power interlock system, and is also used to transmit system resets to the store module.

STORE ACTIVE is held false byeach processor connected to a store unit under two circumstances:

(i) The DC power rails of the processor are not established and consequently interface signals generated by it are unpredictable,

(ii) A system malfunction (interface timeout etc.) has occurred, and the processor wishes to place the store system in a defined reset state.

During normal operation, the processors maintain STORE ACTIVE true.

The response of a store unit to a STORE ACTIVE false signal depends on the circumstances in which it is taken false.

(i) If all processors connected to a store unit hold STORE ACTIVE false at the same time, the store unit as a whole should be placed in the quiescent RESET state.

(ii) If at least one processor is holding ACTIVE true, and a second processor takes ACTIVE false, the resulting action depends on whether the second processor was engaged in a store cycle at that time or not. If it was not engaged in a store cycle (e.g. it either was not requesting a cycle, or was requesting a cycle which has not been granted) then the store port to which that processor is connected is put into the RESET state, and any outstanding request for store cycles is cancelled. The store port ignores any further store cycle requests from that processor until it takes ACTIVE true again. If the processor was engaged in a store cycle at the time that ACTIVE was taken false, then the store port is placed in the RESET state after 2 μsec, to allow time for the cycle in progress to be completed if possible, and a reset signal of not more than 2 μsecs duration is transmitted to the storage unit to terminate any store operation that may still be uncompleted at that time. At the end of this 2 μsec period, the store unit is again free to service requests from any store port holding ACTIVE true.

Rails Correct (RC)

When all rails are correctly established, the RC signal is generated by the store power supply. It is transmitted to the processor as a high integrity signal, and is used by the store link to ensure no spurious signals are transmitted over the interface until all power rails are established.

Hold Facility

The HOLD signal may be used by a processor to secure successive store cycles on a given store unit without allowing any other processors to access that store unit between successive cycles. It is not used by the CPU.

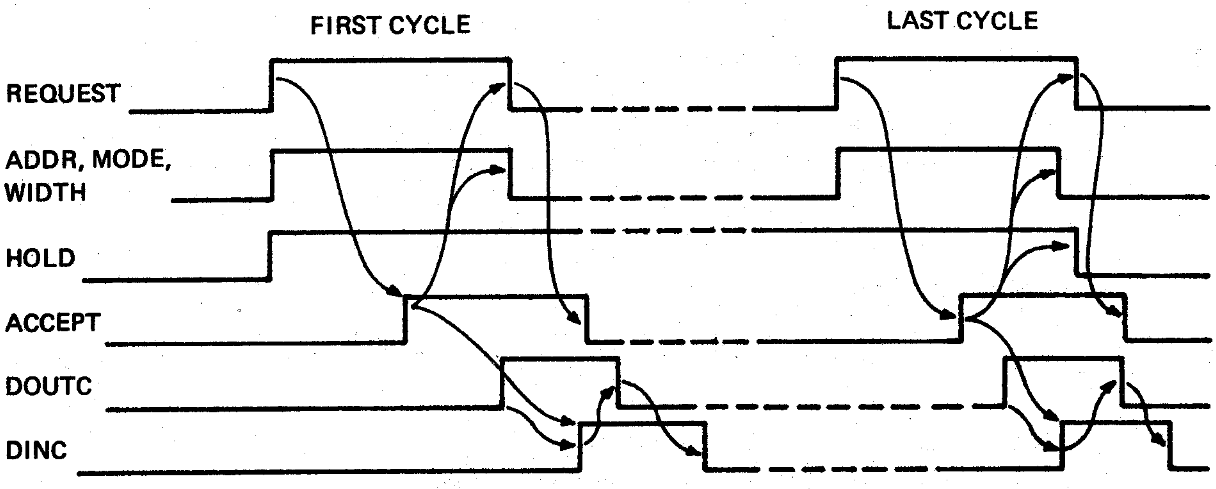
If a processor wishes to perform N uninterrupted accesses on a particular module, HOLD is used as follows:

(a) At the same time as XADDR, ADDR, MODE, and WIDTH are established (i.e. shortly before REQUEST is made true for the first cycle), HOLD is also made true on the interface.

(b) HOLD is held true on the interface after XADDR, ADDR, MODE, and WIDTH are reset, and remains true until the last cycle required.

(c) During this cycle, HOLD is reset to false at the same time as XADDR, ADDR, MODE, and WIDTH are set false (i.e. shortly after ACCEPT is received as true for the last cycle).

This is illustrated in the figure below



Summary of Signals

The following table lists the various signals, their type and direction.

The signals provided on the store interface are as follows:

XADDR/12–15 (4082)	To Store	Information	(14, 15 = 4080:13–15 = 4070)
ADDR/0-15	To Store	Information	
WIDTH	To Store	Information	
MODE	To Store	Information	
HOLD	To Store	Information	(not generated by CPU)
DOUTL/0-7	To Store	Information	
DOUTM/0-7	To Store	Information	
DINL/0-7	To Processor	Information	
DINM/0-7	To Processor	Information	
REQ/0	To Store	Timing	
REQ/1	To Store	Timing	
ACCEPT	To Processor	Timing	
DOUTC	To Store	Timing	
DINC	To Processor	Timing	
MPF/0-3	To Processor	Timing	
RPE	To Store	Timing	
STORE ACTIVE	To Store	Auxiliary	
INTERLOCK	To Processor	Auxiliary	
RC	From Store	Auxiliary	

5.4 METHOD OF CONNECTION

Generally it is convenient to connect, to a standard board in a store unit via a cable to the user's equipment. This section gives the circuit details and the pin allocation of the various signals.

Signal Convention

All signals on the interface are transmitted at Standard TTL levels. All signals are in negative logic convention. Thus:

A signal < 0.5V ('low' level) corresponds to logic '1' (true)
A signal > 2.5V ('high' level) corresponds to logic '0' (false)

Transmitters

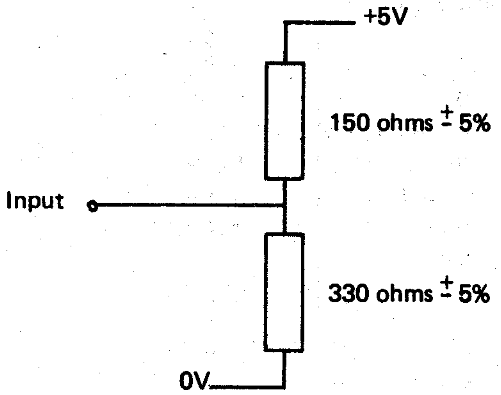
These are TTL integrated circuits, either open collector SN7438 or SN75453B, all timings previously defined assume the use of the SN75453B.

Receivers

Signals may be received by any single load input 74 or 74H integrated circuit. The signal input should not be connected to more than one input load.

Termination

The signal termination network shown below is used to terminate each incoming line at a processor. All outgoing lines are terminated on the store port card.



The termination network should be mounted as close to the receiver as possible. The 0V line associated with each signal input should run parallel and close to the signal line and be connected to the logic 0V close to the receiver. Similarly the output from a transmitter has an 0V line that should run parallel and close to the signal output and be connected to the logic 0V near the transmitter. It may not be practical to run all 0V lines back to their source in which case preference should be given to these control signals:

RC, DOUTC, ACC, INT, MPF, DINC, REQ, ST ACTIVE

Cable

Two ribbon cables connect to EC4 and EC6 of the store link board in the CPU rack. 70 way BICC-Burndy PB4DD 35-A-02-02 edge connectors are provided at each end of the cable. The signal allocation to the EC4 and EC6 pins are shown in Table 6.

5.5 COMPATIBLE WITH 2050 COMPUTER

The external store access interface of the GEC 2050 computer is an 8 bit wide variant of the 4000 Series external store interface, it is possible for external equipment to be designed for use with either interface – for details consult GEC Computers Limited.

	Connector EC4		Connector EC6	
	B	A	B	A
1	0V	ACCEPT£	0V	DINC£
2	INT£	0V	0V	0V
3	0V	RC£	0V	REQ£/1
4	XADDR£/12 (4082)	0V	DOUTC£/1	0V
5	0V	XADDR£/13(4070,4082)	0V	DOUTC£/0
6		0V	REQ£/0	0V
7	0V	DINM£/7	0V	ST ACTIVE£
8	DINM£/6	0V	WIDTH£	0V
9	0V	DINM£/5	0V	HOLD£
10	DINM£/4	0V	MODE£	0V
11	0V	0V	0V	ADDR£/15
12	DINM£/2	0V	ADDR£/14	0V
13	0V	DINM£/1	0V	ADDR£/13
14	DINM£/0	0V	ADDR£/12	0V
15	0V	DINM£/3	0V	ADDR£/11
16	MPF£	0V	ADDR£/10	0
17	0V	DOUTL£/7	0V	ADDR£/9
18	KEYWAY		KEYWAY	
19	0V	DOUTL£/5	0V	ADDR£/8
20	DOUTL£/6	0V	DINL£/7	0V
21	0V	DOUTL£/3	0V	DINL£/5
22	DOUTL£/4	0V	DINL£/6	0V
23	0V	DOUTL£/1	0V	DINL£/3
24	DOUTL£/2	0V	DINL£/4	0V
25	0V	DOUTL£/0	0V	DINL£/1
26	XADDR£/15	0V	DINL£/2	0V
27	0V	XADDR£/14	0V	DINL£/0
28	DOUTM£/7	0V	ADDR£/7	0V
29	0V	DOUTM£/6	0V	ADDR£/6
30	DOUTM£/5	0V	ADDR£/5	0V
31	0V	DOUTM£/4	0V	ADDR£/4
32	DOUTM£/3	0V	ADDR£/3	0V
33	0V	DOUTM£/2	0V	ADDR£/2
34	DOUTM£/1	0V	ADDR£/1	0V
35	0V	DOUTM£/0	0V	ADDR£/0

Connector Pin Allocation

Table 6: EXTERNAL STORE INTERFACE PIN ALLOCATION

6 COMMAND INTERFACE (CI)

6.1 SYSTEM ASPECTS

The Command Interface is the interface over which overall control of Input/Output Processors (IOP's) is carried out.

Function

The Command Interface comprises control and timing signals which are used in conjunction with fixed areas of main store and are used for passing data and control information between the CPU and the IOP's.

The functions which may be performed by the Command Interface are as follows:

(a) Input of Data from an IOP to the CPU

(b) Output of Data to an IOP from the CPU

(c) Sending of Interrupts to the CPU from an IOP

(d) Acknowledgement of Interrupts by the CPU

(e) Interlocking of CPU and IOP's during critical operations.

There are two types of store area used by each IOP. One is the I/O Control Block which occupies a fixed position in store and which is used by all IOP's for holding the data and address associated with certain Command Interface cycles. It is also used for holding the addresses of the appropriate Way Control Blocks to be referred to when an interrupt is signalled by an IOP.

The other types of store area used by IOP's are Way Control Blocks (WCB's). These are used for communicating control information relating to I/O operations between the CPU and the IOP.

6.2 OPERATION

Signals in the Interface

The Command Interface comprises the following signals:

Signal	Direction	Type
CI SELECT	From CPU	Timing
CI ADDR 0–2	From CPU	Information
CI MODE 0–1	From CPU	Information
CI RESPONSE	To CPU	Timing
CI INTERRUPT 0–7	To CPU	Timing
CI ACTIVE	From CPU	Control
CI INTERLOCK	To CPU	Control

The usage of these signals is as follows:

CI SELECT

CI SELECT is used by the CPU to initiate a command interface cycle. It is made true only when the ADDR and MODE information has been established (see Figure 12).

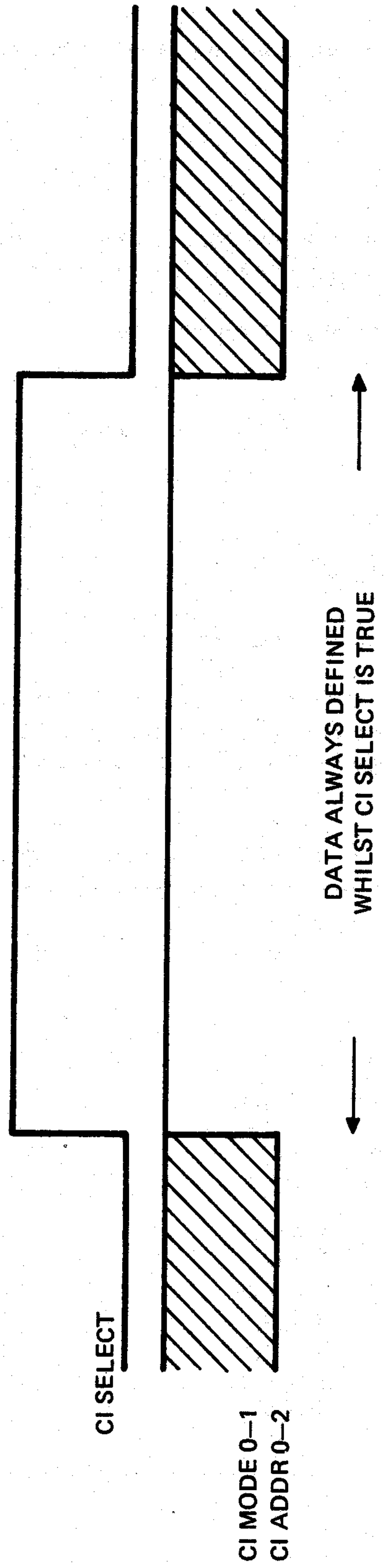


Figure 12: SELECT - DATA TIMING

CI ADDR 0-2

Three address lines are used to indicate which IOP is to respond to an interface cycle. The address lines carry the number of the IOP involved in a Command Interface cycle.

The address lines are defined only during the time CI SELECT is true.

CI MODE 0-1

Two MODE lines are used to determine the operation required to the IOP. The coding of these bits is as follows:

MODE 0 1	Operation
0 0	Reset Interrupt
0 1	Wait for CPU
1 0	Data Input
1 1	Data Output

Reset Interrupt	-	Reset interrupt line immediately
Wait for CPU	-	The CPU is about to perform a critical operation and the IOP must if necessary refrain from performing any operation which might result in undesirable asynchronous effects.
Data Input	-	The IOP is to store a halfword of data in the I/O control block.
Data Output	-	The IOP is to read a halfword of data or command from the I/O control block.

CI RESPONSE

CI RESPONSE is the timing signal used by IOPs to respond to a SELECT signal. Timing details are given under 'Timing of Signals'.

CI INTERRUPT 0-7

Each IOP is connected to one of the eight interrupt lines. The address used by the device and its interrupt number is the same.

CI ACTIVE

CI ACTIVE is held true whenever the CPU is in an operable condition. An IOP must not perform any transfer or store cycle if CI ACTIVE is false.

CI INTERLOCK

CI INTERLOCK is an 'AND bus' signal which may be held false by any IOP which is not in an operable condition. CI INTERLOCK will therefore be taken true only when all IOPs are operable.

Timing of Signals

Figure 13 shows the timing and handshake procedure for the main signals in the interface.

The sequence of events is as follows:

(1) The CPU loads the I/O Control Block, if necessary, with the data appropriate to the transfer. It then sets CI MODE 0-1 to indicate the type of cycle required, sets CI ADDR 0-2 to the address of the address of the IOP and then makes CI SELECT true.

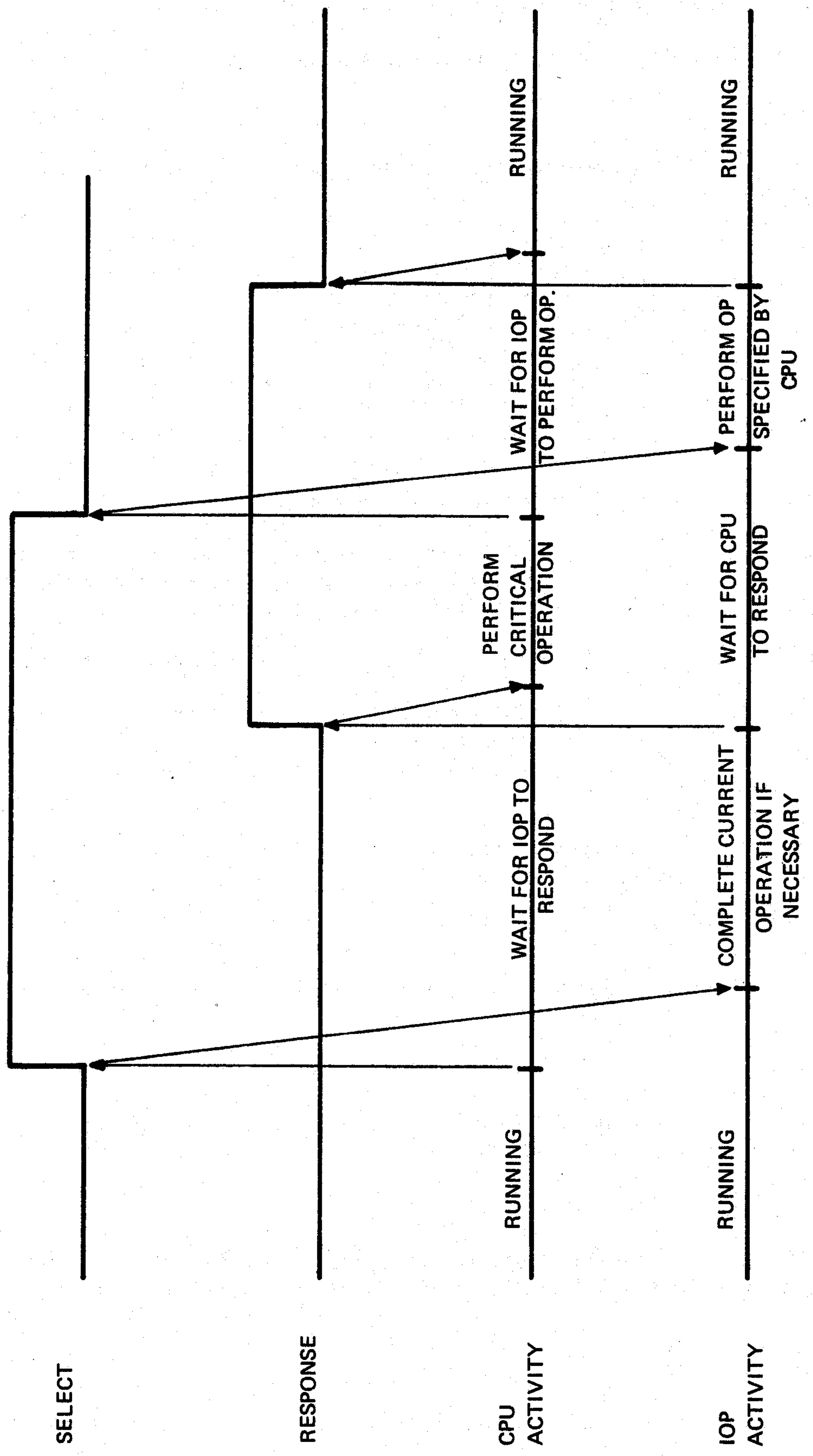

SELECT
RESPONSE
CPU ACTIVITY
IOP ACTIVITY
RUNNING
WAIT FOR IOP TO RESPOND
PERFORM CRITICAL OPERATION
WAIT FOR IOP TO PERFORM OP.
RUNNING
RUNNING
COMPLETE CURRENT OPERATION IF NECESSARY
WAIT FOR CPU TO RESPOND
PERFORM OP SPECIFIED BY CPU
RUNNING

Figure 13: BASIC COMMAND INTERFACE CYCLE

(2) On recognising its own address on the CI ADDR lines and having received CI SELECT as true, the IOP decodes the CI MODE lines to determine the type of transfer required. When it is ready to respond to the CPU (and in the case of the BMC this may not be until the current normal interface cycle is complete), the IOP responds by making CI RESPONSE true.

If the processor wishes to load a WCB it does so at this time. The IOP should not perform any operation which might result in undesirable effects arising from asynchronous conflicts whilst SELECT and RESPONSE are both true. This is achieved in the case of the BMC by holding it in a quiescent halted state during this period.

(3) When the processor is ready for the IOP to proceed it makes CI SELECT false; it may then also remove the MODE and ADDRESS information.

(4) The action taken when CI SELECT has gone false depends on the coding of the CI MODE lines:

(i) If the MODE lines specify a 'RESET INTERRUPT' cycle, the IOP will normally make its CI INTERRUPT line false, and may subsequently signal further interrupts to the CPU. If it wishes to immediately signal a further interrupt, it will leave its CI INTERRUPT line true. In any case, it now sets the CI RESPONSE line false to complete the cycle.

(ii) If the MODE lines specify a 'Wait for CPU' cycle, the IOP sets CI RESPONSE false to complete the cycle.

(iii) If the MODE lines specify a 'Data Input' cycle, the IOP obtains a halfword of data, making use of the DEVICE ADDRESS location of the IOCB to specify the source of the data if necessary. This data is then written into the DATA location of the IOCB. The action specified in (v) below is now performed.

(iv) If the MODE lines specify a 'Data Output' cycle, the IOP reads the DATA location of the IOCB to obtain the data to be output, and uses the DEVICE ADDRESS location to the IOCB specify the destination of the data if necessary. The action specified in (v) below is now performed.

(v) Following the operations specified in (iii) (iv) above the IOP writes an appropriate SIGNAL indication into the DEVICE ADDRESS location of the IOCB, to indicate to the CPU the success or failure of the cycle just performed. The IOP now terminates the cycle by making CI RESPONSE false.

Interrupts

If an IOP wishes to signal an interrupt to the CPU, it does so by making its INTERRUPT line true. If the particular IOP concerned uses more than one WCB for interrupt purposes it first loads its INTADDR location in the I/O control block with the address of an interrupt WCB.

The WCB itself contains information specifying the nature of the interrupt and the program to be run to service the interrupt. This is detailed fully in section 4.7 of Nucleus Manual.

The CPU will in due course respond to an interrupt by a 'Reset Interrupt' command interface cycle as described above.

6.3 LOGICAL IMPLEMENTATION

The Command Interface is implemented in two forms: an internal logic level interface that can be used only by the BMC and an external interface that can be used by other IOPs.

This section describes the external command interface. The interface has been implemented using a single bus connection, each successive IOP being connected in a daisy chain connection, and connected to the CPU at EC6 on the Exponent board (Brd 107). The maximum total permitted cable length is 20 ft.

Logic Convention

The External Command Interface is implemented using negative logic. A signal is true (logic 1) if it is $<+0.4$V relative to 0V and false (logic 0) if it is $>+2.5$V relative to 0V.

Transmitters

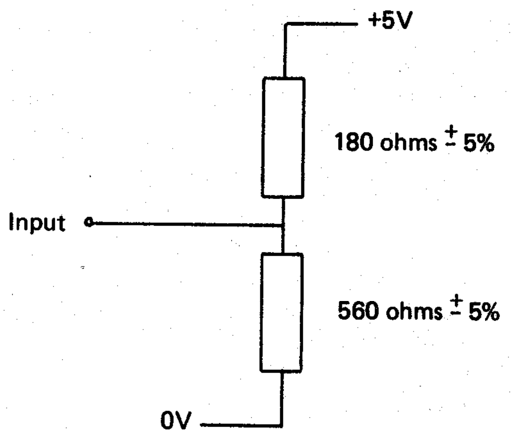
The transmitters are TTL drivers type SN7438 or SN75453B. Each incoming line is terminated at the CPU.

Receivers

Signals may be received by any single load input 74 or 74H integrated circuit. The signal input should not be connected to more than one input load.

Termination

All outgoing signals from the CPU must be terminated at the end of the cable with the resistor network shown below:



The following lines are terminated at the IOP end of the cable:

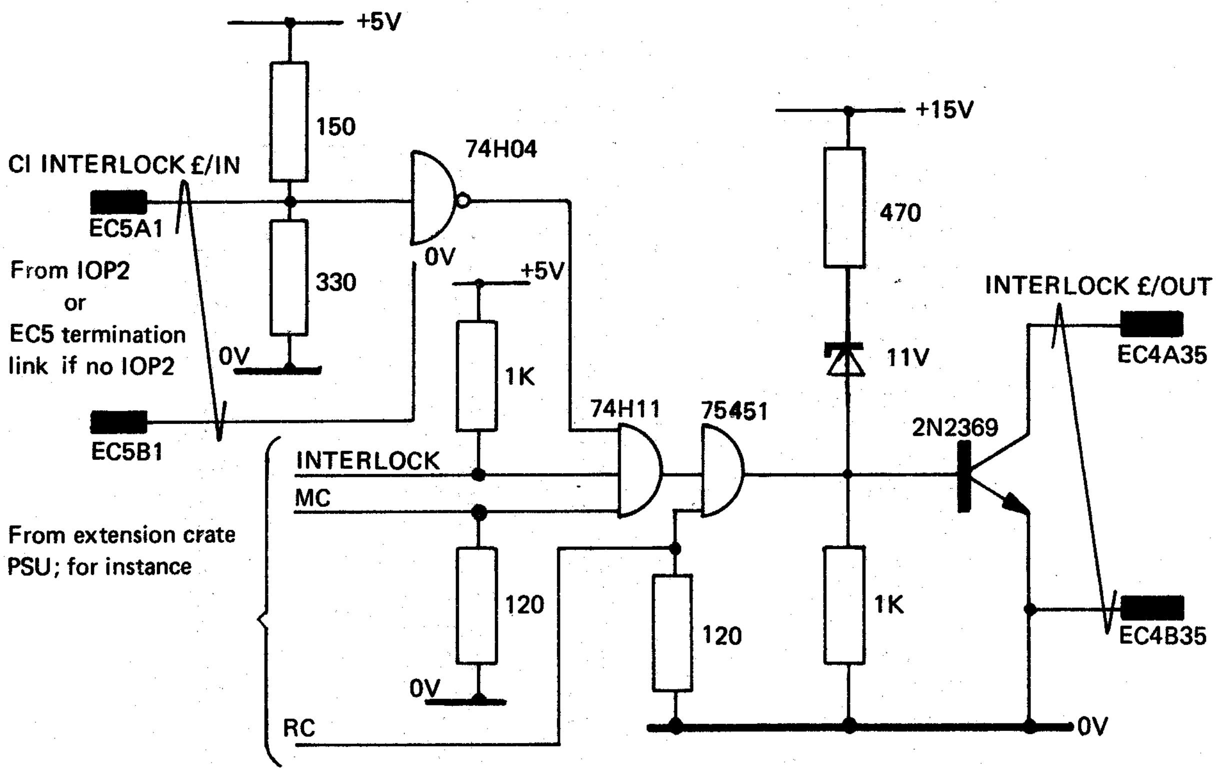
CI	SELECT
CI	ADDR 0 to 2
CI	MODE 1 0 to 1
* CI	ACTIVE

* CI ACTIVE is terminated with 150 ohms and 330 ohms instead of 180 ohms and 560 ohms.

Interlock

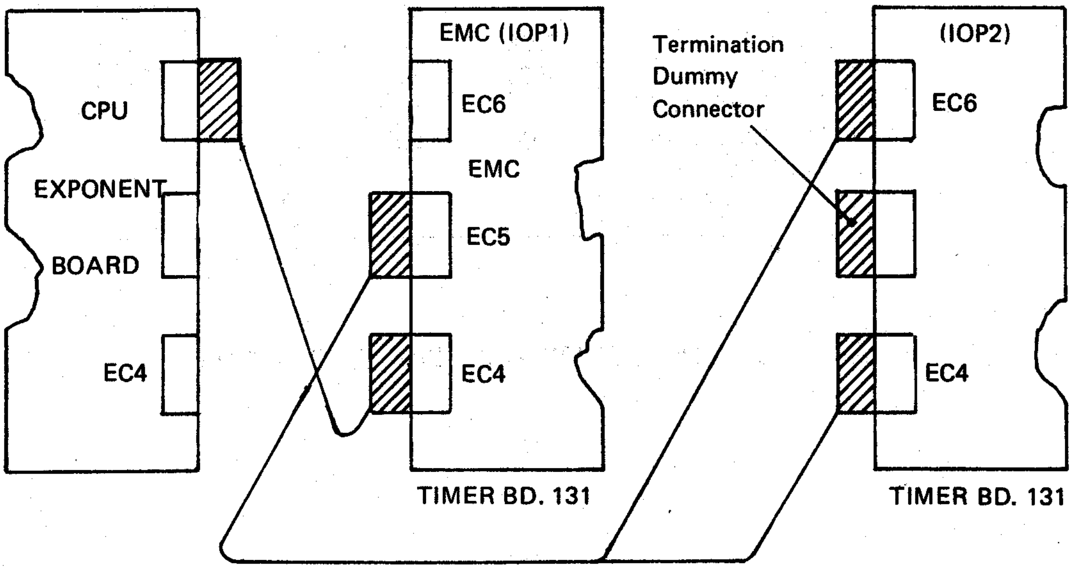
The processor receives INTERLOCK true from IOP's on the Command Interface when all peripherals and IOP's have correctly sequenced-up. It is required to preserve integrity of the logic of the INTERLOCK signal when the power rails are not established. The following figure shows how this is implemented.

INTERLOCK£/IN comes from the IOP on the bus away from the CPU. INTERLOCK£/OUT goes towards the processor. MC (Mains Correct), RC (Rails Correct) are from the IOP power unit.



Cable

The user may connect his own IOP by cable to EC5 on the CPU exponent board or on the EMC board to which the CPU is connected as shown in the figure below. If more than one external IOP is in the configuration it is recommended that standard cables be used as shown in the figure below.



Timing

As the Command Interface uses a handshake the timing is not critical. Address and mode lines are established by the CPU a minimum of 30 ns before SELECT is made true and do not change until at least 30 ns after SELECT is made false. These timings are defined at the CPU end of the cable.

Command Interface Signals

The signal list and their pin allocation for the EMC is given in table 7.

Signal	To CPU		To IOP	
CI SELECT£	EC6	A6	EC6	A30
CI SELECT£/0V	EC6	B6		B30
CI ACTIVE£	EC4	A34		A2
CI ACTIVE£/0V	EC4	B34		B2
CI ADDR0£	EC6	A5		A31
CI ADDR0£/0V	EC6	B5		B31
CI ADDR1£	EC6	A7		A29
CI ADDR1£/0V	EC6	B7		B29
CI ADDR2£	EC6	A8		A28
CI ADDR2£/0V	EC6	B8		B28
CI MODE0£	EC6	A1		A35
CI MODE0£/0V	EC6	B1		B35
CI MODE1£	EC6	A2		A34
CI MODE1£/0V	EC6	B2		B34
CI INT£/7	EC4	A33		A3
CI INT£0V/7	EC4	B33		B3
CI INT£/6	EC4	A27		B9
CI INT£0V/6	EC4	B27		A9
CI INT£/5	EC4	A30		B6
CI INT£0V/5	EC4	B30		A6
CI INT£/4	EC4	A31		B5
CI INT£0V/4	EC4	B31		A5
CI INT£/3	EC4	A32		B4
CI INT£0V/3	EC4	B32		A4
CI INT£/2	EC4	A25		B11
CI INT£0V/2	EC4	B25		A11
CI INT£/1	EC4	A26		B10
CI INT£0V/1	EC4	B26		A10
CI RESP£	EC6	A3		A33
CI RESP£/0V	EC6	B3		B33
CI INTERLOCK£/OUT	EC4	A35		
CI INTERLOCK£/OUT 0V	EC4	B25		
CI INTERLOCK£/IN				A1
CI INTERLOCK£/IN 0V			EC6	B1

Table 7: COMMAND INTERFACE PIN ALLOCATION