AN INTERFACE FOR A MICROPROCESSOR CONTROLLED FREQUENCY RESPONSE ANALYSER (FRA)

R. Ramnarine
Faculty of Engineering
University of the West Indies
St. Augustine, Republic of Trinidad & Tobago

Summary

An interface was designed in order to link a microprocessor controlled FRA to various peripheral devices using the IEEE 488 protocol. The interface Hardware centered around two 8255's general purpose programmable I/O (Input/-Output) device for parallel Input/Output. The software routines were written in the appropriate Intel Assembly language to match the INTEL 8080 microprocessor used, and developed on the MDS 231 development system. The two 8255's were configured into eight ports; four 8—bit and four 4—bit. Control, status and data bytes are read from these ports or written to them. The interface conforms to the IEEE standard for computer interfacing.

1. INTRODUCTION

The interface described here is designed to link the microprocessor controlled Frequency Response Analyser (FRA) to different peripheral devices for the parallel communication of data (Figure la, lb). The problem of the hard-ware/software compromise was considered and a convenient trade—off point was chosen on the grounds of technical and economic factors.

Generally, in electronics, any logic system may be considered as an interface. In other words an interface may be a collection of circuits, interposed between two different devices which translates, and/or buffers signals from one format to another so that the output from one device can be suitable as an input to the other device and vice versa. Inputs are signals that represent real time events or measurements. The outputs are normally digitized version of analogue inputs or control signals. These logic systems rarely operate without an input source and an output device and are used mostly in computer based systems to transmit data in both directions. Normally, storage, computation and sequencing are handled by the computer, whilst multiplexing and line driving are left to the interface logic. Thus it can be seen that before the actual interface design begins, three basic aspects must be identified viz:

- all inputs
- system function
- all outputs

The hardware design then, of necessity falls into the following categories:

- Microprocessor Input/Output Structure
- Instrument control and Output characteristics
- Interface logic and Transmission characteristics
- Construction of Interface Hardware
- Relevant Software

2. MICROPROCESSOR INPUT/OUTPUT STRUCTURE

The microprocessor architecture and full description of the ancillary hardware and available software are well covered in the literature (1, 2, 3).

3. FRA CONTROL AND OUTPUT CHARACTERISTICS

The following routines control the FRA:

Manuscript received on 21st April, 1986.

- I/P parameter processing
- Sweep sequence control
- Measurement control
- Measurement result processing
- Data output processing

Details of these routines may be found in Reference 4.

The FRA is controlled externally by the IEEE 488 Interface (6, 7, 8, 9, 10) by the straight replacement of the keyboard input i.e. the sequence of operations and codes are identical to that used at the keyboard. Output from the FRA to the controlling device is restricted to measurement results.

4. INTERFACE HARDWARE

The interface hardware (Figure 2) centres around the Intel 8255 chip (Figure 3). This is a general purpose programmable I/O device designed for use with the Intel 8—bit microprocessor. The 24 I/O pins could be individually programmed in two groups of twelve and used in three major modes of operation (Mode 0, Mode 1, Mode 2). In mode 0, which is used in this interface each group of 12 I/O pins may be programmed in sets of 4 for I/P or O/P. The basic operation is shown in Table 1 (5). The functional configuration of each port (Table 2) is programmed by the systems software. Here the microprocessor outputs to the 8255 a control word which contains information such as "mode", "bit set" "bit reset" etc., in order to initialize the functional configuration (Figure 4) of the 8255 chip.

TABLE 1: 8255 OPERATION SUMMARY

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	I/P Operation READ
0	0	0	1	0	Port A → Data Bus
0	1	0	1	0	Port B → Data Bus
1	0	0	1	0	Port C → Data Bus
					O/P Operation WRITE
0	0	1	0	0	Data Bus → Port A
0	1	1	0	0	Data Bus → Port B
1	0	1	0	0	Data Bus → Port C
1	1	1	0	0	Data Bus → Control
					Disable Function
X	X	X	X	1	Data Bus → Tri-state
1	1	0	1	0	Illegal Condition

The Read/Write control logic, on the receipt of control words (Table 1) issues commands to the control blocks (Groups A and B) from which the associated ports are controlled. Control group A is associated with Port A and Port C upper (C4—C7), whereas control group B is associated with Port B and Port C lower (C0—C3). One can only write into the control word register, as no read operation is allowed. The various ports are selected by setting the appropriate bits (Figure 5).

Table 2 : IEEE INTERFACE REGISTERS

	PORT ADDRESS	MNEMONIC	TYPE*	CONTENTS
INTERFACE REGISTERS	150	IMCR	R/W	Interface Management Bus Control
	151	ISR	R	Interface Status
	152	DTR	W	Data Byte Transfer Control
	153	CWR 1	W	Control Word 1
	154	DIOR	R/W	Data
	155	DSAR	R	Device Selection Address
	156	DCTR	W	Device Control
	157	CWR 2	W	Control Word 2

* R indicates a READ ONLY REGISTER : R/W indicates a READ AND WRITE REGISTER

W indicates a WRITE ONLY REGISTER

5. INTERFACE SOFTWARE

5.1 General

This interface conforms to the International Electrotechnical Commission Specification Interface System (6) for a programmable measuring apparatus (IEEE 488). Fig. 6 illustrates the interface capabilities and bus structure, Control bytes, status bytes and data bytes are written to or from the eight ports assigned to the interface. Each port exists physically in one of the two 8255 chips (Programmable Peripheral Interfaces) used in the hardware configuration specified by bit allocation.

5.2 Parallel Interface Handler

5.2.1 Introduction

The parallel interface handler which includes software implemented HP—IB functions (10), is designed by aiming at providing FRA, with the capability of accepting remote messages sent by a talker device, or to send results of the measurement to the listener devices on the interface bus.

This interface allows the FRA to be either in "talk only" mode or in "addressable" mode. In talk only mode FRA can output results to any listener on the bus immediately after they are available, and without requiring any bus controller. In this mode when the FRA is switched on the initialisation program sets the "talker" flag and assigns the FRA as the talker. In addressable mode the FRA does not output results until it is assigned as talker by the bus controller. The modes are selected by a switch at the back panel of the FRA and the position of this switch is read by a routine to put the FRA to an initial state. If there is a bus controller then no matter which mode is selected the FRA could be assigned, either as a talker, or as a listener, or as a passive device on the bus (e.g. it is neither talker nor listener). The FRA cannot be a talker and a listener at the same time.

A status byte can be sent over the interface if the bus controller issues a serial poll, although parallel poll function

is not implemented. Figure 7 gives the list of the implemented functions. A bus controller can send the following mul line interface message to the FRA.

1. MTA (talk address of FRA): When this message is received FRA is assigned as the talker.
2. OTA (Other talk address): When this message is received FRA is unaddressed as a talker and stop sending an device dependent message.
3. MLA (Listen Address of FRA): This message assigns FRA as a listener.
4. UNL (Unlisten): FRA does not respond to any device pedendent message sent by the talker over the interfac bus.
5. SPE (Serial Poll Enabel): FRA comes into serial poll active mode and whenever addressed as a talker sends status byte (the format of this byte is not specified, the most common application probably will be to infor the bus controller if an output process is completed or not).
6. SPD (Serial Poll Disable): FRA is put into serial poll idle state and if assigned as a talker sends measuremen results again.

The uniline message "EOI" and "REN" are not responded to by FRA. However "IFC" uniline message is sense and if it is active FRA is put into the initial state, so that according to the position of the mode selection switch, it i assigned either as a talker or as a passive bus device. Figure 8 lists the implemented messages.

5.2.2 Design of the Handler

Parellel interface handler enables data transfer between device functions and parallel interface bus. In particular i monitors three different operations. It inputs data from the bus; it sends data to the bus, and it takes proper action according to the bus controller commands. An active signal (not the "SRQ" signal) among the five signals of the inter face management bus causes a parallel interface interrupt. Since a controller's capability is not implemented as an interface function the service request interrupt due to an active "SRQ" signal is disabled. Although the other four signals (ATN, IFC, REN, EDI) can cause an interrupt, only two signals (ATN and IFC)) obtain a response. When the interrupt routine detects that the cause of the interrupt is either an active "EOI" or "REN" signal, it terminates immediately without taking any action.

The data input operation is activated by an active "DAV" signal, which causes a parallel interface interrupt. Similarly the data output operation is activated by active "NFRD" and "NDAC" signals which cause parallel interface interrupts as well.

All of these operations are implemented within the parallel interface interrupt service routine, so that whenever a parallel interface interrupt occurs, the service routine reads the status ports and detects which signal (EOI, ATN, REN IFC, DAV, NRFD and NDAC) has caused the interrupt and branches to its particular routine, (Figure 9a). As stated above "EOI" and "REN" interrupts are ignored. "DAV" interrupt routine is the acceptor handshake function implemented with all states. The "NRFD" and "NDAC" interrupt routines form the source handshake function. "ATN" interrupt routine calls acceptor handshake function routine as a subroutine to enable acceptance of the interface message. "IFC" interrupt routine puts the parallel interface to a known state (clears input and output buffers, and according to the back panel switch setting assigns FRA either as a talker or a passive device that is neither listener nor talker).

When a parallel interface interrupt is acknowledged, further parallel interface interrupts are disabled until the completion of the current interrupt service routine. Since "ATN" and "IFC" interrupt service requests have priority over "DAV", "NRFD" and "DAC" interrupts the service routines due to data byte transfer control signals which check the status of "ATN" and "IFC" signals by interspersed macros, then jump immediately to the corresponding routine if any of them is detected active.

The handler is designed to satisfy the requirements to transfer data from and to FRA by operating in the interlocked handshaking sequence, as shown in the handshaking Flowchart, in the IEEE specification, and to satisfy the conditions defining the state of the interface functions. An interlocked data transfer between a talker and a listener can be interrupted by the bus controller. Particularly "ATN" and "IFC" lines are responded to by every device within specified time limits. When the bus controller sends a multiline message this interlocked handshaking data transfer is established again to transfer these messages. Upon the completion of this transfer the previously interrupted data transfer between the listener and talker can be resumed and the handshake cycle is established to maintain this transfer. In fact the flow chart is merely a summary of the basic state transitions in interrogating source and acceptor functions.

Those loops which check the status of a handshake signal for an expected transition from false to true are broken and the status is checked only once. If the expected transition has already occurred, then the flow of the control continues in the same way (this occurs in source function flowcharts). However, if the expected transition has not occurred yet, then the routine terminates and the handshake continues when the expected transition causes an interrupt. The states in talker and listener functions are distinguished according to the contents of the flags used (Figure 10) in the implementation and according to the states of the bus signal lines.

5.2.3 Error Condition if RFD, DAC is True

If there is no listener device attached to the interface bus then an error condition arises since both "RFD" and "DAC" uniline messages are "true". However this condition is not checked by FRA parallel handler for two reasons:—

1. Since mesaurement control loop at one stage waits for the parallel interface to complete data output even if there is no listener on the bus, data is outputted for a pseudo listener device, hence is not locked up.

2. This error condition is checked as an optional feature and there is no constraint imposed upon this as the sentence on the line 122 of page 4 of HP—Patent Specification reads: "alternatively the source operating algorithum may be modified to include an interrogation of both "RFD" and "DAC" lines to determine the presence of the common "high" condition on these lines as being indicative of an error condition".

2.4 Parallel Interface Input

All measurement data operator commands entered via FRA keyboard could also be entered via parallel interface FRA is in "listener" mode. When FRA is a listener 'DAV' interrupt is enabled so that a talker could send device pendent messages asynchronously in respect to the measurement modes of FRA. Parallel interface handler stores e data bytes in an input buffer until a delimiter character is sent. Upon receipt of the delimiter the data input buffer processed. If it is a measurement data like frequency, harmonic etc., this string is transferred to the corresponding ld in data entry input buffer page. If, however, an operator command is sent like start, stop, pause, etc., its corre-onding K/B code is passed to the K/B interrupt service routine as if it were sent via FRA key-ard.

2.5 Parallel Interface Output

When measurement results are available HP—IB task outputs these results to the interface bus, if FRA is in alker" mode and 'NFRD' and 'NDAC' interrupts are enabled. After talker function is initiated in HP—IB output sk (Figure 9b), the interrupt driven parallel interface output routine outputs the content of a buffer to the inter-e bus. This buffer is filled in with results data in ASCII representation by the HP—IB output task and the buffer inter points to the first character to be outputted. This task then checks if buffer is emptied by the parallel inter-e handler or not. If buffer is empty the next results are transferred into the buffer and buffer pointer is set again. If buffer is not empty the character pointed to by the buffer pointer is outputted and the buffer pointer is incre-nted. When the buffer pointer holds the address of the last location of the buffer, then the contents of the buffer completely transferred. Although FRA remains in 'talker' mode when the buffer is empty, when new results are put o the buffer, talker function must be initiated again. If bus controller requires a status byte to be sent in a serial ll, then the status byte sends repeatedly, until either serial poll disable command is issued by the controller or FRA naddressed to talk (Figure 11).

INTERFACE CONSTRUCTION

Layout

The discrete components, IC (integrated circuit) packages and wiring parts were positioned in such a way as to luce the effects of internal and external noise.

Noise Immunity

Unwanted signals (noise) may come from many sources. The more troublesome include cross talk, line reflections l supply line spikes. Cross talk was kept to a minimum by avoiding densley packed wiring, keeping interface line gths as short as possible, use of screened cable whenever possible and the separation of I/P and 0/P lines. Standard ctice was used to curb reflections and spikes.

7. OPERATIONS

7.1 Initialization

At switch on, the interface is reset by the hardware into the following state:

- Both 8255's cleared with all ports to I/P (high impedance) state
- Interrupt flip-flop reset

The initialization software accomplishes the following:

- Sets up the 8255's to the required mode and makes the ports I/0 selection
- Reads IEEE hardwired address (from switches)
- Tests whether the microprocessor is the controller
- Tests interface management bus and acts accordingly on commands such as SRQ, IFC etc.

It is also intended that no false active levels will be presented to the interface immediately after switch on. For this reason, control signals are implemented by negative logic, since, in the initial state, all ports are in the high impedance state and are pulled up to logical 1 by means of pull-up resistors.

7.2 Interrupts

If the microprocessor enables an interrupt from the interface, the system bus line IRQ will be activated (low) under any of the following conditions:

— if the microprocessor is a talker:
- NFRD going high (1)
- NDAC going high (1)

— if the microprocessor is a listener:

- DAV going low (0)

— if the micro is a controller:

- SRQ going low (0)

— if the micro is not a controller:

- IFC going low (0)
- ATN going low (0)
- REN going low (0)
- EOI going low (0)

The microprocessor must then execute service or polling routines as necessary, under software control.

If desired, the interface may be status driven by permanently setting Not Interrupt Enable (NIEN: bit 6 of the Port DCTR) to 1. This inhibits interrupts and allows the microprocessor to continuously monitor the status port (ISR).

7.3 Control of IEEE

Control is exercised through the various buses mentioned below and illustrated in Figure 6.

7.3.1 Data Byte Transfer Control Bus

The microprocessor can activate these three lines only under the following conditions:

- DAV may be activated when the microprocessor is a talker
- NRFD, NDAC may only be activated when the microprocessor is a listener.

These lines are all open collector wire ORED to the peripherals.

7.3.2 Interface Management Bus

The microprocessor can only affect these five lines under the following conditions:

- When the micro is not a controller SRQ can be activated
- When the micro is a controller ATN, IFC, REN can be activated
- When the micro is either a talker or a controller (or both) EOI can be activated.

These five lines are all open collector wired-ored among the peripherals.

7.3.3 The Data Bus

The micro can output data onto the 8 line data bus only when it is a talker. These 8 lines may use either open-collector or active pull-up circuitry at the interface bus driver, the selection of which is made by a software controlled switch (bit 3 of Device Control Register DCTR).

When the micro is not a Talker, the bus drivers are in 'receive' mode and to prevent signal conflicts, the corresponding port DIOR should be set to input mode.

The actual operation of the interface can be explained in conjunction with Figure 6. Communication with the micro is established by the following mechanism:— Depending on whether it is high or low, ADD2 line selects either 8255 No. 1 or 8255 No. 2. If 8255 No. 2 is selected, ADD1 and ADD0 lines select one of the three ports (DSAR, DCTR, DIOR). If 8255 No. 1 is selected the ADD1 and ADD0 lines select one of the three remaining ports (ISR, DIR, IMCR). IF $\overline{RD}$ goes low at the same time, the micro can read data or status words from the 8255. If $\overline{WR}$ becomes active, the micro can write data or control words 1 and 2 to the 8255. In other words, ADD0 and ADD1 in conjunction with $\overline{RD}$ and $\overline{WR}$ inputs, control port selection. These ports are all reset by a 'high' in the initialization stage and is in the input mode.

The micro writes the control word to the chosen 8255 under software control and selects the operating mode of the 8255 (mode 0 here). Control word 1 (CWR1) selects port A as IMCR which could be a read/write port; port B as (ISR) an input and port C as (DIR) an 0/P port. CWR2 selects on 8255 No. 2, port A as IWR in write mode, port B as DSAR) an I/P which selects the device address and port C as (DCTR) an 0/P port. These ports feed directly onto four idirectional transceivers which interface to the bus highway. IMCR, DTR and DCTR go to generate an interrupt level, he FRA software then scans the interface to locate and service the particular interrupt.

CONCLUSION

The interface has been designed on the grounds of technical and economic expediency as unskilled operators exclude the use of the compiler and interpreter languages; assembly language was used with an I/0 exclusively for operator/machine interaction (11—20). Other factors taken into account are:

- Interface hardware size
- Power requirements
- I/0 structure elements
- Type of data transfer
- Noise and errors.

Errors caused by noise on the control lines were minimized by the use of twisted pair signal lines and individually screened bus transceivers IC's which include termination networks and Schmitt trigger inputs. Even if a spurious signal the control line causes a processor interrupt, this is ignored because the software does not acknowledge it.

The software handlers occupy 2k byte storage. The interface hardware is simple in design and has proved most liable to date. Interrupt features are also implemented. It is on a single board and fits in a single edge connector. The 2 mA current consumption is low and leaves a large current margin for additional interface units, as 8A are available from the 5V power supply, and 2A from the 12V supply, from which 84mA is taken by the Interface.

FERENCES

TSENG, V., Microprocessors Development and Development Systems, Mc Graw Hill, U.K., 1982.

2. LESEA, A., ZAKA, R. Microprocessor Interfacing Techniques, Sybex, 1978.
3. COLIN, A., Programming for Microprocessor, News — Butterworths, 1978.
4. RAMNARINE, R. M.Sc. Project Report, Polythecnic of Central London, 1979.
5. Intel Application Note, AP—15 (2555), 1978.
6. IEEE Standard 488—1975, "IEEE Standard Digital Interface for programmable Instrumentation".
7. LOUGHRY, C.L., The Hewlett Packard Journal, "The Hewlett-Packard Interface Bus: Current Perspectives' Jan. '75.
8. HEWLETT Packard Publication 5952—0023, The Hewlett-Packard Interface Bus., Nov. '74.
9. International Electrotechnical Commission's Draft on Interface System for Programmable Measuring Aparatu Byte-Serial, Bit-parallel, 1973.
10. HEWLETT Packard Patent Specification for Data Transfer Control Apparatus, 1973.
11. Intel 8080—8085 Microcomputer User's Manual 1980, Intel Publication.
12. Intel Publication, A guide to the Intellec MDS, 1980.
13. Intel Publication, 1515—11 System User's Guide, 1980.
14. Intel Publication, MDS Operator's Manual, 1980.
15. JACKSON, D., Principles of Program Design, Mc Graw Hill, 1978.
16. ICL Handbook — Aids to Program Design, 1978.
17. Intel Publication, 8080/8085 Macro Assembler Operator's Manual, 1979.
18. Intel Publication, 8080/8085 Assembly Language Programming Manual, 1978.
19. ZAMMIT, C., GRIST, S., Macro 11 Programming Notes, 1980.
20 Intel Publication, In circuit Emulator/80 Operator's Manual, 1980.

FIGURE 1a. GENERAL HARDWARE CONNECTIONS FOR THE INTERFACE

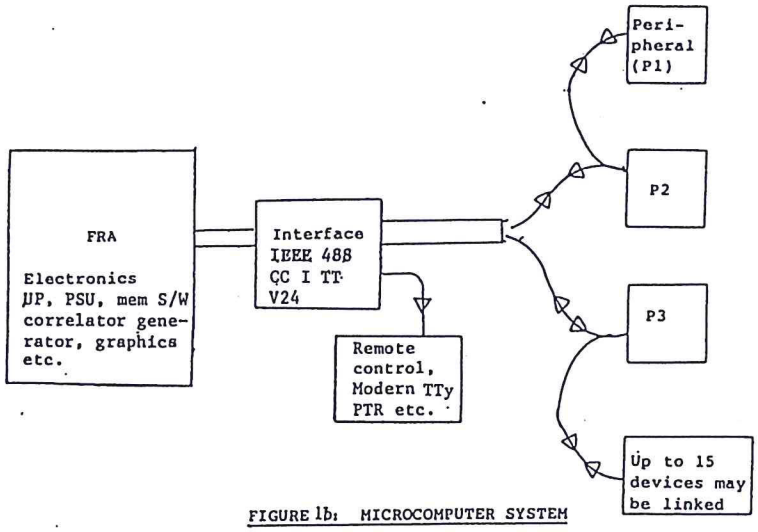


FIGURE 1b: MICROCOMPUTER SYSTEM

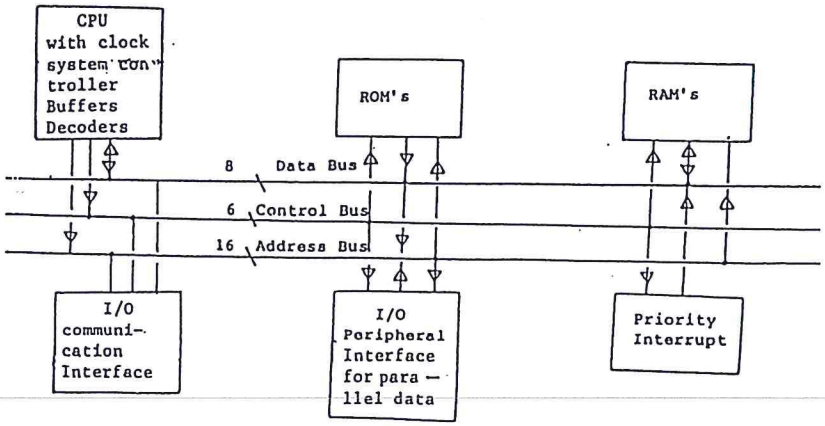


FIGURE 2 : INTERFACE DIAGRAM IN BLOCK FORMAT

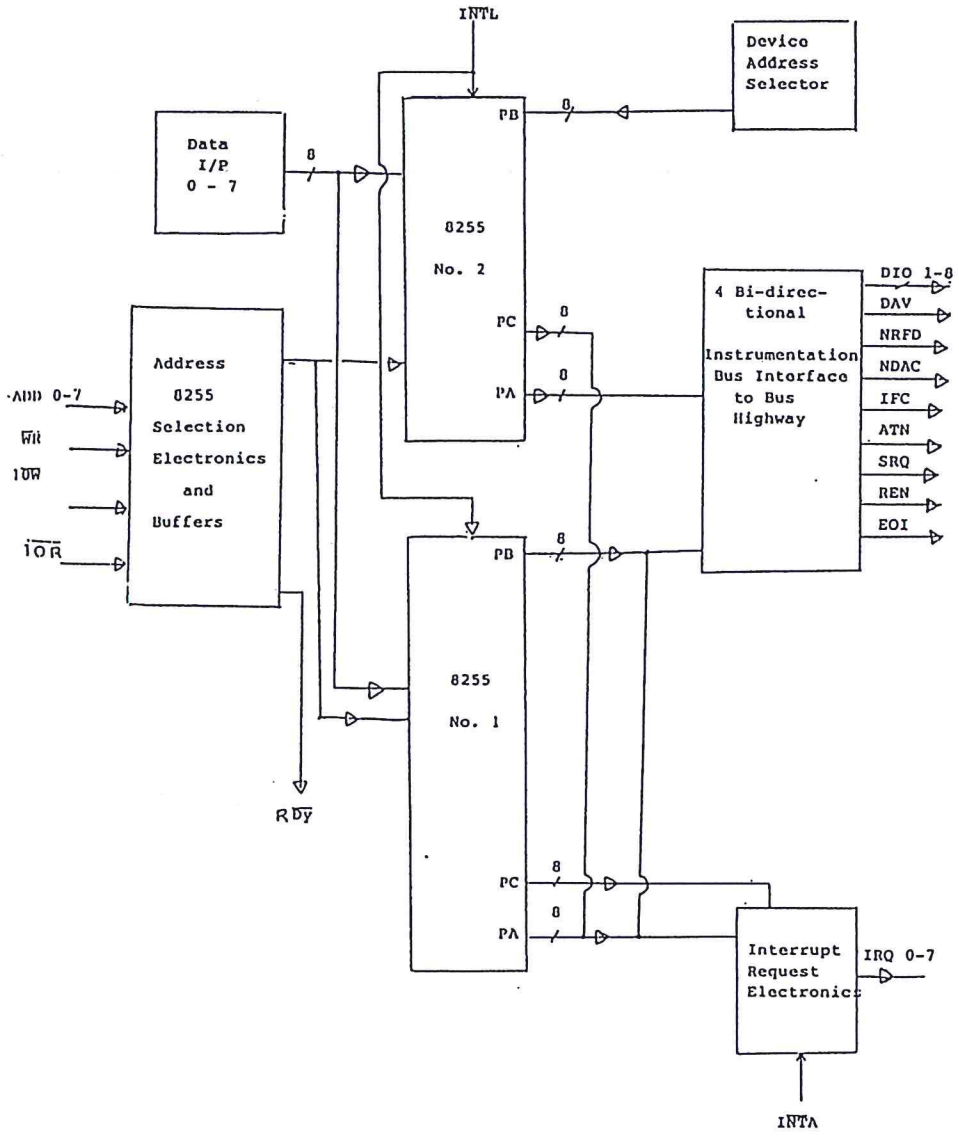
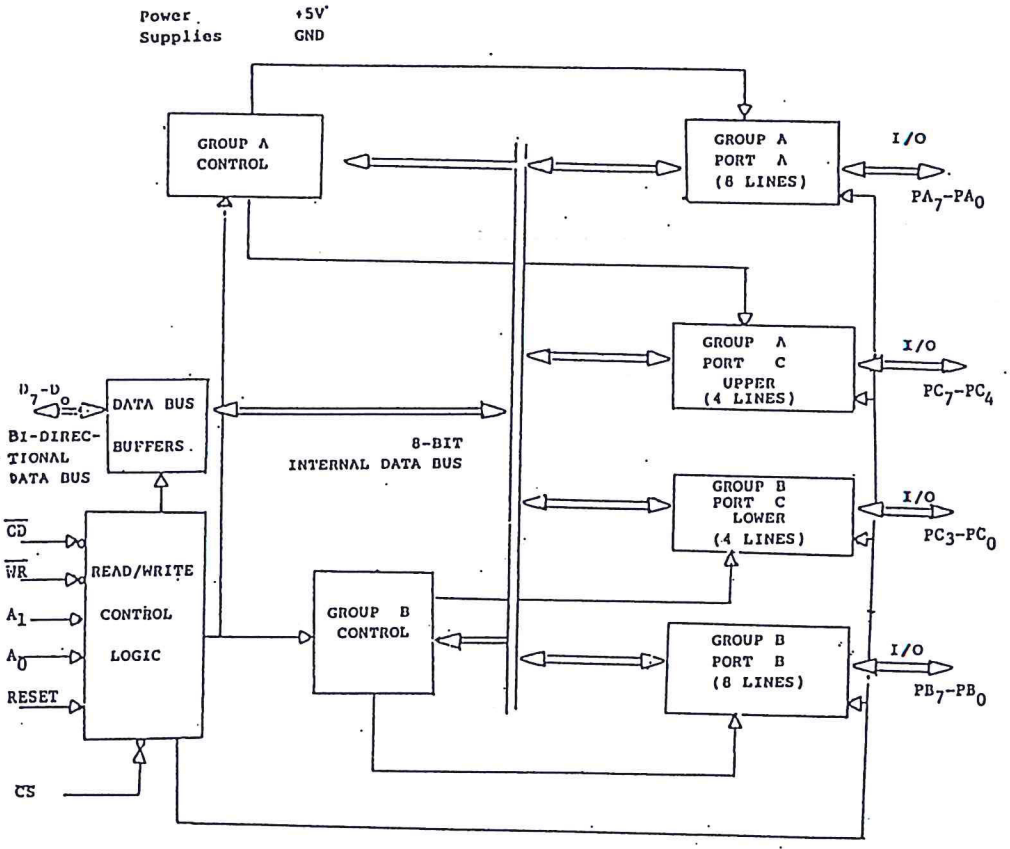


FIGURE 3: BLOCK DIAGRAM OF 8255 CHIP



D_7-D_0	Data Bus (Bi-directional
RESET	Reset I/P
$\overline{CS}$	Chip Select
$\overline{RD}$	Read I/P
$\overline{WR}$	Write I/P
A_0, A_1	Port Address
PA_7-PA_0	Port A (Bit)
PB_7-PB_0	Port B (Bit)
GND	Zero Volts

FIGURE 4: THE 8255 MODE DEFINITION FORMAT

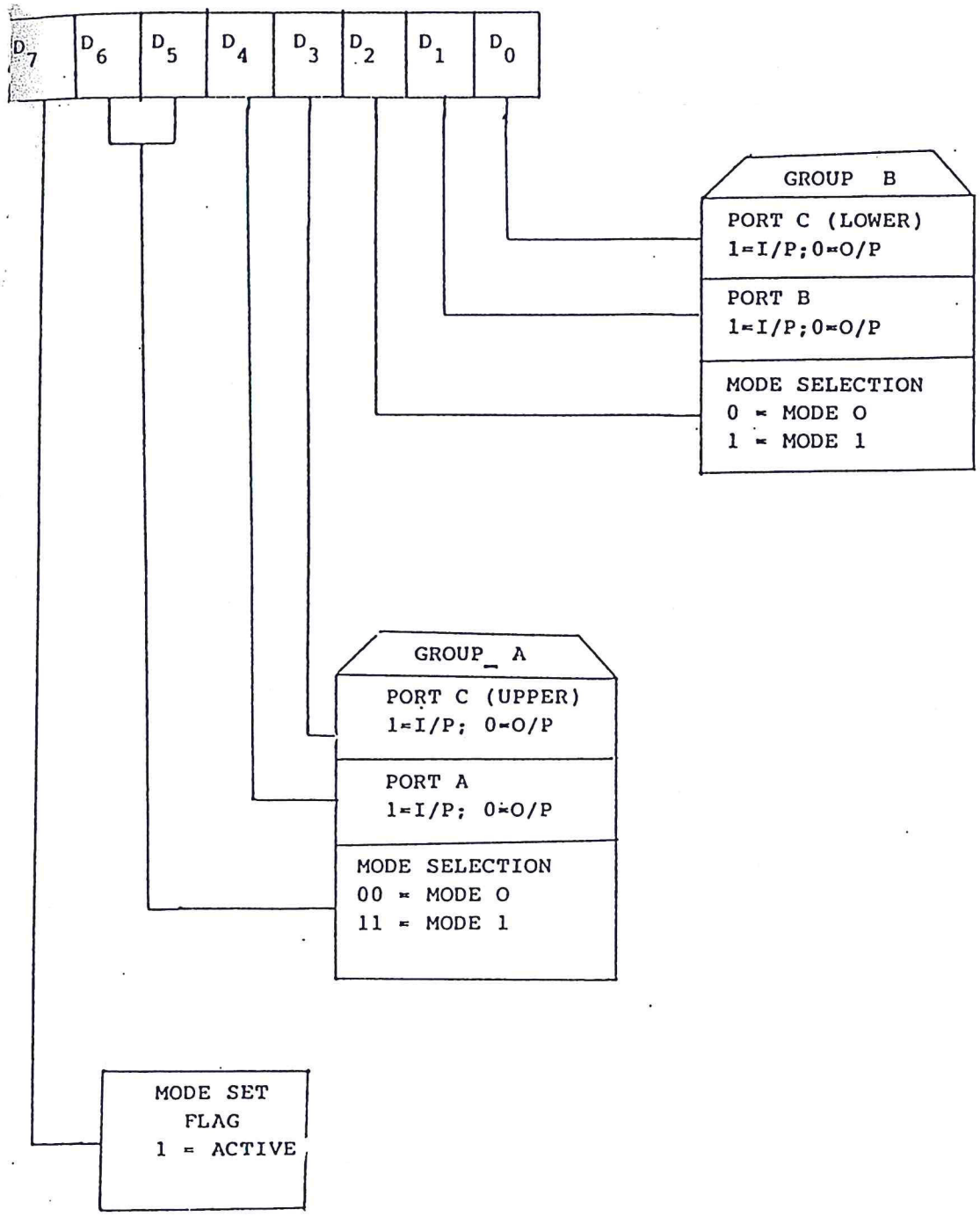
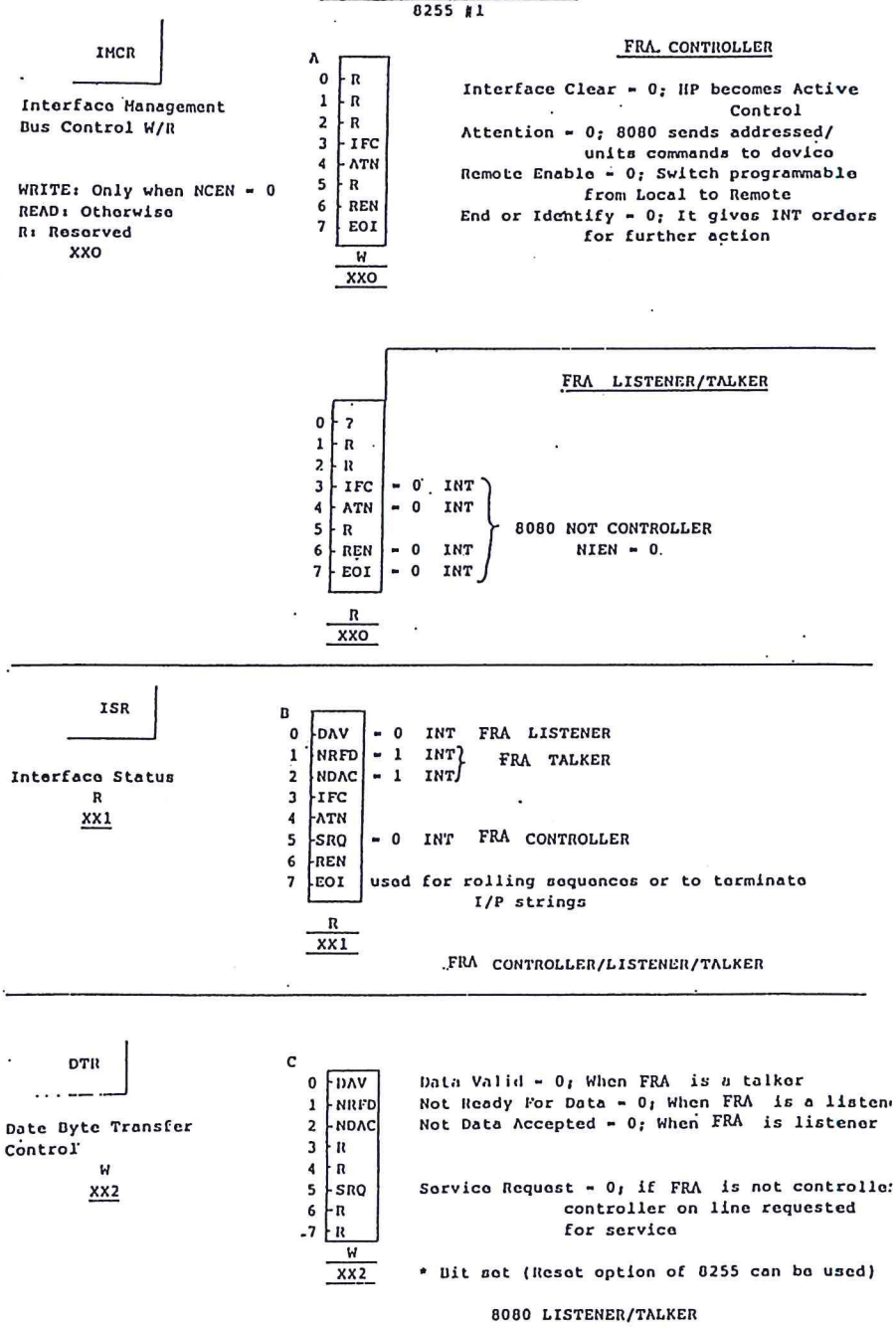


FIGURE 5: PORT SELECTION

8255 #1



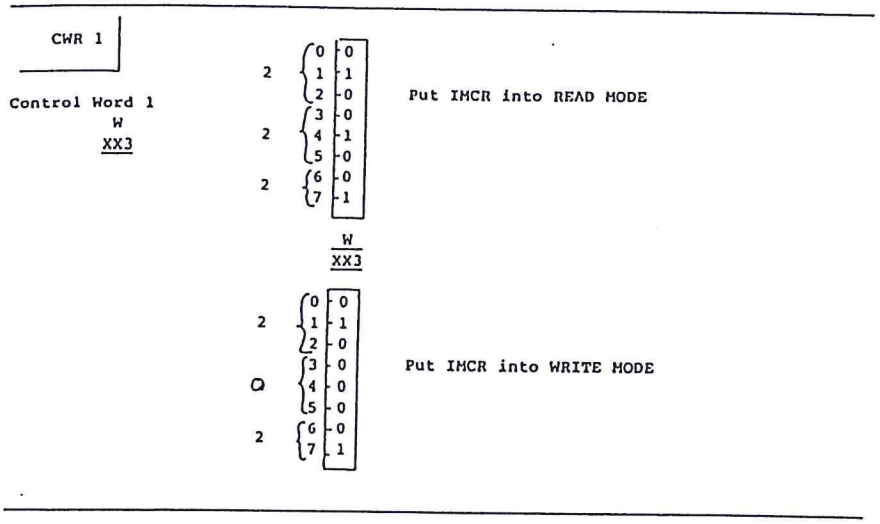


FIGURE 5. Cont'd

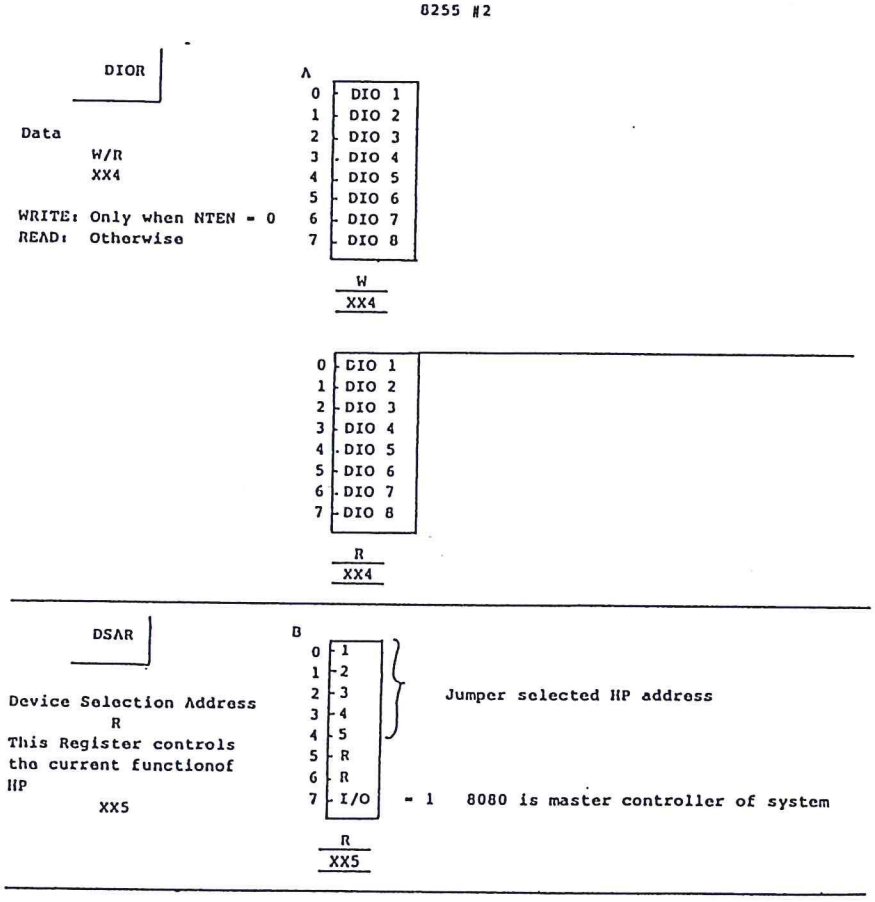


Figure 5 Cont'd

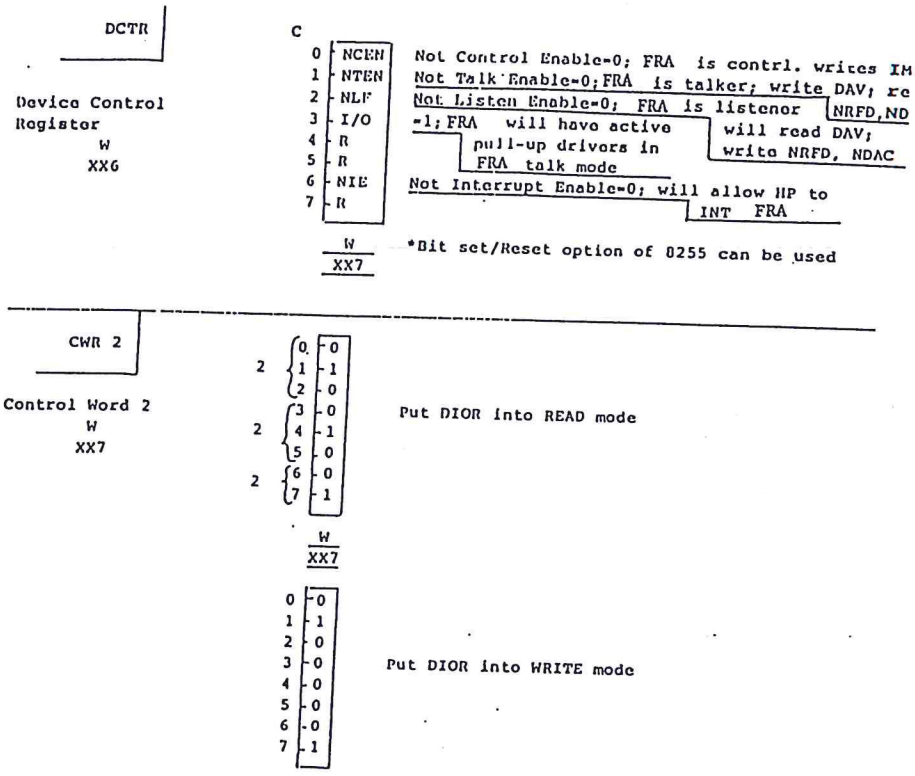


Figure 5 Cont'd

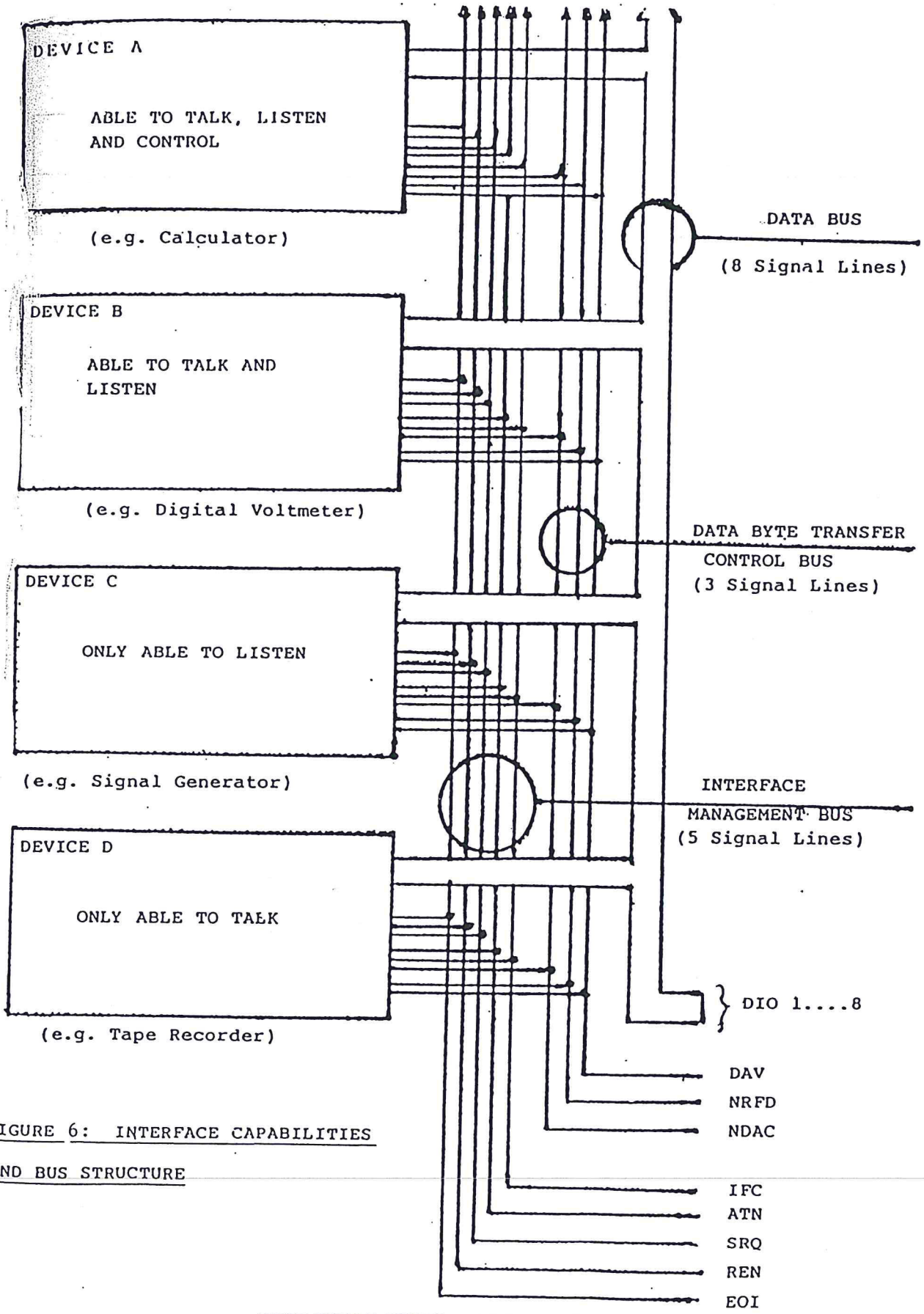


FIGURE 6: INTERFACE CAPABILITIES AND BUS STRUCTURE

FIGURE 7: TABLE OF SOFTWARE FUNCTIONS IMPLEMENTED BY THE INTERFACE

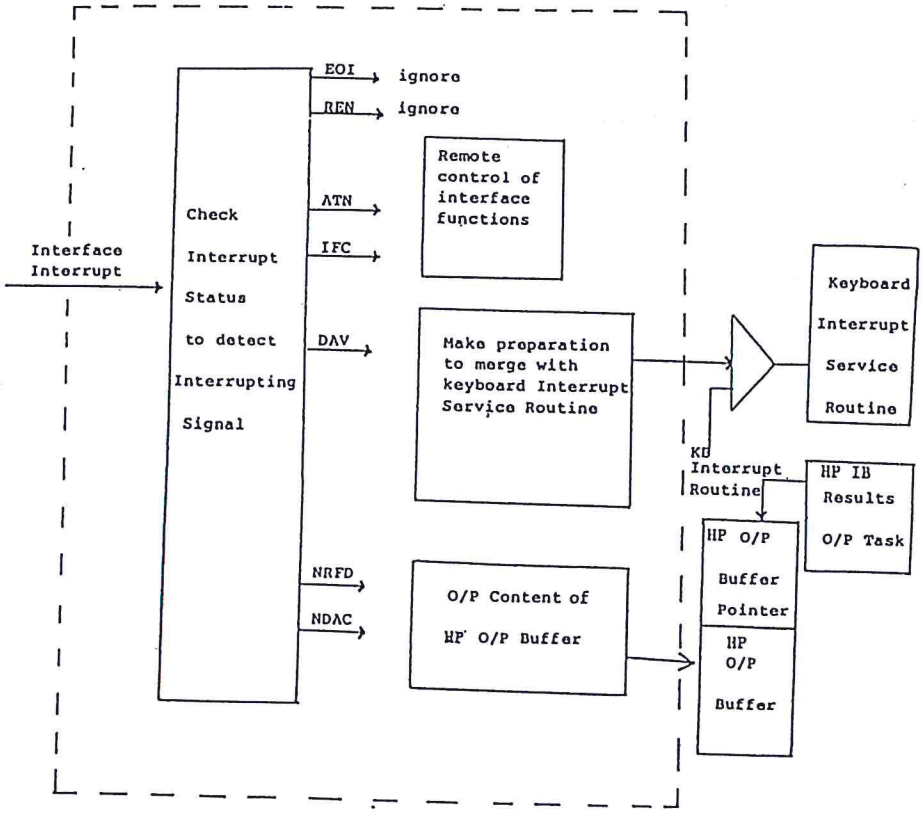
I/F FUNCTIONS	IDENTIFICATION	DESCRIPTION
Source Handshake	SH 1	Complete Capability
Acceptor Handshake	AH 1	Complete Capability
Talker	T5	Complete Capability
Listener	L4	Basic Listener
Service Request	SRQ	No Capability
Remote Local	RLQ	No Capability
Parallel Poll	PPQ	No Capability
Device Clear	DCQ	No Capability
Device Trigger	DTQ	No Capability
Controller	CQ	No Capability

FIGURE 8: MULTILINE/UNILINE MESSAGES

(1) Those implemented are:

ATN (Attention)
IFC (Interface Clear)
DAV (Data valid
RFD (Ready for data)
DAC (Data accepted)
OTA (Other talk address
UNL (Unlisten)
SPE (Special Poll Enable)
SPD (Serial poll disable)
MTA (my talk address)
MLA (my listen address)

FIGURE 9a, INTERFACE HANDLER SHOWN WITHIN DOTTED BOX



(1) If ATN signal becomes active DAV signal and data must be removed from Bus in 200 nS

(2) If IFC signal becomes active, it must be responded to within 100μs

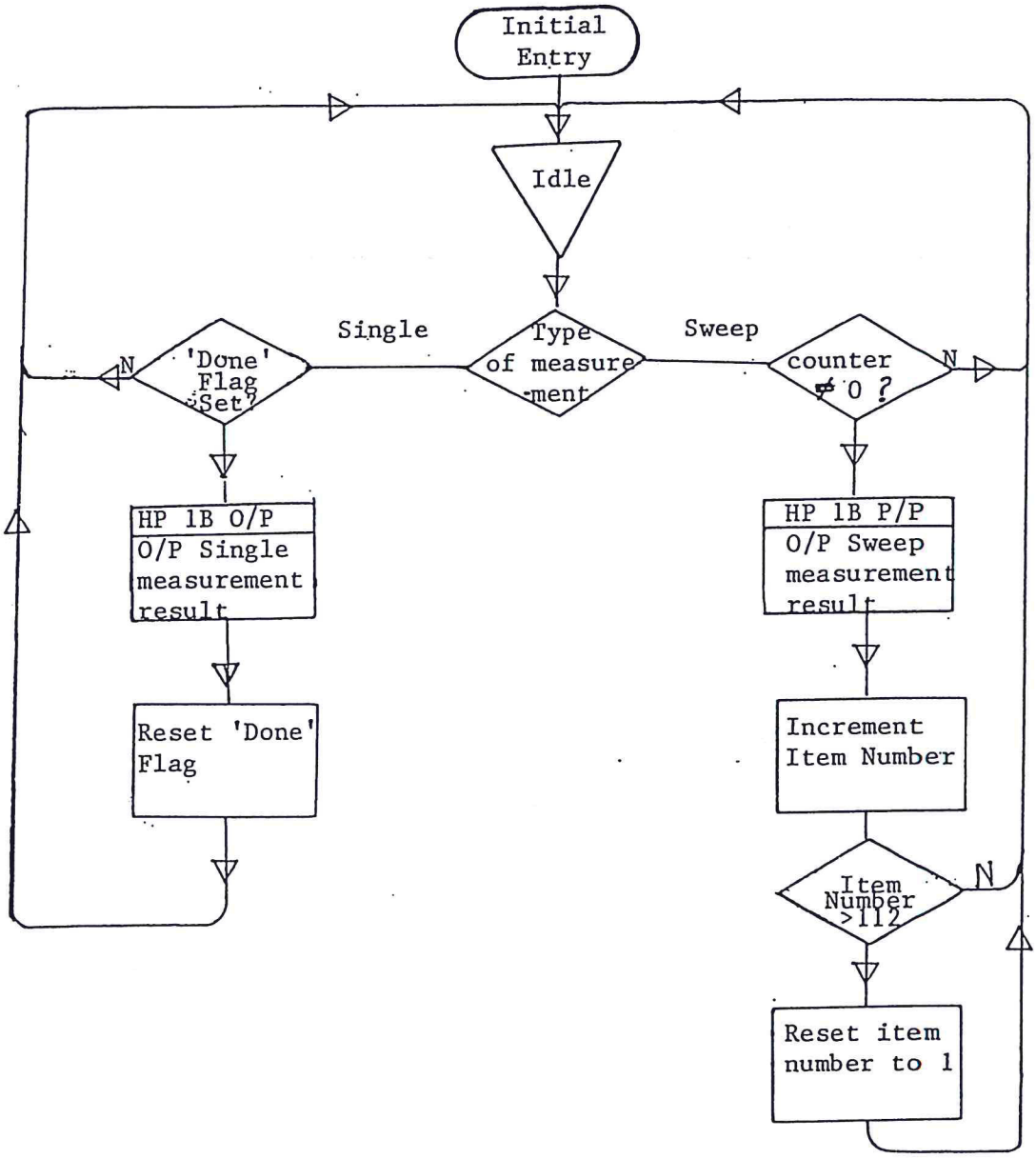


Fig. 9b. General Control Software handler for HP1B output task for IEEE 488 interface.

FIGURE 10: FLAGS USED

ATN @ FG : Used by acceptor function routine to determine if a received multiline message is to be interpreted as an interface message or as a device dependant message.

SPE @ FG : Used by source function routine to determine if a status byte or a data byte is to be sent over the interface bus.

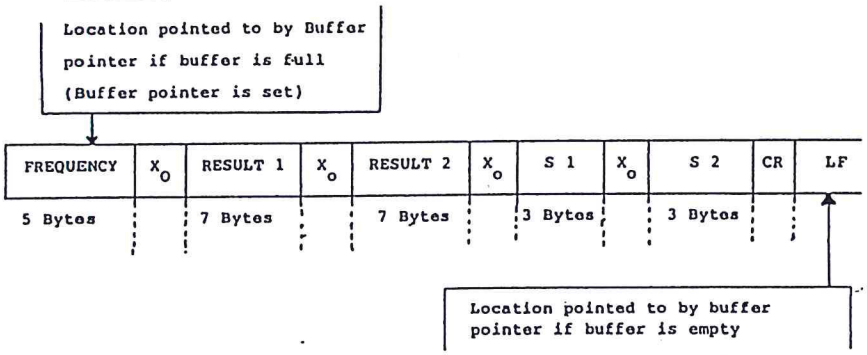
TALKER @ FG : Shows if FRA is a talker or not.

LISTENER @ FG : Shows if FRA is a listener or not.

NDA @ FG : Shows if new data byte is available or not.

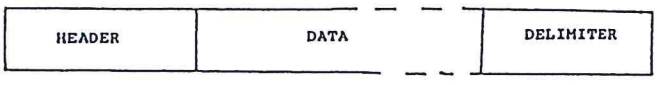
FIGURE 11: INTERFACE BUFFERS

a) O/P Buffer



At Reset, address stored in Buffer pointer is end address of Buffer.

b) I/P Buffer



Buffer Pointer: At reset the address stored in buffer pointer is the start address of the buffer.