VLBA Technical Report No. 12

VLBA STANDARD INTERFACE BOARD MANUAL

WAYNE KOSKI AND DAVID WEBER
JULY, 1991

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

1.0 INTRODUCTION

This manual describes the VLBA Standard Interface Board used as a general purpose monitor and control interface in the VLBA Telescope Monitor and Control System. The Antenna Control Computer is the system Controller in this system. The Controller communicates with Standard Interface Boards installed in the system components (these components are called devices in this manual). The communication path is a two-pair, party-line, time-serial Monitor and Control Bus. The Controller (the computer) outputs time-serial, simplex mode, messages to the system devices via the Command (XMT) bus. XMT bus messages are either control messages to a device or monitor request messages to cause the device to return monitor data messages to the computer on the simplex mode, Monitor Data (RCV) bus.

The Standard Interface Board is typically a modular component of a device and functions as a serial-parallel converter to interface the Monitor and Control Bus to the devices that are to be controlled or monitored. Some devices are modules, such as a receiver control unit or an IF processor; other devices are subsystems, such as the subreflector position control system or the weather station. In either case, the use of the Standard Interface in monitor and control interfacing is very simple; the chief differences between applications are the character and volume of control or monitor operations to be performed.

Because analog signal multiplexing and A/D conversion are frequently required, the Standard Interface Board contains an optional analog multiplexer-A/D converter. The converter is integrated into the logic of the interface so that it may be easily applied to analog signal monitoring applications. The analog mulitplexing capacity of the interface may be extended by additional analog multiplexers installed in the device circuitry. The two versions of the Standard Interface Board differ in the type of analog multiplexing. Version "S" uses a single-ended multiplexer and version "D" uses a differential input multiplexer-instrumentation amplifier for applications where common-mode noise is a concern.

The VLBA Standard Interface Board is always used in conjunction with digital circuitry in the device that implements the device address, command and monitor data functions. Figure 1 (following this page) shows the block diagram of the Standard Interface Board and associated device logic.

This board has been used in several PC-controlled monitor and control systems. In these applications, the PC RS-232 serial port drives the Monitor and Control Bus via RS-232/RS-485 translator chips in the device logic. A Turbo Pascal driver program is available for these applications.

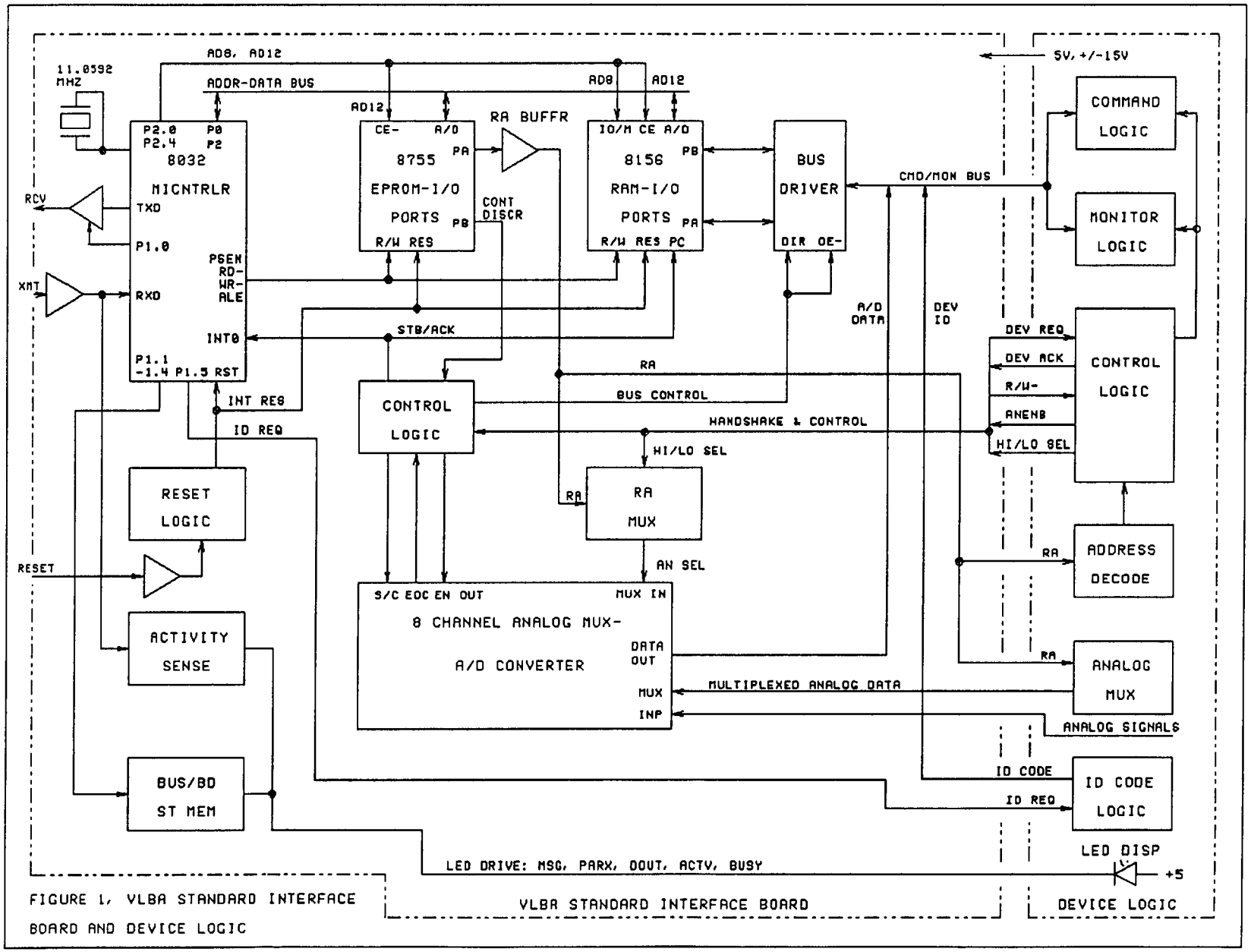


FIGURE 1, VLBA STANDARD INTERFACE BOARD AND DEVICE LOGIC

This interface board has also been used in NRAO applications which are not part of the VLBA project. The application information in this manual may be useful in these cases.

This single, general purpose interface board is used for most of the VLBA Monitor and Control system interface applications. Since it is a complete interface to the Monitor and Control bus, it reduces design time by obviating the need for any other bus interfacing design. Being a single design, it insures uniformity in the design of device control and monitoring circuitry because the logic designs conform to a single set of simple rules. The few cases in which this board is not used in the VLBA station are applications in which, for packaging convenience, the chips and firmware of the interface board are directly incorporated into device circuitry. In these cases, the operation of the interface circuitry is identical to that on the Standard Interface Board.

The VLBA Monitor and Control Bus (XMT and RCV) conform to the EIA RS-485 signal specification (Section 6 of this manual). The XMT and RCV bus message formats and protocol are described by specification A55001N001 (Section 6).

The Standard Interface Board is an address-restricted implementation of Specification A55001N002-A (Section 6). The board device address capability is 256 command and 256 data channels. In addition there are 16 interface internal command and 16 interface internal data addresses. The message format specifications (cited above) do not define the maximum size of an interface address block; it could be any size within the specified address range. The interface board specification is both a functional and physical specification in that it defines the functional properties, physical size, and I/O connector types and pin assignments.

Readers unfamiliar with these two specifications are urged to carefully review them before proceeding further in this manual; these specifications determined the hardware and firmware design of the interface.

Manual Overview

This manual provides a detailed description of the digital and analog circuitry of the interface board and the control firmware which determines the operating characteristics of the interface. Subjects such as message formats, bus protocol, bus signal characteristics, the microcontroller and associated support chips, the analog multiplexer-A/D converters, device interface signals and timing are all described in detail. In addition, this manual describes typical device interface logic and address block assignments. The control firmware algorithms and firmware are fully described in Section 3. These subjects are listed on the Table of Contents and List of Illustrations.

The characteristics of the Intel 8032 microcontroller which are important to the operation of the Standard Interface are described and 8032 data sheets included in Section 6. The characteristics of the Intel 8755 EPROM-I/O port chip and the 8156 RAM-TIMER-I/O port chip are also described and data sheets for these chips are included in Section 6.

The two types of analog multiplexer-A/D converters are described. Data sheets for these components are included in Section 6.

The EIA RS-485 specifications are included in Section 6.

Standard Interface Board Overview

The Standard Interface Board is a 6"x6"x.75" PC board suitable for installation in an NRAO module.

The address capacity of the Standard Interface Board is 256 device command channels and 256 device data channels. This is more than adequate for all reasonable applications. In addition there are interface internal command and data addresses associated with bus conditions and interface internal functions. These addresses consist of 16 command and 16 data channels. The analog multiplexing capacity of the board is 8 channels which may be easily expanded to a maximum capacity of 64 channels by the addition of one or more eight-channel analog multiplexers in the device circuitry.

The address range for all interfaces in a system is 32 K. A flag bit in the XMT message address field distinguishes control messeges from data request messages; thus the VLBA Monitor and Control system can accomodate up to 32K of control and 32 K of data request messages. The message formats and structure are described in Section 2.1.

The Controller assigns unique blocks of contiguous addresses to interfaces. The Interface Board uses a Device ID code (read from the device logic) as an index to the address block assignment stored in the interface's RAM memory.

The Standard Interface Board hardware and firmware design has diagnostic features to identify, record and report bus fault conditions (parity errors) and non-response of the device circuitry. The recorded fault and data conditions are parity error counts and address and arguments of the most recent control and data request messages. These values are read out of the interface as internal interface data. The Controller can set new values (typically resets to zero counts) by control messages. (The values in three addresses cannot be altered by the Controller). The interface can drive front panel display LED's to provide a visual indication of bus and board activity.

It is easy to interface the device logic with the Standard Interface Board; interface logic is simple address decode and hand-shaking. Logic timing is simple and non-critical. Timing diagrams in Section 2.6 graphically illustrate these relationships.

The use of even parity for message control function codes provides unique, unambigous message interface synchronization and response feed-back to the controller.

The VLBA Monitor and Control Bus operates at a 57.6 k-baud rate. The interface board may be adapted to operate at any of the (lower) standard baud rates by reprogramming the 8755 EPROM. Details of the baud rate selection are described in Section 2.7.

The Standard Interface Board has provisions for an external reset input to re-initialize the board logic and microprocessor via an RS-485 Reset (RST) bus. This feature may be selected by a jumper plug on the board. The VLBA Monitor and Control system does not use this reset feature.

2.0 THEORY OF OPERATION

2.1 MESSAGE FORMAT, PROTOCOLS AND TIMING

This section is an abstract of the bus specification A55001N001 and describes the characteristics of the serial VLBA Monitor and Control Bus. The bus Controller is the station computer and its interface to the bus. The Standard Interface in a device connects to the bus.

Interface Address Block

Each Interface is assigned a block of contiguous addresses to which it alone responds. The block may be any length, but it must be disjoint with the address blocks of all other Interfaces. The last 16 addresses of each block are dedicated to monitor and control functions internal to the Interface. These functions are described below.

Monitor and Control Bus

The bus consists of two differential-mode logic signals, each on a shielded twisted pair cable wired as a multi-drop party line. The bus signals are called Transmit Data (XMT) and Receive Data (RCV). There is one Controller (the antenna control computer) and numerous Standard Interfaces (in devices) are connected to the bus. The Controller is the only source of Transmit Data, and the Interfaces (one at a time) are the sources of Receive Data. Data is bit-serial at a rate of 57.6 kbaud and the transmissions are byte asynchronous. Each byte consists of, in order, a start bit (binary 0), eight data bits (least significant bit first), one parity bit, and one stop bit (binary 1).

XMT and RCV bus messages are combinations of data value bytes and control function codes. The data value bytes are address and argument values: Address High (ADH), Address Low (ADL), Control Data High (CDH), Control Data Low (CDL), Monitor Out High (MOH) and Monitor Out Low (MOL). Data value bytes are transmitted with odd parity.

Control function codes (bytes) signal the start of a message or report interface-device status during the execution of control and data request messages. The Control Function codes are: Synchronization (SYN), Acknowledge (ACK), Second Acknowledge (DC1), Non-Response Acknowledge (DC2) and Negative Acknowledge, (NAK). Control Function codes are transmitted with even parity.

XMT Bus

Every message on the XMT bus is exactly five bytes long and the bytes occur in the following sequence: SYN, Address High (ADH), Address Low (ADL), Control Data High (CDH), and Control Data Low (CDL). The SYN byte indicates the beginning of a message, and is the only even bit parity byte on the XMT line (thus distinguishing it from all data bytes). The SYN byte is followed by ADH. If the most significant bit of ADH is 1, then the message is a control message; otherwise it is a monitor request message. The remaining 15 bits of ADH/ADL form a binary address in the range of 0 through 32767.

Each Standard Interface receives ADH and ADL of every message on the XMT line (there is no dead time during which an interface is not listening). Each Interface is assigned a block of contiguous addresses to which it alone responds. The block may be any length, but it must be disjoint with the address blocks of all other Interfaces. The last sixteen addresses of each block are dedicated to functions occuring within the Interface, as described below under Interface Board Parameters.

The Interface must check parity on all bytes received. If SYN has a parity error, an internal counter (BE-4, invalid SYN character counter) is incremented. The value of this counter is assigned to a monitor address.

If ADH or ADL has a parity error, the Interface does not respond (just as if the address were outside its block), but increments an internal address parity error counter (BE-7, address parity error counter) and looks for the next valid SYN. The value of this counter is assigned to a monitor address.

If the address transmitted is within the assigned interface address block, then within 382 microseconds of the last bit (Stop) of ADL, that interface must begin to transmit a one byte acknowledge code (ACK) on the RCV line. If the message was a control message (indicated by a 1 in the msb of the address), then the Interface must also receive and store CDH and CDL, and within 382 microseconds of the end of CDL or 573 microseconds after the initial ACK was put on the line (whichever occurs later) it must begin to transmit a second acknowledge byte (DC1) on RCV.

If SYN, ADH, ADL, CDH and CDL have valid parity, the address is within the assigned block, a control message is specified and the device responds to the interface handshaking properly, a DC1 acknowledge code is transmitted on the RCV line.

If SYN, ADH and ADL have valid parity, the address is within the assigned block and a control message is specified, but CDH or CDL has a parity error, then the second acknowledge byte (DC1) is replaced by a negative-acknowledge code, NAK. In this case CDH/CDL is not passed to the device and a control data parity error counter is incremented. The value of this counter is assigned to a monitor address. A second form of negative acknowledge byte (DC2) is returned if the interface is unable to complete its handshaking with the device to which it interfaces. [This may occur (but does not necessarily occur) if the device is powered down or unplugged, and signals other conditions in which the state of the controlled device is not altered (for commands) or for which it is not meaningful to return monitor data.] The non-response condition is counted by two internal counters: BE-12 for control messages and BE-11 for monitor data request messages.

If the message is a monitor request message within the address space assigned to monitor data, the CDH and CDL byte values are ignored, they have no meaning. The parity of CDH/CDL is, however, tested and the control data parity error counter is incremented if tainted by an error. In this case, the monitor data specified by ADH/ADL is returned in the normal manner because the CDH/CDL parity error has no effect upon the validity of the monitor data.

RCV Bus

Messages on the RCV line are either a two byte command acknowledgement or a two or three byte monitor data acknowledgement, as follows:

Command acknowledgement messages are: 1) ACK, DC1 (normal, no fault command execution acknowledgement), 2) ACK, NAK (CDH/CDL parity error acknowledgement), and 3) ACK, DC2 (device non-response acknowledgement).

Monitor request acknowledgement messages are: 1) ACK, MOH, MOL (normal, no fault acknowledgement followed by two bytes of monitor data obtained from the address specified by ADH/ADL); 2) ACK, DC2 (fault acknowledgement, monitor data is unavailable from the device).

The Controller also checks parity on all bytes received on the RCV line. In the event that monitor data or function codes have parity errors, the Controller application software notes the errors and disqualifies the data.

Bus Timing

The first acknowledge byte (ACK) will also function as a clear to send to the controller, granting the controller the right to begin its next message, and promising to yield the RCV line before it is needed for another interface's response. (The time available to it is at least 764 microseconds, but may be longer if the controller has not begun to transmit CDL on the XMT line at the time the interface begins to send ACK on the RCV line; the interface will have at least 573 microseconds after the end of transmission of CDL to disconnect from RCV). Figure 2 (on the next page) shows the timing relationships.

The Controller may begin transmitting another message following a control message after the receipt of the acknowledge (ACK) byte. The maximum-speed timing (in the sense that all transactions are limited by line speed) for a sequence of control messages and for a sequence of monitor requests is illustrated in Figure 2. The minimum speed timing (in the sense that the controller is responding as rapidly as permitted by this protocol, while the interfaces are responding as slowly as permitted by this protocol) is illustrated in Figure 2.

Note from the figures that in the maximum rate mode of control message reception, the interface is transmitting the second acknowledgement (DC1) concurrent with the reception of a new control message; DC1 is being transmitted at the same time that ADH for the new message is being received on the XMT bus. This concurrency requires that the microprocessor manage both message transmission and reception operations concurrently.

Note also from the figures that in the maximum rate mode of monitor request message reception, MOH/MOL are being transmitted on RCV at the same time that a new message is being received on the XMT bus. Like the case above, the microprocessor must manage both types of message operations concurrently.

Function Codes

The hexadecimal byte values for the control function codes are as shown below. These bytes are transmitted in even parity which makes them unique since data bytes are odd parity.

SYN - 16, Synchronization byte which prefixes all messages from the controller.

ACK - 06, First acknowledgement byte to the controller which signifies that SYN was detected and that SYN, ADH and ADL did not have parity errors. ACK is not transmitted if there was a parity error on any of these three bytes.

DC1 - 11, Second acknowledgement byte to the controller which signifies that in the case of control messages, there were no parity errors in CDH/CDL and the device responded properly to the interface-device handshaking requirements.

NAK - 15, Negative acknowledgement byte to the controller which signifies that, for control messages, CDH/CDL was tainted by a parity error.

DC2 - 12, Second negative acknowledgement byte to the controller which signifies that the device logic did not respond to the interface handshaking within the allocated time period. This non-response acknowledgement is used for both control and monitor request messages.

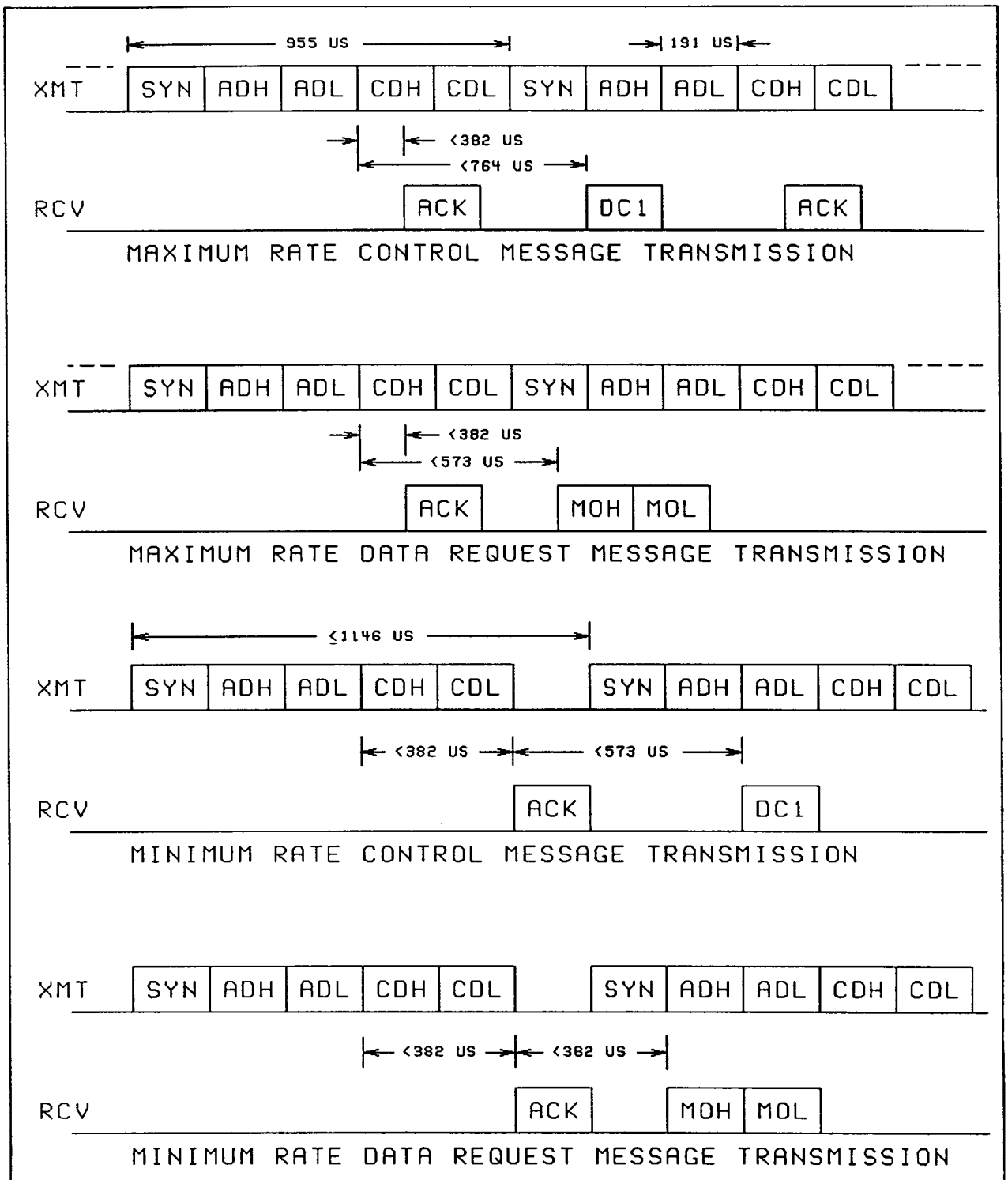


FIGURE 2, XMT AND RCV BUS TIMING

Interface Internal Monitor and Control Functions

The last 16 addresses in the address block are allocated to Monitor and Control parameters internal to the Standard Interface board. The occurance of bus fault conditions (such as parity errors, invalid SYNC, etc.) are accumulated in counters and are available for monitor data readout by the Controller. Device non-response events are also accumulated. These parameters were described above. The counters are capable of being reset or set to other values by a control message from the Controller.

Other parameters in this block are the Interface Type and Revision level, Block ID Code (N) and Address Block start address. This Block ID Code is the index for assignment of the address block by the Controller. None of these locations can be overwritten by the Controller. The last three addresses in the block are reserved for future assignment.

These Interface addresses are identified by the notation: BE-1, BE-2, etc. where BE designates the Block End (last) address. The assignments are as follows:

Address	Value
BE-15	Reserved for future use
BE-14	" " " "
BE-13	" " " "
BE-12	No Control Response counter (i.e, no DEV ACK from device)
BE-11	No Monitor Response counter (i.e., no DEV ACK or ANENB from device)
BE-10	**Interface Type and revision code (cannot be altered by the Controller)**
BE-9	Address of last control message received. (i.e., ADH and ADL)
BE-8	Control data for last control message received. (i.e., CDH and CDL)
BE-7	Address parity error counter, all messages
BE-6	Control data parity error counter, all messages
BE-5	Invalid SYN character
BE-4	Control data parity error counter, messages in block
BE-3	**N, ID byte value from device logic (cannot be altered by the Controller)**
BE-2	Count of correctly received control messages
BE-1	Count of correctly received monitor data request messages
BE-0	**Address of beginning of block (cannot be altered by the Controller)**

The addresses designated by bold print cannot be altered by the Controller. The reasons for this restriction are described in Sections 2.5 and 3.3.

Bus Description

Important properties of the VLBA Monitor and Control bus are as follows:

RS-485 specifications permit up to 32 drivers and receivers on a bus. Thus an RCV line could have up to 32 drivers and one receiver (the Controller), and a XMT line could have one driver (the Controller) and up to 32 receivers in Standard Interfaces. Bus drivers and receivers are 75174 line drivers and 75175 line receiver chips which conform to the EIA standard RS-485 specifications.

In the VLBA station, there is a possibility that the Monitor and Control Bus may have to service more than 32 devices. This exceeds the capacity of a single XMT driver chip in the controller and the capacity of the interface RCV driver chips. To provide adequate drive capacity and to simplify cabling, three bus cable runs are used between the controller and devices in the station. These three buses use separate driver chips that are driven by a single XMT signal from a UART; there is no XMT selection of cable runs by the controller. The UARTs that service the three RCV receiver chips are read by the controller in sequence. The first cable run is used to service devices in the station building (this includes the weather station). Approximately 29 devices are serviced by this bus run. The second cable run

services devices in the antenna Pedestal room; approximately 4 devices are serviced by this cable run. The third cable run services devices in the antenna Vertex room; approximately 16 devices are serviced by this cable run.

The bus transmission rate is 57.6 kbaud, including all framing and parity bits. The transmission rate may be lowered by reprogramming the 8032's baud rate generator counters in the EPROM firmware.

The bus transmission lines are #24 twisted pair, shielded, (roughly 100 ohms characteristic impedance), max length 500 feet, terminated with a 100 ohm resistor. Belden # 9842 cable is used for the bus cable runs between electronics racks. Bus wiring within a rack is two pairs of unshielded wire. At the end of the bus run, 100 ohm terminating resistors terminate the bus.

All rack bus connectors are Cinch or Amphenol DE-9S or equivalents. All bus cable connectors are DE-9P connectors or equivalents.

Controller and interface bus drivers and receivers are bridged across the XMT and RCV bus lines with stubs less than 20 feet.

RCV bus drivers are tri-state, connected to the bus only when required to respond to a monitor request. RCV drivers maintain their high impedance state when the interface is unpowered.

To provide bus line high-voltage safety, clipping surge arrestors are used on the bus lines between the control building and the antenna. Surge arrestors are located at each end of the bus cable run and shunt the surge currents to antenna frame and building counterpoise ground.

Interfaces which service equipment subject to lightning-induced currents are protected by optical isolators with high voltage (typically 2500 volts) hold-off capability. Examples of such devices are the subreflector drive, the weather station, and feed heater controller. In the case of the weather station, a fiber optics interface drives the bus lines to the Standard Interface Board in the weather station enclosure. In the subreflector position controler, the bus is protected by optical isolators which are interposed in the discrete signal lines between the controller and the apex. Resolver signals are protected by surge arrestors and isolation circuitry in the R/D converters. In the case of the feed heater controller that powers heaters in the dish, optical isolators in the controller isolate the controller from the Utility Interface.

Bus signals shall conform to EIA RS-485. Important properties of the RS-485 specification are:

Transmission and reception modes are differential, +-2 to +-6 volt signal range.

Drivers and receivers are capable of operation in the presence of common mode voltages between -7 to +12 volts.

When two or more drivers contend for a bus, there is no device damage but the bus signal may be distorted or garbled.

The maximum driver output current in the tri-state condition is - +- 100 ua.

The maximum driver output current with power off is: - +- 100ua.

Receiver input sensitivity is +- 200 mv, minimum, differential mode.

The receiver input resistance is 12 kohms, minimum.

The driver output signal is +- 1.5 V minimum, differential mode, into a 54 ohm load.

Loss of power on one or more drivers or receivers does not cause device damage.

The RCV bus has many interface tri-state drivers connected to it. To eliminate occasional signal transitions resulting from tri-state current interactions of the driver chips, the RCV bus lines are biased at the input to the Controller. On the RCV+ line, a 1 kohm resistor is connected to the Controller +5 Volts line. The RCV- line is connected to the Controller logic common through a 1 kohm resistor and a 100 ohm line terminating resistor is connected across the two lines.

The Controller interface is a modified Motorola MVME705A, six-channel serial driver/receiver board. Motorola document MVME705A/GD1 describes this interface. Drawing C55001A015 describes the modifications to adapt the board to the RS-485 signal domain. This entails replacement of the RS-232 drivers and receivers with 75174 and 75175 chips. The termination resistors mentioned above are also installed on this board.

2.2 8032 MICROCONTROLLER AND SUPPORT CHIPS DESCRIPTION

This section describes the characteristics of the 8032 microcontroller, 8755 and 8156 support chips in the context of the Standard Interface Board application. Details which are not relevant to this function are briefly discussed or passed over. The application of these chips signals to the Standard Interface Board logic is described in the Board Logic description below.

The Standard Interface Board microcontroller is an Intel 8032, which is a member of the eight-bit Intel MCS-51 family. The architecture is shown in Figure 1. For a detailed description of the chip architecture refer to the MCS-51 data sheets in Section 6. The 8032 has three timer/counters (Timer 0, 1 and 2), a 256 byte on-chip RAM memory, and uses an external program memory. The 8032 support chips are: an Intel 8755 2-Kilobyte EPROM-I/O Port chip for firmware program storage and I/O, an Intel 8156 256 byte RAM-I/O-Timer, and three 74LS245 bidirectional bus drivers. The 8755 chip has two eight-bit I/O ports with bits that can be individually assigned to input or outputs and has internal address latches for the multiplexed address-data bus. The 8156 provides 256 bytes of RAM, a timer, three I/O ports (two 8-bit and one 6-bit) with handshaking capabilities, and internal address latches for the multiplexed address-data bus. The 74LS245 bus drivers provides robust tri-state buffering of the sixteen-bit RAM I/O ports to the (potentially heavily loaded) CMD/MON lines in the device logic. A third 74LS245 buffers the eight RA lines to the board and device logic.

The MCS-51 series of microcontrollers is designed for control applications and has programmable internal timer/counters and a programmable, buffered full-duplex serial port. In a typical application, one of the timers may be used for a baud rate generator for the serial port and the others as event counters or watch-dog timers. Because the timers and serial port are programmable, they can be used in many different modes. Other valuable features are the use of separate program and data memories, a six-source interrupt structure with two priority levels, 32 I/O lines, a Boolean processor and an Accumulator parity bit in the Program Status Word (convenient for serial transmissions). On-chip data memory contains both byte and bit storage capabilities. Data memory and I/O ports are bit addressable which makes it easy to manipulate discretes without having to do byte read/write and masking operations. I/O Port pins may be assigned to alternate functions such as interrupt inputs rather than standard I/O bit functions. Program and data memory are in separate address spaces and may be as large as 64 Kilobytes.

The 8032 description in this manual has the following restrictions. First, this is not an 8032, 8755 or 8156 data book and it does not contain a full description of all these chip's features and modes. The description is restricted to those features which are relevant to the Standard Interface Board application. A complete set of 8032, 8755 and 8156 data sheets is included in Section 6. Second, the 8032 instruction set and assembly language programming techniques are not discussed but the program is fully described. The details of programming the 8755 EPROM are not discussed.

The 8032 clock is an on-chip oscillator which uses an external 11.0592 Mhz crystal. A programmed subdivision of this clock rate by Timer 1 is used as the serial port transmit and receive clocks.

Figure 3, 8032 Block Diagram, is shown on the next page.

Special Function Registers (SFR's)

On-chip Special Function Registers (SFR's) control the operating modes of the timer/counters, serial port, and interrupts. Arguments and values for these functions are stored as data in the SFR's. Some SFR's are both bit and byte addressable (e.g. IP, IE, T2CON and SCON).

Timer/counters 0 and 1 have separate mode and control registers while Timer/counter 2 has a combined mode/control register. Thus TMOD establishes the modes of timers 0 and 1 and TCON turns the timers on and off. T2CON controls the modes and operation of T2. SCON controls both the mode and operation of the serial port. The SMOD bit in PCON causes the serial port baud rate to be doubled. Control bits in T2CON select Timer/counter 1 or 2 as the clock for the serial port. The IE register enables the interrupts (some of which are the result of timer overflow). The IP register establishes the priority of the six interrupt sources.

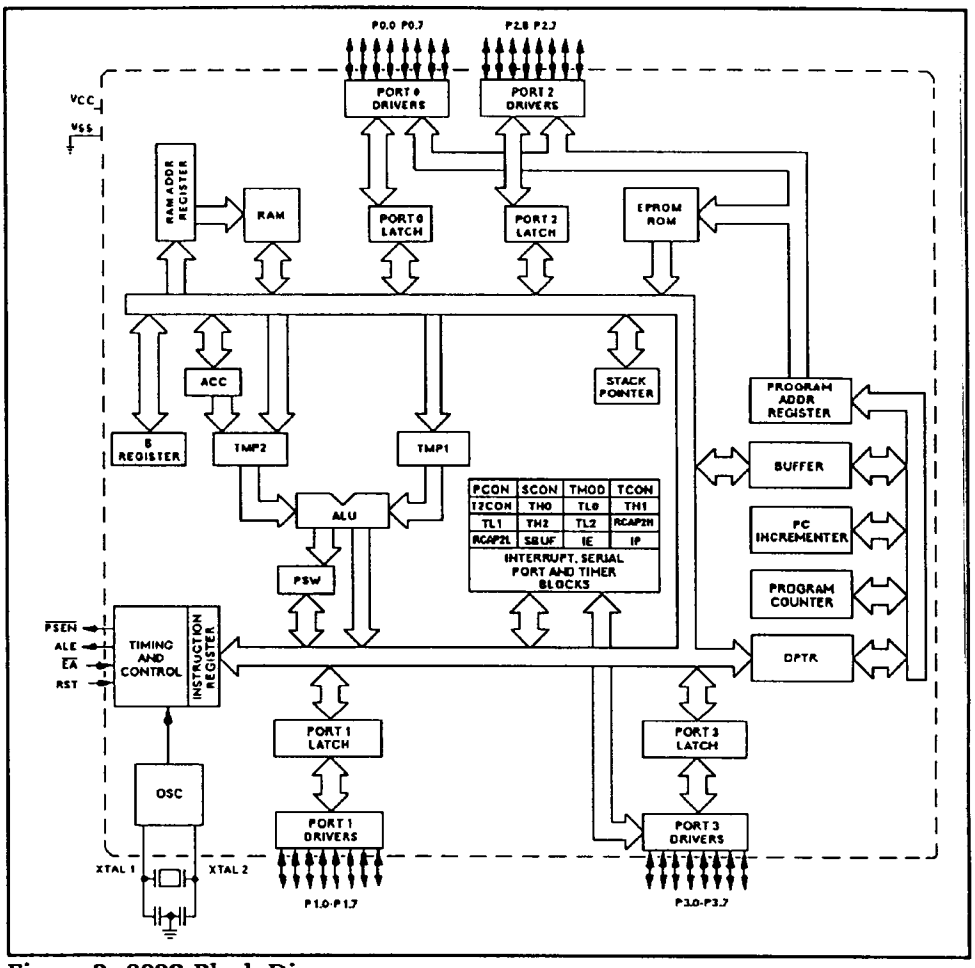


Figure 3, 8032 Block Diagram

The functions performed by these SFR mode and control registers will be described in more detail below. They are mentioned here to introduce their function.

CPU Timing

An 8032 machine cycle consists of 6 states (12 oscillator periods). Each state is divided into a Phase 1 half during which the Phase 1 clock is active and a Phase 2 half, during which the Phase 2 clock is active. Thus a machine cycle consists of 12 oscillator periods, numbered S1P1 (State 1, Phase 1) through S6P2 (State 6, Phase 2). Each phase lasts for two oscillator periods. Typically, arithmetic and logical operations take place during Phase 1 and internal register-to-register transfers take place during Phase 2.

The timing diagrams of Figure 4 (next page) show the instruction fetch/execute timing referenced to the internal states and phases. Since these internal clock signals are not user accessible, the XTAL2 oscillator signal and the ALE (Address Latch Enable) are shown for external reference. ALE is normally activated twice during each machine cycle: once during S1P2 and S2P1 and again during S4P2 and S5P1. **The trailing (falling edge) of ALE is used to latch the address for data read during the next cycle.**

At the board clock frequency of 11.0592 Mhz, the ALE period is 542 nanoseconds; this is half of a machine cycle.

Execution of a one-cycle instruction begins at S1P2, when the opcode is latched into the Instruction Register. If it is a two-byte instruction, the second byte is read during S4 of the same machine cycle. If it is a one-byte instruction, there is still a fetch at S4, but the byte read (which would be the next opcode) is ignored and the Program Counter is not incremented. In any case, execution is complete at the end of S6P2. Figures 4-A and 4-B (this page) show the timing for a 1-byte, 1-cycle instruction and for a 2-byte, 1-cycle instruction.

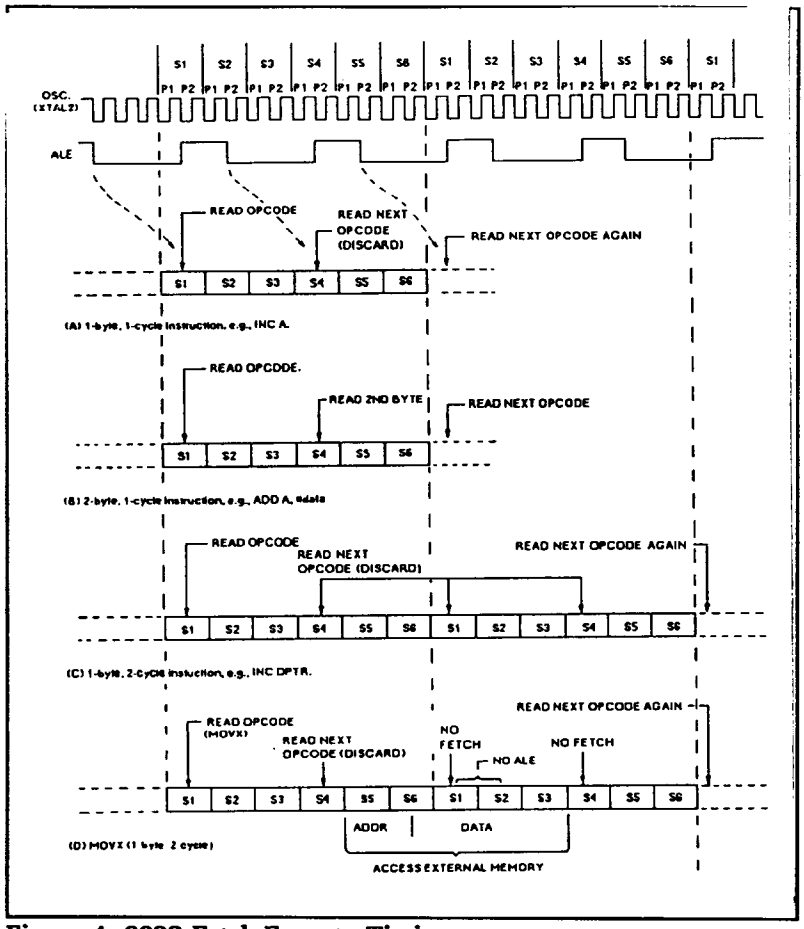


Figure 4, 8032 Fetch-Execute Timing

An important aspect of the Figure 4 timing diagram which is not emphasized in the Intel data books is that ALE is used for the **fetch for the next cycle.** The ALE that occurs at S1P2-S2P1 latches the address for data which is read at S4P2 and the ALE at S4P2-S5P1 latches the address for data read at S1P2 of the next cycle. This timing relationship is shown by the dashed lines on Figure 4.

Most 8032 instructions execute in one machine cycle. MUL (multiply) and DIV (divide) are the only instructions that take more than two machine cycles to complete; they take four cycles.

Normally, two code bytes are fetched from Program Memory during every machine cycle. The only exception to this is when a MOVX instruction is executed. MOVX is a 1-byte 2 cycle instruction that accesses external Data Memory. During a MOVX, two fetches are skipped while the external Data Memory is being addressed and strobed. Figures 4-C and 4-D show the timing for normal 1-byte, 2 cycle instruction and for a MOVX instruction.

I/O Ports

All four 8032 I/O ports (shown on the next page) are bidirectional. Each consists of a latch (SFR's P0 through P3), an output driver, and an input buffer. These ports are shown in Figure 5. In the Standard Interface Board application (with the exception of P3.3 and P3.4), only a few (P1.6, P1.7, P3.3 and P3.4) of the 8032 I/O pins are accessible. Since these inputs are not used in the Standard Interface Board application, the properties of these ports are not described in detail. For specifics on the port structures, refer to Sections 6.4.1, 6.4.2, 6.4.3 and 6.4.4 in the 8032 data sheets in Section 6. P3.3 and P3.4 are accessible on connector P2 but are not used in the VLBA Standard Interface Board application.

desiring to use these inputs are referred to the 8032 data sheet sections mentioned above.

Note that the Ports 1 and Port 3 alternate functions are set when the associated port latch bit is set.

The output drivers of Ports 0 and 2 and the input buffers of Port 0 are used in accesses to external memory. In this application, Port 0 outputs the low byte of the external memory address, time-multiplexed with the byte being written or read. Port 2 outputs the high byte of the external memory address when the address is 16 bits wide. Otherwise the Port 2 pins continue to emit the P2 SFR content.

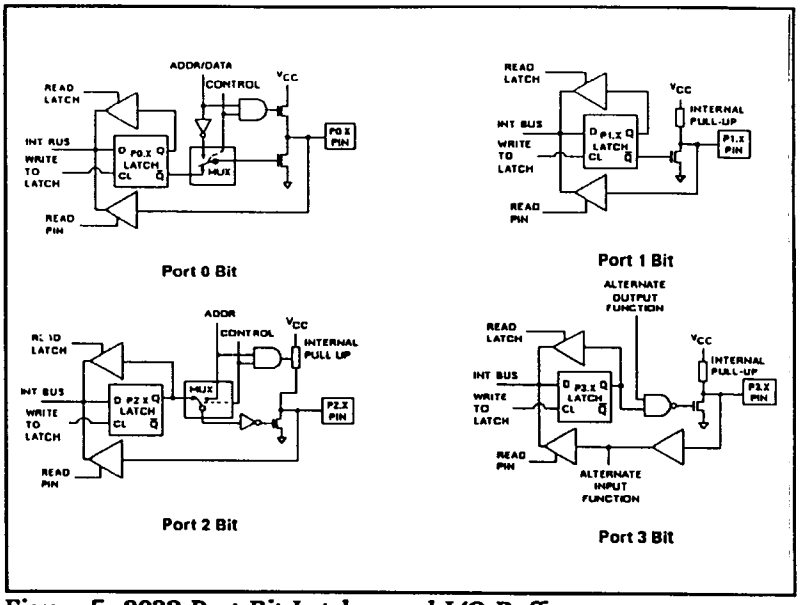


Figure 5, 8032 Port Bit Latches and I/O Buffers

The last three instructions in this list read the port byte, all 8 bits, modify the addressed bit, then write the new byte back to the latch.

All of the Port 3 pins and two Port 1 pins (P1.0 and P1.1) are multifunctional. They are not only port pins, but also serve the functions of various special features listed below. The alternate functions are selected when the associated port latch bits are set. Power reset sets all the alternate function latch bits.

P1.0 T2 (Timer/Counter 2 external imput. This function is not used in the Standard Interface Board.)
P1.1 T2EX (Timer/Counter 2 capture/reload trigger. This function is not used in the Standard Interface board).
P3.0 RXD (Serial Input port)
P3.1 TXD (Serial Output port)
P3.2 INT0- (External Interrupt, low-true. This pin is wired to P2 and is available for external interrupt use with firmware changes.)
P3.3 INT1- (External Interrupt, low-true. This pin is wired to P2 and is available for external interrupt use with firmware changes.)
P3.4 T0 (Timer/Counter 0 external input. This input is available on the Standard Interface Board P2 connector but T0 is not used in the standard firmware. With appropriate care taken for firmware timing, T0 could be used in special applications.)
P3.5 T1 (Timer/Counter 1 external input. T1 is dedicated for baud rate generation so this input is not used in the standard application. Other applications could use this input if the baud rate generation function is shifted to another timer/counter.)
P3.6 WR- (external Data Memory write strobe, low-true. Used to strobe the 8755 and 8156 WR- inputs when writing to the 8755 I/O ports and data direction registers, and the 8156 RAM and I/O ports and port command registers.)
P3.7 RD- (external Data Memory read strobe, low-true. Used to strobe the 8755 and 8156 RD- inputs when reading from the 8755 I/O ports and the 8156 RAM, I/O ports and port status register.)

In the Standard Interface Board application most of the Port 1 bits are used for control discretes for board functions such as enabling the RCV bus tri-state line driver. Port bits P1.6 and P1.7 are available for interfacing external equipment. This use requires alteration of the control firmware in the EPROM.

Some instructions that read a port read the latch and others read the pin. Which ones do which? The instructions that read the latch rather than the pin are the ones that read a value, possibly change it, and then rewrite it to the latch. These are called "read-modify-write" instructions. The instructions listed below are read-modify-write instructions. When the destination operand is a port, or a port bit, these instructions read the latch rather than the pin:

```
ANL         (Logical AND, e.g., ANL P1,A)
ORL         (Logical OR, e.g., ORL P2,A)
XRL         (Logical EX-OR, e.g., XRL P3,A)
JBC         (Jump if bit=1 and clear bit, e.g., JBC P1.1, LABEL)
CPL         (Complement bit, e.g., CPL P3.0)
INC         (Increment, e.g., INC P2)
DEC         (Decrement, e.g., DEC P2)
DJNZ        (Decrement and jump if not zero, e.g., DJNZ P3, LABEL)
MOV PX.Y,C  (Move carry bit to bit Y of Port X)
CLR PX.Y    (Clear bit Y of Port X)
Set PX.Y    (Set bit Y of Port X)
```

Accessing External Memory

Accesses to external memory (i.e., memory external to the 8032) are of two types: accesses to external Program Memory and accesses to external Data Memory. Accesses to external Program Memory use signal PSEN- (Program Store Enable) as the read strobe. Figures 6A and 6B (below) shows External Program and Data Memory read Timing. **Note that program data from the EPROM is read on the rise (leading edge) of PSEN-.** This signal is connected to the 8755 RD- pin to read the firmware program stored in the EPROM memory. Accesses to external Data Memory use RD- or WR- to strobe the memory. These signals are connected to the 8156 RD- and WR- pins to read and write to the RAM memory, I/O ports, and I/O ports command and status registers.

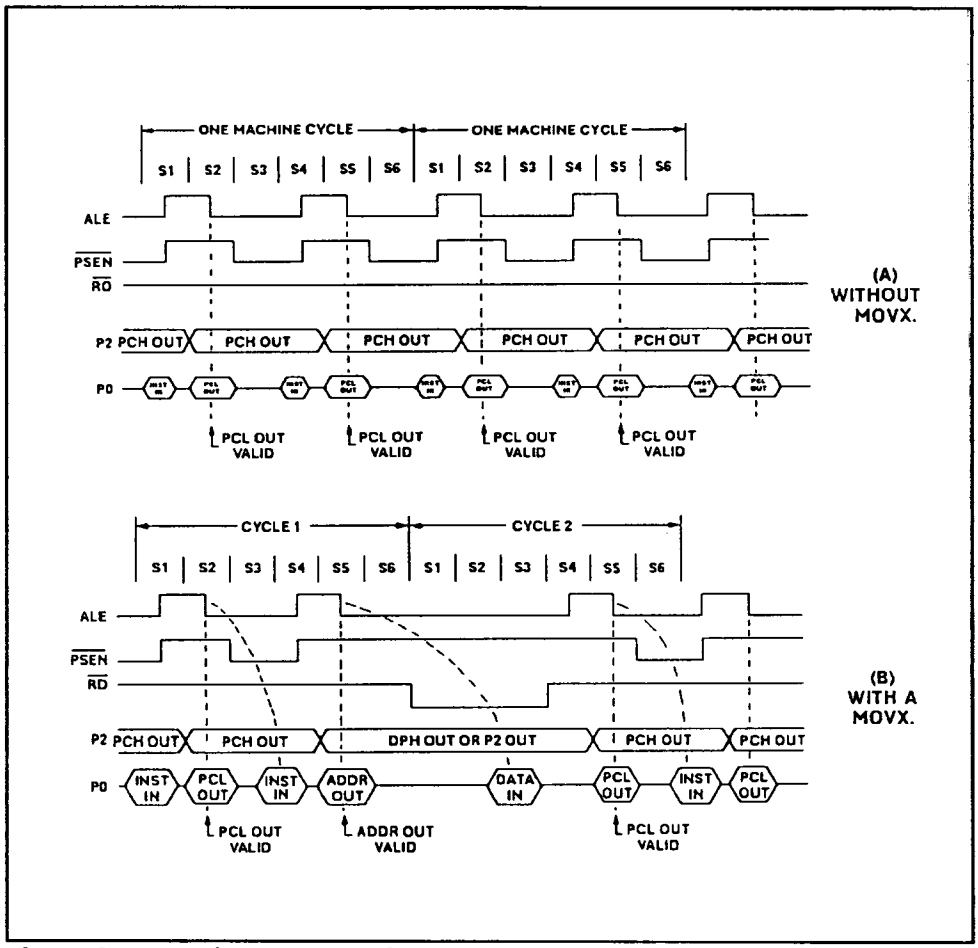


Figure 6, External Memory Fetch-Execute Timing

Like many other Intel microprocessors and microprocontrollers, the 8032 uses a multiplexed address

and data bus to the EPROM and RAM. Ports 0 and 2 are dedicated to this function.

Fetches from external Program Memory always use a 16-bit address. Accesses to external Data Memory can use either a 16-bit address (MOVX @DPTR) or an 8-bit address (MOVX @Ri).

Whenever a 16-bit address is used, the high byte of the address comes out on Port 2, where it is held for the duration of the read or write cycle. During this time the Port 2 latch (the Special Function Register) does not have to contain 1's and the contents of Port 2 SFR are not modified. If the external memory cycle is not immediately followed by another external memory cycle, the undisturbed contents of the Port 2 SFR will reappear in the next cycle.

If an 8-bit address is being used (MOVX @Ri), the contents of the Port 2 SFR remain at the Port 2 pins throughout the external memory cycle.

In any case, the low byte of the address is time-multiplexed with the data byte on Port 0. The ADDR/DATA signal drives both FET's in the Port 0 output buffers. (See Figure 6 above and Section 6.4 in the 8032 data sheets for additional details.) Signal ALE should be used to capture the address byte into an external latch. The address byte is valid at the negative transition of ALE. Then in a write cycle, the data byte to be written appears on Port 0 just before WRbar is activated and remains there until after WRbar is deactivated. Note that on the Standard Interface Board schematic diagram (drawing D55002S004), ALE is connected to the ALE inputs of the 8755 and 8156 to store the lower address byte in the chip latches.

PSEN (Program Store Enable)

The fetch/execute sequences are the same whether the Program Memory is internal or external to the chip. Execution times do not depend upon whether the Program Memory is internal or external.

The read strobe for external Program Memory fetches is PSEN-. **The 8032 reads program data on the rise (leading edge) of PSEN-.** When the CPU is accessing external program memory, PSEN- is activated twice every machine cycle (except during a MOVX instruction) whether or not the byte fetched is actually needed for the current instruction. A complete PSEN- cycle, including activation and deactivation of ALE and PSEN-, takes 6 oscillator periods. This timing is illustrated in Figure 6A in which there is not an access to external data memory. When PSEN- is activated, its timing is not the same as RD-.

If an access to external Data Memory occurs, as shown in Figure 6B, two PSEN-'s are skipped, because the address-data bus is being used for Data Memory access.

Note that an external Data Memory bus cycle takes twice as much time as a Program Memory bus cycle. This is probably done to provide a greater read time for external data memory since the 8032 does not have a wait pin. Figure 6B shows the relative timing of the addresses being emitted at Ports 0 and 2, and of ALE and PSEN-. ALE is used to latch the low address byte from P0 into the address latch.

The execution sequence for these two types of read cycles are shown in Figure 4 for comparison. The dashed lines in Figure 6B shows the ALE for External Program and External Data memory. In the Standard Interface application, interface board I/O lines are driven by the 8156 and 8755 I/O ports which are in the External Data memory address space. The timing for these I/O ports is thus identical to External Data memory timing. See the memory map in Section 3.2 for details on the address assignments.

ALE (Address Latch Enable)

The main function of ALE is to provide a properly timed signal to latch the low byte of an address from P0 to an external address latch during fetches from external Program Memory. For that purpose ALE is activated twice every machine cycle. The falling (i.e. trailing) edge of ALE is used as the latch strobe. This activation takes place even when the cycle involves no external fetch. Contrary to statements in earlier MCS-51 data books, ALE is used with external Data Memory cycles; any external memory read or write operation must have an address and ALE is the signal to latch the lower byte of the memory address. Figure 6B shows the timing for a MOVX instruction. ALE for reading the external Data Memory is shown by the dashed line. Cycle 2 of this instruction has only one ALE which is for the instruction fetch for the cycle following cycle 2. The first ALE of the second cycle of a MOVX instruction is missing (see Figure 6). Consequently, in any system that does not use external Data Memory, ALE is activated at a constant rate of 1/6 the oscillator frequency and can be used for external clocking or timing purposes.

Timer/Counters

The 8032 has three 16-bit timer/counter registers. All three can be configured to operate as timers or event counters.

In the "timer" function, the register is incremented every machine cycle. Thus one can think of it as counting machine cycles. Since a machine cycle consists of 12 oscillator periods, the count rate is 1/12 of the oscillator frequency.

In the "counter" function, the register is incremented in response to a 1-to-0 transition at its corresponding external input pin, T0, T1 or T2. In this function, the external input is sampled during S5P2 of every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register in the S3P1 of the cycle following the one in which the transition was detected. Since it takes 2 machine cycles (24 oscillator periods) to recognize a 1-to-0 transition, the maximum count rate is 1/24 of the oscillator frequency. There are no restrictions on the duty cycle of the exteral input signal but to insure that a given level is sampled at least once before it changes, it should be held for at least one full machine cycle.

In addition to the "timer" or "counter" selection, Timer 0 and Timer 1 have four operating modes from which to select. Timer 2 is has three modes of operation: "capture," "auto reload" and baud rate generator.

Timer/Counters 0 and 1

In the Standard Interface application, T0 is not used and T1 is used in mode 2 to generate the serial port baud rates. The description of modes 0, 1 and 3 is very brief to high-light the main features of these modes. Interface board users who intend to use the timers in these other modes are referred to the 8032 data sheets in Section 6 for a more complete description of the timer/counter modes.

Register TMOD specifies Timer/Counters 0 and 1 modes. Register TCON controls the operation (i.e. turns them on and off) of these two timer/counters.

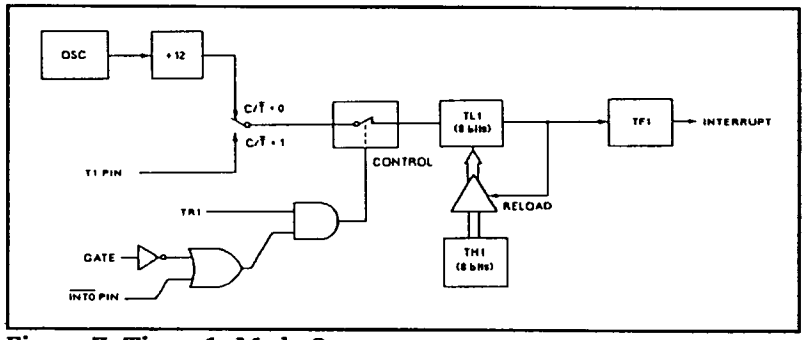


Figure 7, Timer 1, Mode 2

The TMOD register bit assignments are depicted below:

```
TMOD: Timer/Counters 0 and 1 Mode Register

----- Timer/Counter 0 -----      ----- Timer/Counter 1 -----
(MSB)                                                    (LSB)
GATE    C/T-    M1      M0      GATE    C/T-    M1      M0
0       0       1       0       0       0       0       0       <--- Value set by INITIAL routine
```

GATE Gating control. When set, Timer/Counter "X" is enabled only while INTX pin (Port pins 3.2 or 3.3) is high and TRX bit is set in TCON.

C/T- Timer or Counter selector, cleared for Timer operation (timer input from internal system clock). Set for Counter operation (counter input from TX input pin).

Figure 7 (previous page) shows the Timer 1 configuration.

T0 and T1 have four operating modes, which are selected by bit-pairs (M1, M0) in TMOD. Modes 0, 1, and 2 are the same for both counter/timers. Note that Mode 3 is different in that Timer/Counter 0 operates as two independent 8-bit counters. The four operating modes are shown below:

M1	M0	Operating Mode	
0	0	0	Timer TLX serves as a five-bit prescaler
0	1	1	16 bit Timer/Counter. THX and TLX are cascaded; there is no prescaler.
1	0	2	**8-bit auto-reload timer-counter. THX holds a value which is to be reloaded into TLX each time it overflows. Timer 1 operating in mode 2 generates the serial port transmit and receive clocks. (VLBA operating mode)**
1	1	3	(Timer 0) TL0 is an eight-bit timer/counter controlled by the standard Timer 0 control bits. TH0 is an eight-bit timer (only) controlled by Timer 1 control bits.
1	1	3	(Timer 1) Timer/counter 1 is stopped.

Mode 0 In mode 0, T0 and T1 are an 8-bit counter with a divide-by-32 prescaler. In this mode the timer register is configured as a 13-bit register. As the count rolls over from all 1's to all 0's, it sets the Timer Interrupt Flag, TF1 in TCON. For additional details on this mode see the Timer/Counters section of the 8032 data sheets.

Mode 1 Mode 1 is the same as Mode 0, except that the Timer register is being run with all 16 bits.

Mode 2 Mode 2 for Timer 1 is used to generate the 57.6 kilo-baud serial port shift clocks in the Standard Interface application. Mode 2 configures the timer register as an 8-bit counter (TL1) with automatic-reload as shown in Figure 7. Overflow from TL1 not only sets TF1 but also reloads TL1 with the contents of TH1, which is preset by software. The reload leaves TH1 unchanged. Mode 2 is the same for Timer/Counter 0. Figure 7 shows the configuration of Timer/Counter 1 in Mode 2. The application of Timer 1 for baud rate generation is found below in the description of the serial port. Figure 7 (previous page) shows the Timer 1, Mode 2 configuration.

Mode 3 Timer 1 in Mode 3 simply holds its count. The effect is the same as setting TR1 = 0 in TCON. Timer 0 in Mode 3 establishes TL0 and TH0 as two separate counters. For additional details on this mode see the Timer/Counter section of the 8032 data sheets.

Timer/Counters 0 and 1 are controlled by the TCON register. The TMOD register determines the timer/counter modes. The TCON register bit assignments are shown on the next page.

TCON: Timer/Counter 0 and 1 Control Register

(MSB) TF1	TR1	TF0	TR0	IE1	IT1	IE0	(LSB) IT0

Symbol	Position	Name and Significance
TF1	**TCON.7**	**Timer 1 overflow Flag. Set by hardware on timer/counter overflow. Cleared by hardware when processor vectors to interrupt routine.**
TR1	**TCON.6**	**Timer 1 Run control bit. Set/cleared by software to turn timer/counter on/off.**
TF0	TCON.5	Timer 0 overflow Flag. Set/cleared in the same manner as TF1 above.
TR0	TCON.4	Timer 0 Run control bit. Set/cleared in the same manner as TR1 above.
IE1	TCON.3	Interrupt 1 Edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.
IT1	TCON.2	Interrupt 1 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.
IE0	TCON.1	Interrupt 0 Edge flag. Set and cleared in the same manner as IE1 above.
IT0	TCON.0	Interrupt 0 Type control bit. Set and cleared in the same manner as IT1 above.

Bold lettering in the TCON table indicates active bits in the Standard Interface Board application.

Timer 2 (Watch-Dog Timer)

Timer/Counter 2 is implemented in only the 8032 and 8052 versions of the MCS-51 family. Timer/counter 2 is controlled by SFR register T2CON. In the Standard Interface Board application, Timer 2 is used as a watch-dog timer and operates in the "auto-reload" mode.

Timer 2 is a 16-bit timer/counter. Like Timers 0 and 1 it can operate either as timer or as an event counter. The function is selected by bit C/T- in the Special Function Register T2CON. Timer 2 has three operating modes: "capture", "auto-reload" and "baud rate generator," which are selected by bits in TCON as shown below.

The configuration of Timer/Counter 2 in the Auto-Reload Mode is shown in Figure 8, below:

In the capture mode there are two options selected by bit EXEN2 in TCON. If EXEN2 = 0, then timer 2 is a 16 bit timer or counter which upon overflowing sets bit TF2 (the Timer 2 overflow bit) which can be used to generate an interrupt. If EXEN2 = 1, then Timer 2 generates an interrupt and in addition, a 1-to-0 transition at external input T2EX causes the current value in the Timer 2 registers, TL2 and TH2 to be captured into registers RCAP2L and RCAP2H, respectively. In addition, the transition at T2EX causes bit T2EX in TCON to be set, and EXF2, like TF2, can generate an interrupt.

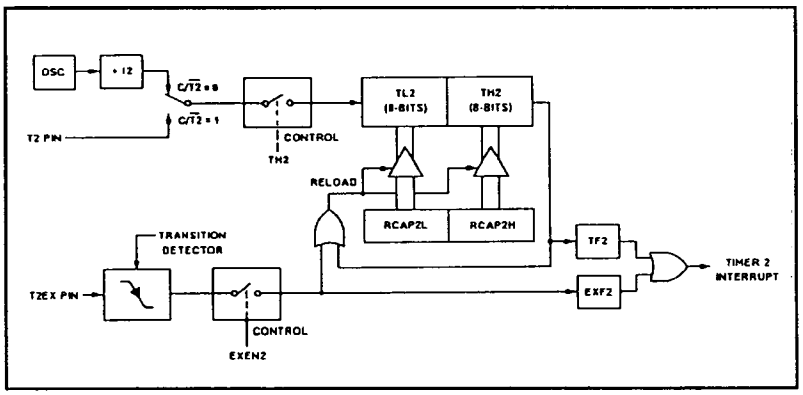


Figure 8, Timer 2, Auto-Reload Mode

The capture mode configuration of Timer 2 is shown in Section 6 of the 8032 data sheets in Section 6.

In the auto-reload mode there are also two options, which are selected by bit EXEN2 in T2CON. If EXEN2 = 0, when Timer 2 rolls over it not only sets TF2, but also causes the Timer 2 registers to be reloaded with the 16-bit values in RCAP2L and RCAP2H, which are preset by software. If EXEN2 = 1, then Timer 2 still does the above, but with the added feature that a 1-to-0 transition at external input T2EX will also trigger the 16-bit reload and set EXF2. The auto-reload configuration of Timer 2 is described in more detail in Section 6.6 of the 8032 data sheets.

Referring to the TCON; Timer/Counter 2 Control Register figure below, the baud rate generator mode for Timer 2 is selected by RCLK = 1 and TCLK = 1 in T2CON. Note that the baud rates for transmit and receive can be simultaneously different. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode.

The Timer 2 baud rate generator mode is similar to the auto-reload mode in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit values in RCAP2H and RCAP2L, which are preset by software. For a full description of the operation of Timer 2 see Section 6.7.3 in the 8032 data sheets in Section 6.

T2CON: Timer/Counter 2 Control Register

(MSB)							(LSB)	
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2bar	CP/RL2bar	
0	0	0	0	0	1	0	0	<--- Value set by INITIAL routine

Symbol	Position	Name and Significance
TF2	**T2CON.7**	**Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.**
EXF2	T2CON.6	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software.
RCLK	**T2CON.5**	Receive clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in modes 1 and 3. **RCLK = 0 causes Timer 1 overflow to be used for the receive clock.**
TCLK	**T2CON.4**	Transmit clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in modes 1 and 3. **TCLK = 0 causes the Timer 1 overflows to be used for the transmit clock.**
EXEN2	T2CON.3	Timer 2 external enable flag. When set , allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore the events at T2EX.
TR2	**T2CON.2**	**Start/stop control for Timer 2. A logic 1 starts the timer.**
C/T2-	T2CON.1	Timer or counter select (Timer 2) 0 = Internal timer (OSC/12) 1 = External event counter (falling edge triggered).
CP/RL2-	**T2CON.0**	Capture/Reload flag. When set, captures will occur on negative transitions at T2EX if EXEN2 = 1. **When cleared, auto reloads will occur either with Timer 2 overflows** or negative transitions at T2EX when EXEN2 = 1. When either RCLK = 1 or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

Timer 2 has four operating modes; these are tabulated on the next page.

RCLK + TCLK	CP/RL2bar	TR2	MODE
0	0	1	**16-bit auto-reload**
0	1	1	16-bit capture
1	X	1	baud rate generator
X	X	0	(off)

Serial Port Interface

The serial port is full duplex, meaning it can transmit and receive simultaneously. It is also receive-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the receive register. (However, if the first byte still hasn't been read by the time reception of the second byte is complete, one of the bytes will be lost). The serial port receive and transmit registers are both accessed at Special Function Register SBUF. Writing to SBUF loads the transmit register, and reading SBUF accesses a physically separate receive register.

The serial port is controlled by SFR register SCON which is shown below:

SCON: Serial Port Control Register

(MSB)							(LSB)	
SM0	SM1	SM2	REN	TB8	RB8	TI	RI	
1	1	0	1	0	0	0	0	<--- Value set by INITIAL routine

SM2 enables the multiprocessor communication feature in modes 2 and 3. In Mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In Mode 1, if SM2 =1 then RI will not be activated if a valid stop bit was not received. In Mode 0, SM2 should be 0.

REN enables serial reception, set by software. Clear by software to disable reception.

TB8 is the 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software.

RB8 in Modes 2 and 3, is the 9th data bit that was received. In Mode 1, if SM2 = 0, RB8 is the stop bit that was received. In Mode 0, RB8 is not used.

TI is the transmit interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or at the beginning of the stop bit in the other modes in any serial transmission. TI must be cleared by software.

RI is the receive interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM2). Must be cleared by software.

SM0,SM1 specify the serial port mode as follows:

SM0	SM1	Mode	Description	Baud Rate
0	0	0	shift register	$f_{osc.}/12$
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	$f_{osc.}/64$ or $f_{osc.}/32$*
1	1	3	9-bit UART	variable

* Determined by SMOD value in PCON. **Bold print denotes the Standard Interface mode.**

The serial port can operate in 4 modes. **(Mode 3 is used in the Standard Interface Board application).**

Mode 0: Serial data enters and exits through RXD. TXD outputs the shift clock. 8 bits are transmitted/received: 8 data bits (LSB first). The baud rate is fixed at 1/12 the oscillator frequency.

Mode 1: 10 bits are transmitted (through TXD) or received (through RXD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB8 in Special Function Register SCON. The baud rate is variable.

Mode 2: 11 bits are transmitted (through TXD) or received (through RXD). This mode is identical to Mode 3 except for the baud rate which is either 1/32 or 1/64 the oscillator frequency.

Mode 3: (Standard Interface mode). 11 bits are transmitted (through TXD) or received (through RXD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On transmit, the 9th data bit (TB8 in SCON) can be assigned the value 0 or 1. Or, for example, the parity bit (P, in the PSW) could be moved into TB8. On receive, the 9th data bit goes into RB8 in Special Function Register SCON, while the stop bit is ignored. The baud rate is variable and determined by Timer/Counters 1 or 2. In the Standard Interface application, Timer 1 is used as the baud rate generator.

In all four modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in Mode 0 by the condition RI = 0 and REN = 0 in SCON. Reception is initiated in the other modes by the incoming start bit if REN = 1.

The serial port control and status register is the Special Function Register SCON, shown below. This register contains not only the mode selection bits but also the 9th data bit for transmit and receive. (TB8 and RB8), and the serial port interrupt bits (TI and RI). SCON is shown below.

Baud Rates

The serial port baud rate in Mode 0 is fixed at $f_{osc.}/12$.

The serial port baud rate in Mode 2 depends on the value of bit SMOD in Special Function Register PCON (shown below). If SMOD = 0 (which is its value on reset), the baud rate is 1/64 the oscillator frequency. If SMOD = 1, the baud rate is 1/32 the oscillator frequency.

In Modes 1 and 3, the serial port baud rates can be determined by Timer 1, or by Timer 2, or by both (one for transmit and the other for receive).

When Timer 1 is used as the baud rate generator (as in the Standard Interface Board application), the baud rate in Modes 1 and 3 are determined by the Timer 1 overflow rate and the value of SMOD (in PCON) as follows:

Baud Rate = $(2^{SMOD}/32)$ x (Timer 1 Overflow Rate) The Timer 1 interrupt should be disabled in this application. The timer itself can be configured for either "timer" or "counter" operation, in any of its 3 operating modes. In the most typical application, it is configured for "timer" operation, in the auto-reload mode (high nibble of TH1 = 0010B). In that case, the baud rate is given by the formula:

Baud Rate = $(2^{SMOD}/32)$ x (Oscillator Frequency)/(12 x [256 - (TH1)]) In the Standard Interface application, SMOD = 1, the oscillator frequency is 11.0592 Mhz and TH1 is loaded with a value of 255. These values produce a baud rate of 57.6 kilo-hz.

It is possible to produce very low baud rates with Timer 1 by leaving the Timer 1 interrupt enabled, configuring the timer to run as a 16-bit timer (high nibble of TH1 = 0010B), and using the Timer 1 interrupt to do a 16-bit software reload.

Figure 9 (next page) shows a functional diagram of the serial port and timing in Modes 2 and 3.

Timer 2 can also be used to generate baud rates but this function will not be described here since it is not used in the Standard Interface application.

Transmission is initiated by any instruction that uses SBUF as a destination register. The "write to SBUF" signal also loads TB8 into the 9th position of the transmit shift register and flags the TX Control unit that a transmission is requested. Transmission commences at S1P1 of the machine cycle following the next rollover in the divide-by-16 counter. (Thus the bit times are synchronized to the divide-by-16 counter, not to the "write to SBUF" signal).

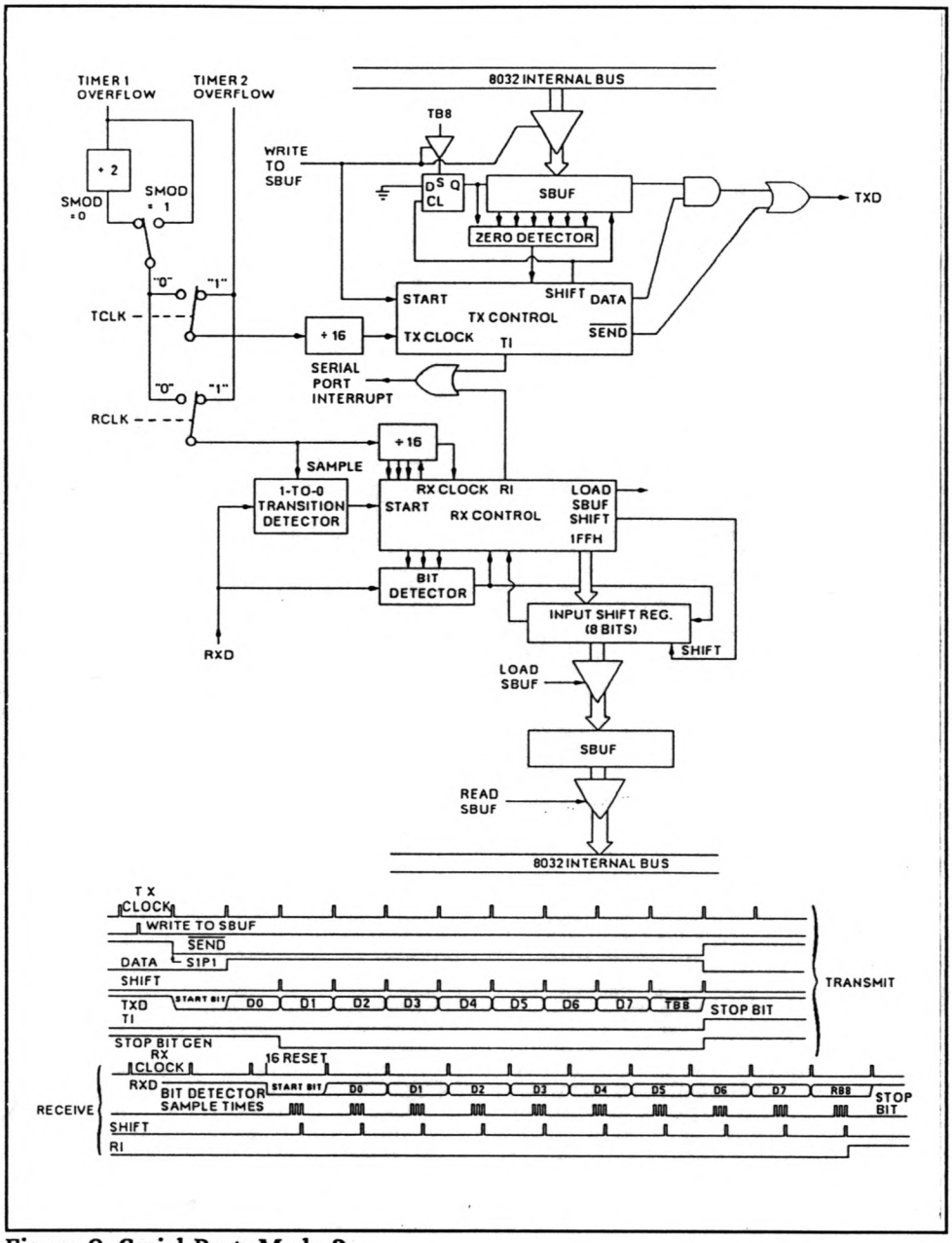


Figure 9, Serial Port, Mode 3

Transmission begins with activation of SEND, which puts the start bit at TXD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TXD. The first shift pulse occurs one bit time after that. The first shift clocks a 1 (the stop bit) into the 9th bit position of the shift register. Thereafter, only zeroes are clocked in. Thus as data bits shift out to the right, zeroes are clocked in from the left. When TB8 is at the output position of the shift register, then the stop bit is just to the left of TB8, and all positions to the left of that contain zeroes. This condition flags the TX Control unit to do one last shift and then deactivate SEND- and set TI. This occurs at the 11th divide-by-16 rollover after the "write to SBUF."

Reception is initiated by a detected 1-to-0 transition at RXD. For this purpose RXD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and 1FFH is written to the input shift register.

At the 7th, 8th and 9th counter states of each bit time, the bit detector samples the value of RXD. The value accepted is the value that was seen in at least 2 of the 3 samples. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit reverts back to looking for another 1-to-0 transition. If the start bit proves valid, it is shifted into the input shift register and reception of the rest of the frame will proceed.

As data bits come in from the right, 1's shift out to the left. When the start bit arrives at the leftmost position in the shift register (which in Modes 2 and 3 is a 9-bit register), it flags the RX Control block to do one last shift, load SBUF and RB8, and set RI. The signal to load SBUF and RB8, and to set RI, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated:

1) RI = 0, and 2) Either SM2 = 0 or the 9th data bit = 1.

If either of these condition are not met, the received frame is irretrievably lost and RI is not set. If both conditions are met, the received 9th data bit goes into RB8 and the first 8 data bits go into SBUF. One bit time later, whether the above conditions were met or not, the unit goes back to looking for a new 1-to-0 transition at the RXD input.

Note that the value of the received stop bit is irrelevant to the SBUF, RB8, or RI.

The VLBA Standard Interface Board has been used in other applications at 19.2 and 1.2 Kilo-baud rates. To program standard rates (in Modes 1 or 3 with the board's 11.0592 Mhz crystal), the TH1 and SMOD values in the equation above should be set to the values shown in the table below. See Sections 2.7 (Baud Rates) and 3.3 (Firmware, SPOOUT) for a description of the effect of baud rate changes on the firmware.

	Timer 1			
Baud Rate	SMOD	C/Tbar	Mode	TH1
19.2 K	1	0	2	FDH
9.6 K	0	0	2	FDH
4.8 K	0	0	2	FAH
2.4 K	0	0	2	F4H
1.2 K	0	0	2	E8H

Interrupts

The Interrupt Control System is shown in Figure 10 (next page). As shown on Figure 10, the 8032 provides 6 interrupt sources.

The External Interrupts INT0bar and INT1bar can be either level-activated or transition-activated, depending on bits IT0 and IT1 in TCON. The flags that actually generate these interrupts are bits IE0 and IE1 in TCON. When an external interrupt is generated, the flag that generated it is cleared by the hardware when the service routine is vectored to only if the interrupt was transition-activated. If the interrupt was level-activated, then the external requesting source is what controls the request flag, rather than the on-chip hardware.

The Timer 1 and Timer 0 Interrupts are generated by TF0 and TF1, which are set by a rollover in their respective timer/counter registers (except see the 8032 data sheets for Timer 0 in Mode 3). When a timer interrupt is generated, the flag that generated it is cleared by the on-chip hardware when the service routine is vectored to.

The Serial Port Interrupt is generated by the logical OR of RI and TI in the serial port control

logic. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine will normally have to determine whether it was RI or TI that generated the interrupt, and the bit will have to be cleared in software.

The Timer 2 interrupt is generated by the logical OR of RF2 and EXF2. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine may have to determine whether it was TF2 or EXF2 that generated the interrupt, and the bit will have to be generated in software.

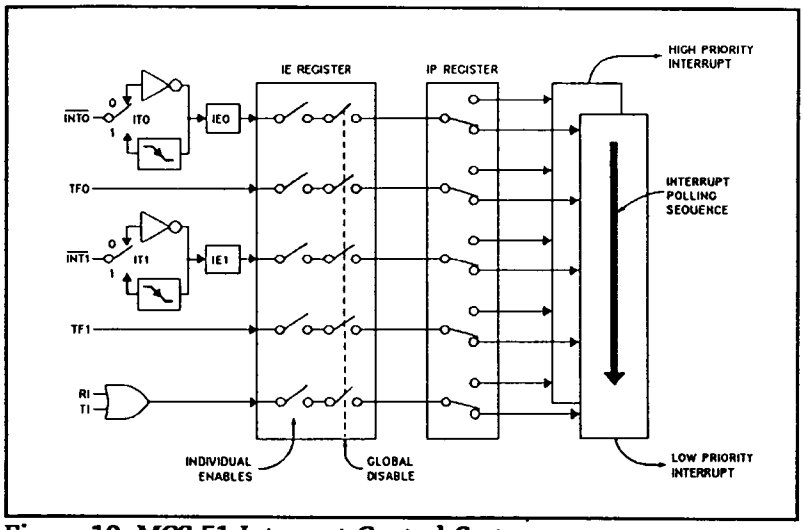


Figure 10, MCS-51 Interrupt Control System

All of the bits that generate interrupts can be set or cleared by software, with the same result as though it had been set or cleared by hardware. That is, interrupts can be generated or pending interrupts can be cancelled in software.

Each of these interrupts can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE (below). Note that IE contains also a global disable bit, EA, which disables all interrupts at once.

IE: Interrupt Enable Register

(MSB)							(LSB)
EA	X	ET2	ES	ET1	EX1	ET0	EX0

Symbol	Position	Function
EA	IE.7	Disables all interrupts if EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
-	IE.6	Reserved
ET2	IE.5	Enables or disables the Timer 2 overflow or capture interrupt. If ET2 = 0, the Timer 2 interrupt is disabled.
ES	IE.4	Enables or disables the Serial Port interrupt. If ES = 0, the Serial Port interrupt is disabled.
ET1	IE.3	Enables or disables the Timer 1 overflow interrupt. If ET1 = 0, the Timer 1 interrupt is disabled.
EX1	IE.2	Enables or disables External Interrupt 1. If EX1 = 0, External Interrupt 1 is disabled.
ET0	IE.1	Enables or disables the Timer 0 Overflow Interrupt. If ET0 = 0, the Timer 0 Interrupt is disabled.
EX0	IE.0	Enables or disables External Interrupt 0. If EX0 = 0, External Interrupt 0 is disabled.

Priority Level Structure

Each interrupt source can also be individually programmed to one of two priority levels by setting or clearing a bit in Special Function Register IP, shown below. A low-priority interrupt can itself be interrupted by a high-priority interrupt but not by another low-priority interrupt. A high-priority interrupt can't be interrupted by any other interrupt source.

IP: Interrupt Priority Register

(MSB)						(LSB)
X	X	PT2	PS	PT1	PT0	PX0

Symbol	Position	Function
-	IP.7	Reserved
-	IP.6	Reserved
PT2	IP.5	Defines the Timer 2 Interrupt priority level. PT2 = 1 programs it to the higher priority level.
PS	IP.4	Defines the Serial Port Interrupt priority level. PS = 1 programs it to the higher priority level.
PT1	IP.3	Defines the Timer 1 Interrupt priority level. PT1 = 1 programs it to the higher priority level.
PX1	IP.2	Defines the External Interrupt priority level. PX = 1 programs it to the higher priority level.
PT0	IP.1	Defines the Timer 0 Interrupt priority level. PT0 = 1 programs it to the higher priority level.
PX0	IP.0	Defines the External Interrupt priority level. EX0 = 1 programs it to the higher priority level.

If two requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence as follows:

	SOURCE	PRIORITY WITHIN LEVEL	VECTOR ADDRESS
1.	IE0	(highest)	0003H
2.	TF0		000BH
3.	IE1		0013H
4.	TF1		001BH
5.	RI + TI		0023H
6.	TF2 + EXF2	(lowest)	002BH

Note that the "priority within level" structure is only used to resolve simultaneous requests of the same priority level.

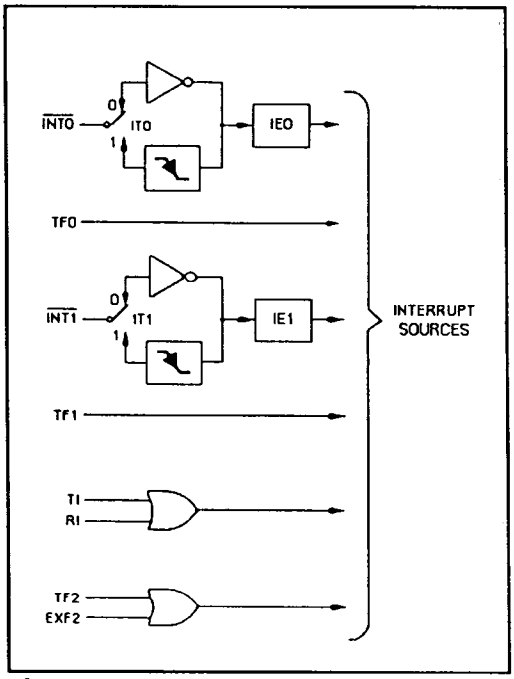


Figure 11, Interrupt Sources

How Interrupts Are Handled

The interrupt flags are sampled at S5P2 of every machine cycle. The samples are polled during the following machine cycle. If one of the flags was in a set condition at S5P2 of the preceding cycle, the polling cycle will find it and the interrupt system will generate an LCALL to the appropriate service routine, provided this hardware-generated LCALL is not blocked by any of the following conditions:

1. An interrupt of equal or higher priority level is already in progress.

2. The current (polling) cycle is not the final cycle in the execution of the instruction in progress.

3. The instruction in progress is RETI or any access to the IE or IP registers.

Any of these three conditions will block the generation of the LCALL to the interrupt service routine. Condition 2 ensures that the instruction in progress will be completed before vectoring to any service routine. Condition 3 ensures that if the instruction in progress is RETI or any access to IE or IP, then at least one more instruction will be executed before any interrupt is vectored to.

The polling sequence is repeated with each machine cycle, and the values polled are the values that were present at S5P2 of the previous machine cycle. Note that if an interrupt flag is active but not being responded to for one of the above conditions, if the flag does not continue to be active when the blocking condition is removed, the denied interrupt will not be serviced. The interrupt logic does not store the occurrance of unserviced interrupts. Every polling cycle is new. The user must take this potential loss of unserviced interrupts into account in the logic design.

The polling cycle/LCALL sequence is illustrated in Figure 12.

Note that if an interrupt of higher priority level goes active prior to S5P2 of the machine cycle labeled C3 in Figure 11, then in accordance with the above rules it will be vectored to during C5 and C6 without any instruction of the lower priority routine having been executed.

Thus the processor acknowledges an interrupt request by executing a hardware-generated LCALL to the appropriate servicing routine. In some cases it also clears the flag that generated the interrupt, and in other cases it doesn't. It never clears the Serial Port or Timer 2 flags. This has to be done in the user's software. It clears an external interrupt flag (IE0 or IE1) only if it was transition-activated. The hardware-generated LCALL pushes the contents of the Program Counter onto the stack (but does not save the PSW) and reloads the PC with an address that depends on the source of the interrupt being vectored to, as shown on the previous page.

Execution proceeds from that location until the RETI instruction is encountered. The RETI instruction informs the processor that this interrupt routine is no longer in progress, then pops the top two bytes from the stack and reloads the Program Counter. Execution of the interrupted program continues from where it left off.

Note that a simple RET instruction would also have returned execution to the interrupted program, but it would have left the interrupt control system thinking an interrupt was still in progress.

External Interrupts

The external sources can be programmed to be level-activated or transition-activated by setting or clearing bit IT1 or IT0 in Register TCON. If ITx = 0, external interrupt x is triggered by a detected

low at the INTx- pin. If INTx- = 1, external interrupt x is edge-triggered. In this mode if sucessive samples of the INTx- pin show a high in one cycle and a low in the next cycle, interrupt request flag IEx in TCON is set. Flag bit IEx then requests the interrupt.

Since the external interrupt pins are sampled once each machine cycle, an input high or low should hold for at least 12 oscillator periods to ensure sampling. If the external interrupt is transition-activated, the external source has to hold the request pin high for at least one cycle, and then hold it low for at least one cycle to ensure that the transition is seen so that the interrupt request flag IEx will be set. IEx will be automatically cleared by the CPU when the service routine is called.

If the external interrupt is level-activated, the external source has to hold the request active until the requested interrupt is actually generated. Then it has to deactivate the request before the service routine is completed or else another interrupt will be generated.

Response Time

The INT0- and INT1- levels are inverted and latched into IE0 and IE1 at S5P2 of every machine cycle. The values are not actually polled by the circuitry until the next machine cycle. If a request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The call itself takes two cycles. Thus a minimum of three complete machine cycles elapse between activation of an external interrupt request and the beginning of execution of the first instruction of the service routine. Figure 12 shows the interrupt response timings.

A longer response time would result if the request is blocked by one of the previously listed conditions. If an interrupt of equal or higher priority level is already in progress, the additional wait time obviously depends on the nature of the other interrupt's service routine. If the instruction in progress is not in the final cycle, the additional wait time cannot be more than 3 cycles, since the longest instructions (MUL AND DIV) are only 4 cycles long. If the instruction in progress is RETI or an access to IE or IP, the additional wait time cannot be more than 5 cycles (a maximum of one more cycle to complete the instruction in progress, plus 4 cycles to complete the next instruction if the instruction is MUL or DIV). Thus, in a single-interrupt system the response time is always more than 3 cycles and less than 8 cycles.

Accumulator Register

The Accumulator (usually designated A in instruction mnemonics and ACC elsewhere) is the primary data manipulation register and is used in logical, numeric and data transfer operations. It is also the source or destination register for I/O operations.

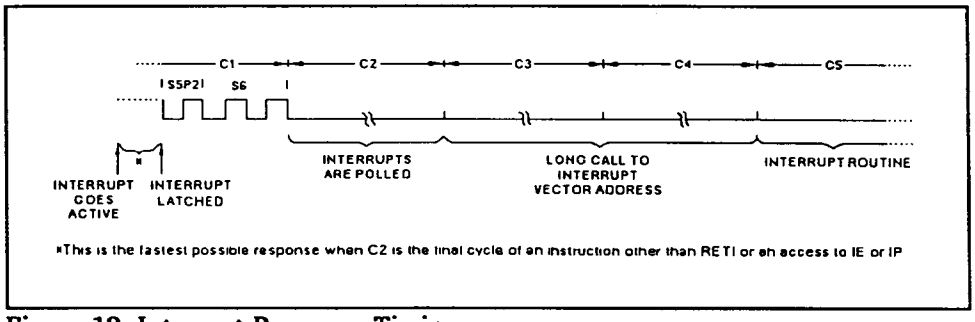


Figure 12, Interrupt Response Timing

B Register

The B register is used during multiply and divide operations. For other instructions it can be treated as another scratch pad register.

Stack Pointer

The STACK POINTER register is 8 bits wide. It is incremented before data is stored during PUSH and CALL executions. While the STACK may reside anywhere in on-chip RAM, the STACK POINTER is initialized to 07H after a reset. This causes the STACK to begin at location 08H.

Data Pointer

The DATA POINTER (DPTR) consists of a high byte (DPH) and a low byte (DPL). It is intended function is to hold a 16-bit address. It may be manipulated as a 16-bit register or as two independent 8-bit registers.

Serial Data Buffer

The Serial Data Buffer (SBUF) is actually two separate registers, a transmit buffer and a receive buffer register. When data is moved to SBUF, it goes to the transmit buffer where it is held for serial transmission. (Moving a byte to SBUF is what initiates the transmission). When data is moved from SBUF, it comes from the receive buffer.

Program Status Word

The Program Status Word (PSW) is a very important SFR. Some of the PSW flag bits are used extensively in the Standard Interface application. PSW.5 is used as the "LOST" flag to detect occurrances of program perturbations caused by power glitches. The Parity flag P (PSW.0) is used in formulating and testing serial message byte parity. RS0 and RS1 select the working register bank (only Bank 0 is used in the Standard Interface application). The CY flag is used in Boolean and arithmetic operations. The format of the PSW is shown below.

```
        (MSB)                                                   (LSB)
        CY      AC      F0      RS1             RS0     OV      -       P
```

Symbol	Position	Name and Significance	Symbol	Position	Name and Significance
CY	PSW.7	Carry flag	OV	PSW.2	Overflow flag
AC	PSW.6	Auxiliary carry (for BCD operations)	-	-	PSW.1 User defineable flag
			P	PSW.0	Parity flag (P) Set/cleared by hardware each instruction cycle to indicate an odd/even number of "one" bits in the Accumulator. i.e., even parity
F0	PSW.5	Flag 0 Used for the "LOST" flag in the Std Int firmware.			
RS1	PSW.4	Register bank select bits 1 & 0 Set/cleared by firmware to determine working register bank. (See note below).			
RS0	PSW.3				

Note:
The contents of (RS1, RS0) enable the working register banks as follows:

```
        (0,0) -- Bank 0  (00H - 7FH)
        (0,1) -- Bank 1  (08H - 0FH)
        (1,0) -- Bank 2  (10H - 17H)
        (1,1) -- Bank 3  (18H - 1FH)
```

Timer Registers

Register pairs (TH), TL0), (TH1, TL1), and (TH2, TL2) are the 16-bit Counting registers for Timer/Counters 0, 1, and 2 respectively.

Capture Registers

The register pair (RCAP2H, RCAP2L) hold the the 16-bit reload value for the "Auto-Reload" mode of Timer 2. This is the Timer 2 mode used in the Standard Interface application. In the "Capture Mode" TH2 and TL2 are copied into RCAP2H and RCAP2L in response to a transition at the T2EX pin. A more detailed description of the Timer 2 Capture Mode is found in the 8032 data sheets in Section 6.

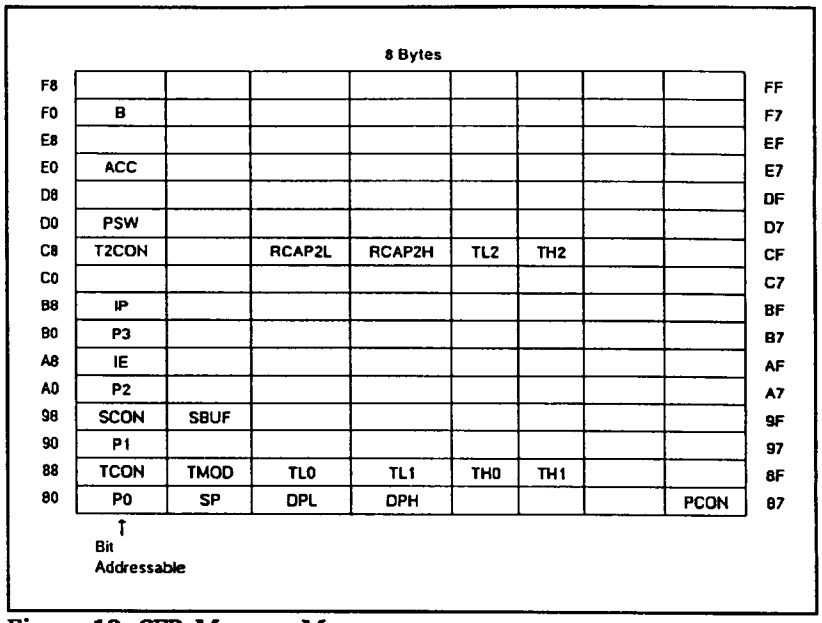


Figure 13, SFR Memory Map

SFR Memory Map

The SFR memory map is shown on this page. Note that the left column contains the bit-addressible SFR and memory locations. Note also that in the Standard Interface application, only TL0 and TH0 are unused.

Reset

The reset input is the RST pin, which is the input to a Schmidt Trigger. A reset is accomplished by holding the RST pin high for at least two machine cycles (24 oscillator periods), while the oscillator is running. The CPU responds by executing an internal reset. It also configures the ALE and PSEN- pins as inputs (they are quasi-bidirectional). The internal reset is executed during the second cycle in which RST is high and is repeated every cycle until RST goes low. It leaves the internal registers as shown on the next page:

REGISTER	CONTENT	REGISTER	CONTENT	REGISTER	CONTENT
PC	0000H	IE	(XX000000B)	TH2	00H
ACC	00H	TMOD	00H	TL2	00H
B	00H	TCON	00H	RCAP2H	00H
PSW	00H	T2CON	00H	RCAP2L	00H
SP	07H	TH0	00H	SCON	00H
DPTR	0000H	TL0	00H	SBUF	Indeterminate
P0 - P3	0FFH	TH1	00H	PCON	(0XXX0000B)
IP	(XX000000B)	IE	(0X000000B)		

The internal RAM is not affected by reset. When VCC is turned on, the RAM content is indeterminate unless the 8032 is returning from a reduced power mode of operation. The reduced power mode of operation is not used in the Standard Interface Board application but one bit of the PCON power control Register in the Special Function Registers is important. This bit is SMOD which is bit PCON.7. It causes the serial port baud rate to be doubled when set. For completeness, the PCON register is shown on the next page.

PCON: Power Control Register

(MSB)							(LSB)	
SMOD	-	-	-	GF1	GF0	PD	IDL	
1	0	0	0	0	0	0	0	<-- Value set by INITIAL routine

Symbol	Position	Name and Function
SMOD	PCON.7*	Double Baud rate bit. When set to 1, the baud rate is doubled when the serial port is being used in either Modes 1, 2 or 3.
-	PCON.6	(Reserved)
-	PCON.5	(Reserved)
-	PCON.4	(Reserved)
GF1	PCON.3	General purpose flag bit.**
GF0	PCON.2	General purpose flag bit.**
PD	PCON.1	Power Down bit. Setting this bit activates power down operation.**
IDL	PCON.0	Idle mode bit. Setting this bit activates idle mode operation.**

* Indicates an active control function in the Standard Interface application.
** The Power Down mode, Idle mode, GF1 and GF0 are not used in the Standard Interface application. See the 8032 data sheets for a description of these modes and features.

8755 EPROM-I/O Ports description

The Standard Interface Board control firmware is programmed in an Intel 8755A-2 EPROM-I/O Port chip which has a 16 K memory organized into 2048 words by 8 bits. The 8755A-2 has two eight-bit general-purpose ports and each port line can be individually programmed to be either an input or an output. The input/output states of the ports are controlled by Data Direction Registers which define the mode of each bit. The port output latches may be read to verify their state but the state of the Data Direction Registers cannot be read. In the Standard Interface application, the state of the port latches cannot be read because IO/M- is tied low.

The 8755 requires an 11 bit address to read the 2048 word EPROM memory. On-chip address latches store the low byte of address output on the multiplexed Address-Data Bus. The trailing (falling) edge of the 8032 ALE (Address Latch Enable) stores address bits AD0 through AD7 (8032 port bits P0.0 through P0.7) in the 8755 address latches. The upper 3 bits of the 8755 address are driven by the 8032 Port 2 bits (bits 2.0, 2.1 and 2.2) and are stable during the memory operation.

All interactions with the 8755 require activation of either one or both chip enables. In this application, the low-true 8755 chip select CE1bar (AD12) is driven by 8032 Port 2.4. The high true chip select (CE2) is held statically high by a pull-up resistor to +5 Volts. In address space, CE1bar is low for all addresses under 4096; thus the EPROM address space

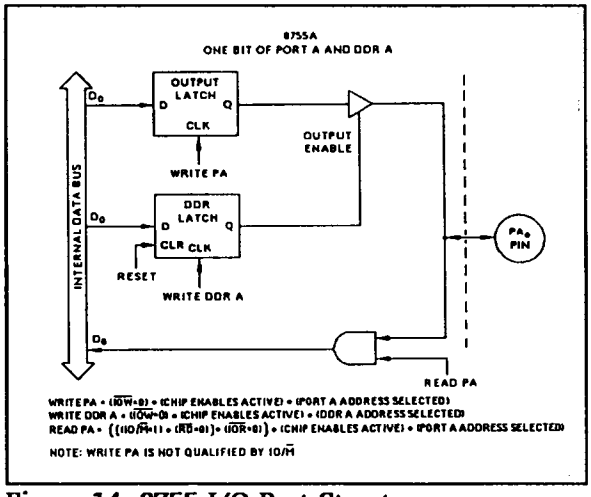


Figure 14, 8755 I/O Port Structure

is 0 to 4095D or 0 to FFFH.

The 8032 performs two types of operations on the 8755: program read cycles from the EPROM memory and I/O read/write cycles with the I/O ports and Data Direction Registers. The 11 address bits select words in the memory but only the bottom address bits are used for I/O operations (the upper 9 bits are don't-cares). The address assignment of the I/O ports and Data Direction registers are:

AD1	AD0	Selection
0	0	Port A
0	1	Port B
1	0	Port A Data Direction Register
1	1	Port B Data Direction Register

When IOWbar goes low and the chip is selected by CE1-, the data on the Address-Data bus is written into the the port or register selected by the state of AD0 and AD1. The port outputs change when IOW- goes high. In this application IOW- is driven by the 8032 WR- signal. The address space for these I/O functions is 0 to 4.

The port state can be read out when the address is latched, Chip enables are active and either RD- goes low with IOM- high or IOR- goes low. The port data is impressed upon the Address-Data bus and sampled by the microprocessor. In this application, IOM is tied low so the port data is read by the 8032 RD- signal which drives the IOR- input.

The I/O mode of the port bits are controlled by the state of the associated Data Direction Register bit; a 1 in a DDR bit causes the associated port bit to be in the output mode and a 0 is the converse.

To read the EPROM memory, the address must be in the appropriate range, CE1- low, and I/OM low. When RDbar goes low the contents of the addressed memory location are impressed upon the Address-Data bus where it is sampled by the microprocessor. In this application, IO/M is tied to logic ground and the 8032 PSEN- (Program Store Enable) output drives the 8755 RD- input.

The 8755 Port A drives the Relative Address (RA) bus and remains static at the value estabalished for the most recent control message or monitor request message execution.

The 8755 Port B outputs four discrete control terms which activate control logic in the device and Multiplexer- A/D Converter. The usage of these control discretes is discussed below in the Device Interface Timing description and in the firmware description of Section 3.

The 8755 I/O port structure is shown in Figure 14 on the previous page. For additional details on the 8755A-2, see the 8755 data sheets in Section 6.

8156 RAM-I/O Ports-Timer Description

The 8156H-2 RAM-I/O Ports-Timer chip provides 256 bytes of RAM, two eight-bit I/O ports (Ports A and B), a third six-bit I/O port (Port C) which can be used for handshake control of Ports A and B, and a programmable 14-bit counter/timer. The counter/timer is not used in the Standard Interface application and its input and output pins are not connected to the board circuits or I/O connectors.

Like the 8755, the 8156 has on-chip address latches to store the 8-bit address (AD0 through AD7) output by the 8032 on the multiplexed Address-Data bus. The trailing edge of ALE strobes these address bits into the address latches.

The 8156 uses a high-true Chip Enable. In the Standard Interface application, CE is bit AD12 from the 8032. AD12 has a weight of 4096 so the RAM address space is immediately above the 8755 address space.

The IO/M- input selects either the RAM memory or the I/O ports and registers during I/O operations. If IO/M- is low, the RAM is selected; if high, the ports and registers are selected. In the Standard Interface application, IO/M- is driven by address bit AD8 which has the address weight of 256. Thus the 8156 I/O address space is 4352D through 4358D (4096 + 256 through 4096 + 256 + 6).

A read or write operation to the 8156 requires a chip enable, an 8-bit address, a high or low IO/M- and either a low-true read strobe (RD-) or a low-true write strobe (WR-). The IO/M- input determines the I/O mode of read/write operations. If IO/M- is a 1 during a read or write operation, the ports, Command Register or Status registers may be written or read (the Command and Status registers are described below). If IO/M- is a 0 during read or write operation, the RAM memory is written or read.

The 8 address bits select either a RAM address or an I/O port, Control Register, Status Register or Timer Registers. The table below shows the addresses of these ports and registers.

Address				Selection
A7 ... A3	A2	A1	A0	
X	0	0	0	Command or Status Register
X	0	0	1	Port A
X	0	1	0	Port B
X	0	1	1	Port C, 6 - Bit I/O or Control
X	1	0	0	Low order 8 bits of Timer Count
X	1	0	1	Upper 6 bits of Timer Count & 2 bits of mode

X = Don't Care

The programmable 8-bit Command Register determines the operating modes of the ports and timer. The Command Register is loaded by an I/O write operation and may not be read. The Command Register addresses and functions are tabulated below.

Depending upon the state of the Command Register, Port C has four operating modes which consist of simple parallel input/output and two modes of handshake control for ports A and B. Both simple parallel and handshake modes are used in the Standard Interface application.

Unlike the 8755, the 8156 Port A and B bits are not individually programmable to input or output; the eight-bit ports are settable to either input or output mode. In the Standard Interface Board application, both modes are used (in concert, both A and B are set to input or output mode as tandem pairs) because Ports A and B are the interface to the CMD/MON bus which inputs or outputs 16 bits of parallel data to the device logic. Ports A and B can operate as simple I/O ports or as handshake ports depending upon the state of the Command Register. (These two modes are described below). The handshake mode of Ports A and B is used to input monitor data from the device; a port strobe clocks the data into the input latch. The strobe is derived from the DEV ACK response from the device logic. The simple input mode is used to input the ID byte and serial number because this portion of the device logic does not provide a DEV ACK signal. Ports A and B are strobe-loaded by the STB/ACK term which results from either a Device Acknowledge or A/D Converter EOC (end-of-conversion) pulse. The function of this term is discussed below in Section 2.4, Standard Interface Board Logic.

The Command Register consists of eight latches. Four bits (0 to 3) define the mode of the ports, two bits (4 and 5) enable or disable the interrupt from port C when it acts as control port, and the last two bits (6 and 7) are for the timer.

The Command Register contents can be altered at any time by using the I/O address XXXXX000 during a WRITE operation with the Chip Enables active and IO/M = 1 (driven by AD8). In the Standard Interface application the 8156 Command Register address is 1100H. See the Program Memory Map in Section 3.2. The state of the Command Register cannot be read out. The Command Register bit assignment is shown below:

8156 Command Register

(MSB)							(LSB)
AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
TM_2	TM_1	IEB	IEA	PC_2	PC_1	PB	PA

PA and PB: a 1 defines either PA_{0-7} or PB_{0-7} as output. A 0 defines PA_{0-7} or PB_{0-7} as inputs.

PC_2 and PC_1: 00 = Alt Mode 1*
01 = Alt Mode 2
10 = Alt Mode 3
11 = Alt Mode 4*

* Both ALT 1 and ALT 4 modes are used in the Standard Interface application, see the description below.

IEB and IEA: a 1 enables Ports A or B interrupts, not used in the Standard Interface application.

TM2 and TM1: Timer mode control, the 8156 Timer is not used in the Standard Interface application.

An 8-bit Status Register has discrete flags that indicate the interrupt and handshake status of ports A and B when Port C is used for handshake control of these ports. The Status Register also indicates the interrupt state of the counter/timer. The Status Register is read by an I/O read operation but cannot be written to. It has the same I/O address as the Command Register. The Status Register format is shown below.

(MSB)							(LSB)
AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
NU	Timer	INTEB	BBUF	INTRB	INTEA	ABUF	INTRA

NU = Not used. INTEX = Port A or B Interrupt enabled. XBUF = Port A or B latches loaded.
INTRX = Port A or B Interrupt Request. Timer = Timer Interrupt, not used.

Figure 15 shows the logic for one bit of Ports A and B. Note that a multiplexer controls the input mode for these ports. In position 1 the state of the output latch may be read as an input to verify the output latch state. In position 2, the port pin is connected directly to the read port gate; this is the simple input mode. In position 3 the output of an input latch is connected to the read port gate. The input latch is strobed by Port A or Port B strobes on PC2 or PC5 when Port C is set to the ALT 4 mode. In ALT 4 configured for control of the Port A and B inputs.

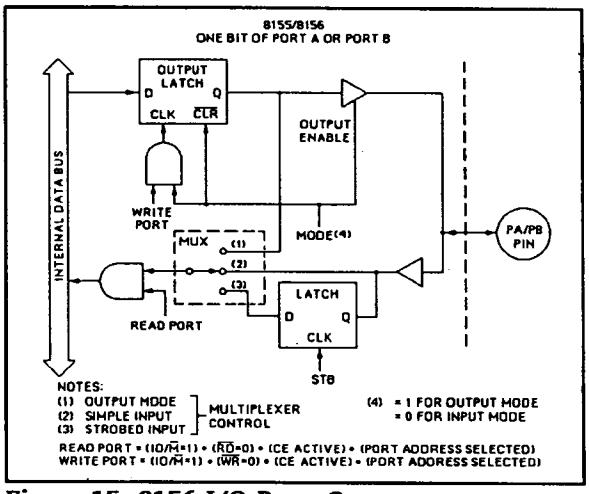


Figure 15, 8156 I/O Ports Structure

The simple input mode (ALT 1) is used for unsynchronized sampling of a port state. In the Standard Interface application, the ID code and Module Serial Number are read as simple inputs on Ports A and B. Figure 16 (next page) shows the

timing for a simple input on Ports A or B.

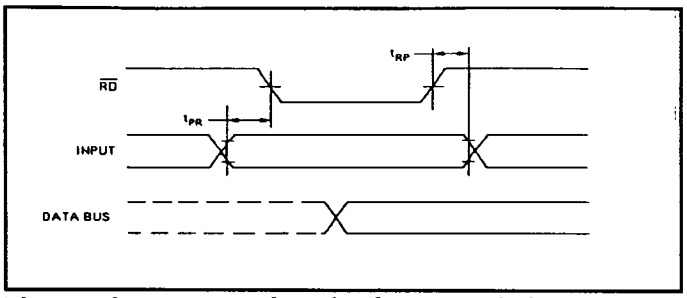


Figure 16, Ports A and B Simple Input Timing

The strobed input mode (ALT 4) is used to read Port A or B input states in response to an external strobe signal. The strobe causes the input latches to store the state of the port pins. In the Standard Interface application, the CON/Mon lines are connected to the device logic; in a monitor data read operation, the DEV ACK signal causes the ports to be strobed to store the state of the CON/MON lines in the port A and B input latches. Figure 17 shows the Timing for the strobed input mode of Ports A and B. The details of the monitor data read logic are described in the Board Logic section.

In the strobed input mode, PC0 and PC3 output interrupt pulses in response to (high-true) activation of the A STB or B STB signals. These PC0 and PC3 pulses can be used as an interrupt to a microprocessor. In the Standard Interface application, the PC0 and PC3 pins are not connected to the 8032 interrupt inputs but the occurance of an interrupt is sensed by the firmware which recurrently tests the states of Status Register bits AD0 (Port A Interrupt Request Flag) and AD3 (Port B Interrupt Request Flag). These interrupt flag bits are set by the trailing edge of the PC0 and PC3 strobe signals. PC0 is set when A STB (PC2) goes high and PC2 is set when B STB (PC5) goes high. In the Standard Interface application, the Device Acknowledge (DEV ACK) signal from the device logic activates the A STB and B STB inputs. The trailing edge of these signals sets the interrupt flag in the Status Register. Figure 18 shows the timing for the strobed output mode of Ports A and B.

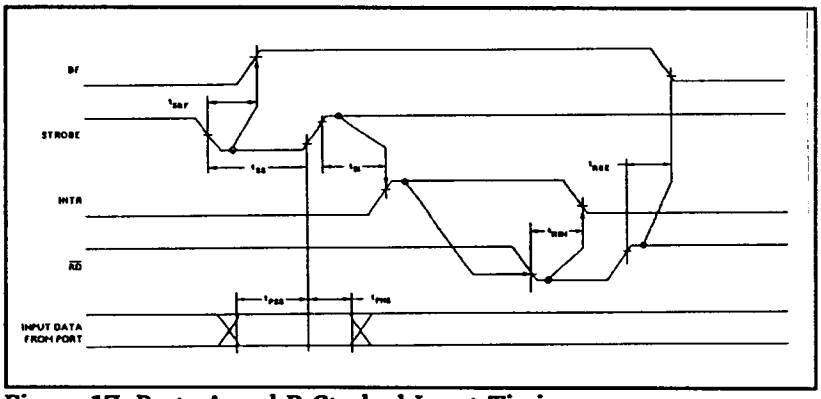


Figure 17, Ports A and B Strobed Input Timing

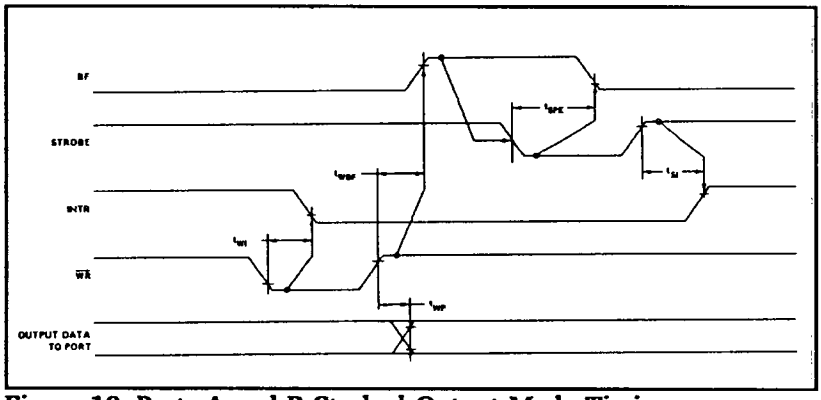


Figure 18, Ports A and B Strobed Output Mode Timing

The strobed output mode is used in the Standard Interface application to output the command argument to the CMD/MON bus. In the strobed output mode, PC1 and PC4 signal external logic that the port output latches have been loaded and that the port states may be read. These signals are not used in the Standard Interface application because DEV REQ signals the device logic that an I/O operation has been started. The Device Acknowledge (DEV ACK) signal from the device logic activates the A STB and B STB inputs. The trailing edge of these signals set the interrupt flags AD0 and AD3 in the Status

Register. Figure 18 (previous page) shows the timing for the 8156 Strobed Output mode of Ports A and B.

When Port C is set to the simple input (ALT 1) or output (ALT 2) mode, Port C logic is similar to the Ports A and B simple input or output logic pictured in Figure 13 above.

The multiplexer control is an implicit function of the Command Register control bits PC_2 and PC_1. When Alt 3 or Alt 4 modes are specified by these bits, the affected ports operate in the strobed mode. When these bits do not specify the strobed modes for Ports A or B, the ports operate as simple inputs or outputs.

The Port C pin functions in the ALT 4 mode are tabulated below.

Pin	ALT 4 Function
PC0	Port A Interrupt, used in the Standard Interface application. *
PC1	Port A Buffer Full, not used in the Standard Interface application.
PC2	Port A Strobe, used to strobe the data on Port A into the port A input latches.
PC3	Port B Interrupt, used in the Standard Interface application. *
PC4	Port B Buffer Full, not used in the Standard Interface application.
PC5	Port B Strobe, used to strobe the data on Port B into the port B input latches.

* The PC0 and PC3 pins are not connected to the 8032 interrupt inputs, see the description above.

The high-true 8156 Reset input initializes the three ports to the input mode. In the Standard Interface application the Reset signal is provided by the board reset chip, U12.

For additional details of the 8156 characteristics see the data sheets in Section 6.

74LS245 Bidirectional Bus Driver

The 74LS245 is an eight-bit, tri-state bidirectional bus driver that drives buses in either direction under control of a direction (DIR) input. The outputs are tri-state and enabled onto the bus by a low-true output enable. 74LS logic data books describe this chip in detail.

2.3 STANDARD INTERFACE BOARD LOGIC

During the following description, the reader should have drawings D55002S001 and D55002S002 at hand for reference.

The 8032 microcontroller and its support chips were described previously with the emphasis on the properties of these chips and their signals. The emphasis in this section is on the interactions of the microcontroller and its support chips with the board-device interface logic.

The MCB bus and protocols were described in Section 1. Except for power-up reset, all interface board activity is a response to the XMT bus messages.

Interface logic chips on the board connect the microcontroller and support chips to the buses and device circuits. Device logic is treated briefly here but is described in more detail in the Device Interface Timing and Typical Device logic sections below. Although the 8032 firmware performs the bulk of the interface's logical operations, the board's analog and digital circuitry are vital elements in the interface function.

Figure 1 (following the signal description) is the block diagram of the Standard Interface and Device logic. Note the interface-device and Monitor and Control bus signals. The board circuitry interfaces these signals to the microcontroller.

Interface Signals

The interface signals and their functions are briefly listed below. This list is an abstraction from specification A55001N002-A. For additional details see this specification in Section 6.

The MCB bus signals are XMT and RCV.

XMT The Controller outputs control and monitor data request messages to the interface boards on the XMT bus.

RCV The RCV bus conveys function codes and monitor data messages to the Controller.

RESET The RESET bus conveys a reset signal to the interface boards. RESET is not used in the VLBA Standard Interface application.

The following signals interface the Standard Interface board and device circuitry.

RA Relative Address is the difference between the 16-bit bus address and the first address of the block. RA is 8 bits which provides 256 addresses.

CON/MON CON/MON is a 16-line parallel tri-state bus used for message argument interchange between the interface board and device logic. During the intervals between argument transfers, the board tri-state drivers are disconnected. Device logic should also be disconnected from the CON/MON bus after an argument transfer.

R/W- R/-W (read/notwrite) designates the type of interaction with the device logic. If low, it requires the device to read the data standing on the CON/MON bus. If high, it requires the device (or the on-board A/D converter) to impress monitor data on the CON/MON bus. R/-W is held low during the interval between message executions.

DEV REQ — DEV REQ (Device Request) signals the device that a command or monitor operation must be performed. The device must decode the RA and execute the control or monitor data action as a function of the address decode. DEV REQ remains high until the device returns a DEV ACK (Device Acknowledge) or an ANENB signal which then causes DEV REQ to go low. In the event the device does not return a DEV ACK (or ANENB) signal within 500 usec, the interface disconnects the CON/MON bus, drops DEV REQ and signals a no-response fault to the controller by a DC2 function code on the RCV bus.

DEV ACK — DEV ACK (Device Acknowledge) signals that the device has (if a command) read the command data on the CON/MON lines or (if the requested data is digital) has impressed the requested data upon the CON/MON lines.

ANENB — ANENB (Analog Enable)is the DEV ACK counterpart when RA specifies an analog signal. ANENB signals the interface that RA specifies an analog signal which is to be sampled and converted by the on-board analog multiplexer and A/D converter. ANENB is generated by the device address decode logic. When ANENB goes true, the analog multiplexer selects the RA-designated channel and the A/D conversion process is initiated. In this case the device logic must hold DEV ACK low. If RA does not specify an analog signal, the device logic must hold ANENB low.

HI/LO SEL — HI/LO SEL signals the interface that either the lower three RA bits (RA:0,1,2) or the next three RA bits (RA:3,4,5) are to be used by the on-board analog multiplexer. HI/LO SEL controls this multiplexer. HI/LO SEL should always be low if only the on-board analog multiplexer is used. If both the on-board and device multiplexers are used in conjunction, HI/LO SEL should be set high if the RA specificies an address greater than 7. The HI/LO SEL term enables analog addresses to be contiguous in address space. See the HI/LO SEL Usage description below. Section 8.2 in specification A55001N002-A also describes the generation and usage of this term.

ANLG-X — ANLG-X are eight sets of differential or single-ended analog signals which are to be selected and converted to digital values for monitor data readout.

ID REQ — ID REQ is 8032 port P1.5. When ID REQ goes low, the device logic should impress the ID code on the CON/MON bus as described below.

At this point, the reader should refer to the Interface Block Diagram (Figure 1) on the next page to put these bus and device interface signals into context. Although it is easy to interface devices to the VLBA Standard Interface board, it is important that the device interface designer has a proper understanding of the character and usage of these signals. Section 2.6 Interface - Device Timing describes the timing relationships of these signals. Figure 23 in Section 2.6 depicts this timing.

There are two versions of the VLBA Standard Interface board. Version "S" uses a single-ended multiplexer-A/D converter; version "D" uses a differential multiplexer, an instrumentation amplifier and A/D converter. The control firmware and connector pin-out are identical for the two versions. Most of the digital circuitry is identical in the two versions; the differences are the result of the way that the A/D converters are controlled and read out. From the board digital interface signal perspective, the two versions are identical. The bulk of the description applies to both versions; differences between versions are described as they are encountered.

The following description assumes that the interface board is receiving some message within the assigned address block.

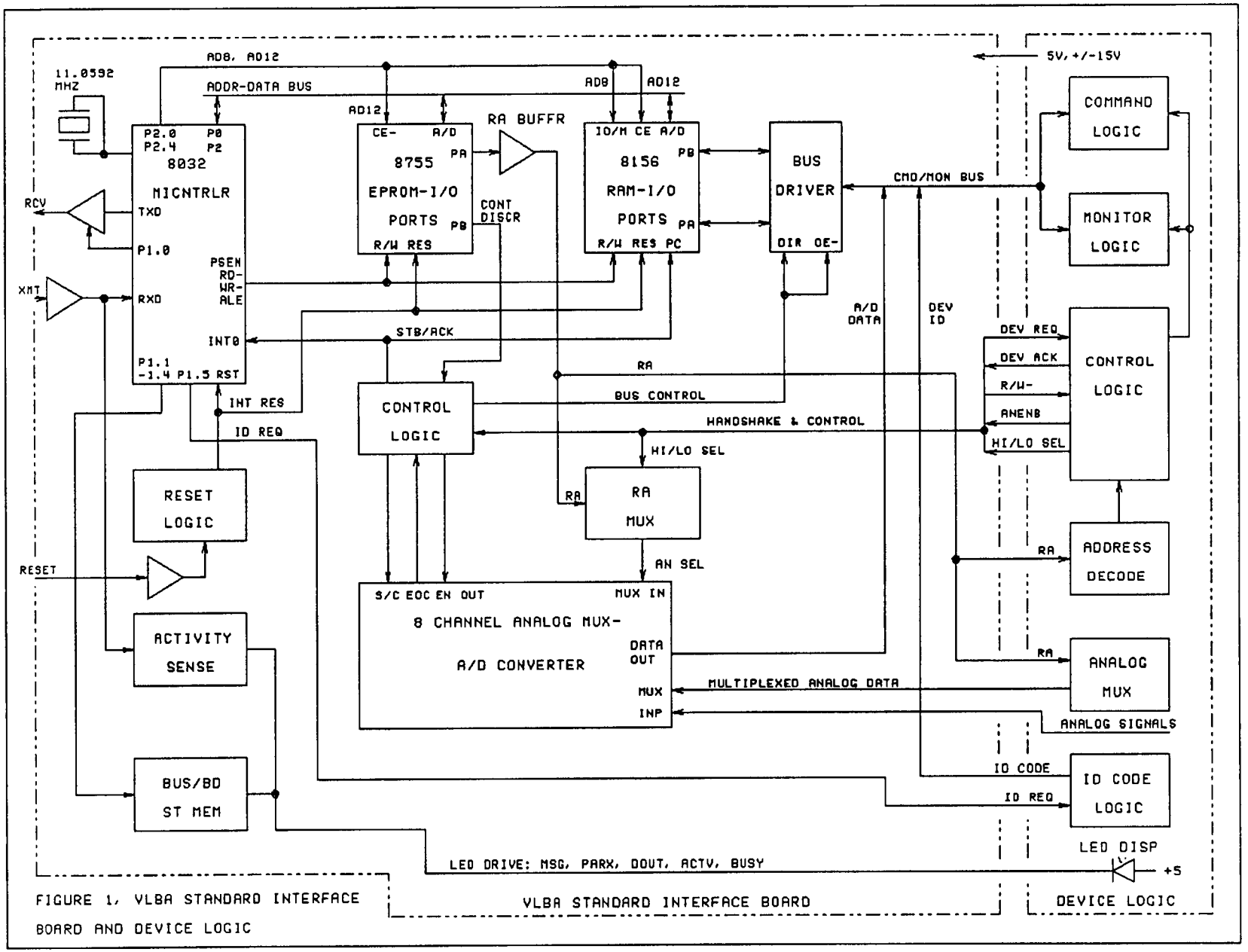


FIGURE 1, VLBA STANDARD INTERFACE BOARD AND DEVICE LOGIC

The board circuitry performs five types of operations, four of which are controlled by the microcontroller. After the hardware power-up reset and firmware initialization of hardware, the board operations are a response to the XMT bus signals described in Section 2.1.

The four firmware-controlled operations are: 1) Device ID code read, 2) Execution of a control message command to the device, 3) Acquisition of digital monitor data from the device in response to a monitor data request message, 4) Acquisition of analog monitor data from the device in response to a monitor data request message.

Bus Drivers and Receivers

The XMT bus receiver is a 75175 and the RCV bus driver is a 75174. These chips interface the board TTL levels to the bus RS-485 levels. The 75175 XMT bus receiver drives the 8032 serial port RXD input. 8032 port P1.0 drives the enable input of the the RCV bus driver chip (75174) which outputs the 8032 TXD signal to the RCV bus. A high input to the driver enable pin connects (i.e. turns on) the driver tri-state output circuits. In responding to a control or monitor data request message, the 8032 connects the driver to the RCV bus to output control function codes or monitor data values at the approptiate times. Prior to these responses the driver is disconnected. After the function codes and monitor data have been output, the driver is disconnected by the 8032 firmware to permit another interface to use the RCV bus.

Power On Reset

Power-on reset is accomplished by R1, CR1, C10 and U7A in version D, and CR1, R8, C12 and U1B in version S. The RC circuit is a delay circuit; the one-shot is triggered when the charge on C10 reaches the switching threshold of U7. U7 (or U1) is a Schmidt-input, 74LS221 one-shot which triggers reliably on input signals having slopes as low as 1 volt/second. A dV/dt this small permits the RC delay circuit to have a large time constant so that a long delay is realized after logic power is applied. The positive-going trigger threshold of the B input is typically 1 volt and a maximum of 2 volts. The RC board time-constant is about 0.1 seconds so that with a 1-volt threshold, the delay time from power-on to trigger time is about 32 milliseconds (assuming the 5 volt power is a step function). The one-shot delay time is about 75 milliseconds. The high-true U7-13 reset pulse resets the 8032, 8755 and 8156. Note that an RS-485 reset bus option can be used by repositioning the reset jumper plug. This bus reset feature is not used in the VLBA Standard Interface application.

Device ID Byte Read

The interface board must be able to relate the addresses of XMT bus messages to its assigned address block. Messages with addresses inside the block must be executed, messages outside the block are to be ignored. To identify messages addressed to the board, incoming message addresses are compared with the block starting address (in address 2N+1) and block length (in address 2N) values.

The board does not have internal address assignment logic; this function is performed by the device logic and Controller. When the board emerges from the power-on reset, 8032 initialization firmware sets the address block to 7FF0H through 7FFFH. After this initial assignment, the board reads the unique ID byte from the device ID logic on the lower byte of the CON/MON bus. The ID byte value (call it N), is used to establish the board's addresses. At a later time the Controller will send a control message which will set the block starting address in 2N+1. The Controller will send a second control message which will set the block length in address 2N. Monitor data requests to these addresses will return the assigned values.

The firmware operations involved in acquiring the block starting address and block size are described in Section 3.3.

During the ID byte read process, the A/D converter digital outputs are disconnected and the device logic must not impress monitor data on the bus. R/W- from 8755 PB6 sets the 74LS245 bus driver DIR input low (enabling signal flow from B pins to A pins) for input to the 8156. The buffer enable (BFR ENBL) from 8755 PB4 enables the 245 outputs to drive the 8156 Port A and B inputs. To read the ID byte, the 8032 firmware sets 8032 port P1.5 (ID REQ) low; this causes the device logic to impress the seven-bit ID byte on CON/MON lines 0 through 6. The device logic must also impress a parity bit on CON/MON-7 so that CON/MON 0 .. 7 has odd parity. The device logic may leave the upper byte (i.e. CON/MON 8 .. 15) floating (in which case the upper byte value is indeterminate) or use a tri-state buffer to read in zeroes. The device logic may also use the ID REQ signal to impress a user-defined value on the upper eight bits of the CON/MON bus. An earlier (discontinued) use of the upper byte was for device serial number readout. The current convention for device serial number readout is to use a digital monitor channel for this purpose.

The ID code and user-defined value (if implemented) is input from the 8156 A and B ports in the simple input mode described in Section 2.2.

Periodically thereafter, a 5-second firmare timer causes the 8032 to re-read the ID byte to insure that the stored value is correct.

Execution of a Control Message

The following description assumes that there are no parity errors.

When a control message is received with an address within the address block, the firmware writes the eight bits of Relative Address (RA) into Port A of the 8755. RA is immediately available to the device address decode logic and precedes the interface handshaking terms. RA remains static until changed by another message execution.

The message ADH byte MSB is a 1; this identifies a control message.

8156 Ports A and B are set to the output mode and Port C is set to ALT 4 mode by the firmware. The ALT 4 (i.e. strobe-loaded mode) is described in Section 2.2.

8755 I/O Port B bits PB4 ... PB7 are used to output control discretes BFR ENBL, CMD REL and R/W-; these terms condition the interface and device logic. These discretes are loaded into the 8755 Port B latches by the firmware and are sequenced to provide generous timing margins for the device logic.

The R/W- control discrete will be low because the message is a control command. R/W- and BUFR ENBL are loaded into EPROM ports PB6 and PB4 at the same time. CMD RLSE is the stimulus for control message handshaking and is loaded into PB5 30.25 microseconds after R/W- and BFR ENBL.

The signal IOR is also present on EPROM PB7 but has no effect as it is disqualified by R/W- in gate U17-12. At this point in the program, IOR is an artifact resulting from code development.

R/W- is low which sets the 74LS245 DIR input high. This steers the CON/MON bus drivers signal flow to be from the 8156 ports to the device. R/W- inhibits gate U17-12 (version D). The device uses the low state of R/W- to enable the device logic to accept a command argument.

BUFR ENBL controls the connection of the 74LS245 tri-state drivers to the CON/MON bus. The firmware connects the drivers only when signals are to be passed over the bus.

CMD RLSE is peculiar to the command function and becomes true after R/W- and BFR ENBL. CMD RLSE activates DEV REQ via gate U17-13.

RA is set to the appropriate command address 23.8 microseconds before DEV REQ goes true. When DEV REQ becomes true, the device logic initiates the device command execution sequence designated by the decode of RA. Since the message is a control command, the address decode will typically cause the device logic to store the argument standing on the CON/MON bus. The device might have more than one set of command storage latches; in this case RA will select the appropriate latches. Another possible implementation is to use DEV REQ as an address-gated strobe for some device function (in this case the argument is irrelevent).

When the device logic completes the command execution sequence, it should return a DEV ACK to the interface board to signal the completion of the sequence. The leading edge of DEV ACK triggers one-shot U12; U12-7 is a 2 usec low-true pulse which drives the 8032 INT0 and 8156 PC2 and PC5 strobe inputs. (The 8032 firmware does not use the INT0 input; the interrupt is disabled). The 8156 PC2 and PC5 inputs activate the Port A and Port B interrupt bits in the Status Register which are polled by the 8032. When the 8032 sees these interrupt status flags, it lowers DEV REQ which completes the command execution. When the device logic sees DEV REQ fall, it should lower DEV ACK.

In the event of a malfunction in the device logic that inhibits the DEV ACK response, the firmware has a 500 usec timer which causes a DC2 function code to be output to the Controller via the RCV bus. In this case, the interface will lower the DEV REQ signal.

After completion of the command, the R/W- line stays low and RA remains static until the next message execution.

The command execution firmware is described in Section 3.3.

Execution of a Monitor Data Request for Digital Monitor Data

When a monitor request message is received with the address within the address block, the firmware writes the eight bits of Relative Address (RA) into Port A of the 8755. RA is immediately available to the device address decode logic and precedes the other interface handshaking terms. RA remains static until changed by another message execution.

The message ADH MSB will be 0; this identifies a monitor data request. Consequently the R/W- control discrete (8755 PB6) will be set high. The high state of R/W- will condition the device logic for data input to the board.

8156 Ports A and B are set to the input mode by the firmware. Port C is set to the ALT 4 mode (strobed mode, described in Section 2.2). DEV ACK generates the strobe pulse (as described below).

The monitor data request could be for either digital data from the device or for the digital value of a device analog signal which must be multiplexed and converted to a digital value. The device logic must decode RA to distinguish between the two cases. In the present case, a digital value is to be read from the device.

8755 Port B is loaded with three control discretes: R/W-, IOR and BFR ENBL. R/W- and BFR ENBL are set first to permit the device logic to set up. R/W- is high so inverter U18-12 makes the 74LS245 CON/MON buffer DIR input low; this steers data from the device to the 8156 Ports A and B.

When BFR ENBL goes high, inverter U18-2 enables the 74LS245 driver outputs to pass the device data on the CON/MON bus to 8156 Ports A and B.

IOR is set in PB7 18.4 microseconds after R/W- and BFR ENBL are set in PB4 and PB6.

When IOR is set in PB7, gate U17-12 goes true (low) causing gate U15-11 to output a DEV REQ to the device. DEV REQ initiates the monitor data gathering sequence. If the RA designates digital data, the device drivers impress the requested data on the CON/MON bus and raise the DEV ACK line to the interface board which triggers one-shot U12-7 via gate U15-2. The one-shot output, STRB/ACK-, drives INT0 on the 8032 and the 8156 PC2 and PC5 strobe inputs on 8156 Port C. The 8032 INT0 input is not used but the 8156 PC2 and PC5 inputs cause the device data on the CON/MON lines to be stored in 8156 Ports A and B. The firmware recurrently polls the 8156 status register and tests for the presence of the Port A and B interrupt flags which resulted from the activation of the strobe inputs. When the firmware detects these two status flags, it inputs the Ports A and B data via the Address-Data bus and formats the data for output on the RCV bus.

In the event that DEV ACK does not go true within 500 usec, a firmware timer causes the DC2 function code to be output to the Controller on the RCV bus; DEV REQ is also lowered.

After the monitor data has been output to the Controller, the R/W- line will revert to the low state. RA will stay at the current message value until changed by the next message within the address block.

Because the requested data is digital, the device must hold the ANENB line low. The HI/LO SEL line is a "don't care" case; the state of this signal does not affect the digital data gathering sequence.

The monitor data acquisition firmware is described in Section 3.3.

Execution of a Monitor Data Request for Analog Data

When a monitor request message for analog data is received with an address within the address block, the firmware writes the eight bits of Relative Address (RA) into Port A of the 8755. RA is immediately available to the device address decode logic. RA remains static until changed by another message execution.

The first steps in the execution of a request for analog data are similar to that for gathering digital monitor data. The device must decode RA to determine whether the monitor data request is for digital or analog data. If analog data is specified by RA, the logic operations are quite different than for the digital data case. Instead of reading device registers on the CON/MON bus as in the digital input case, analog multiplexers on the board and in the device select the analog signal designated by RA. The selected analog signal is sampled by the A/D converter sample-and-hold circuit and converted to a digital value. The A/D converter EOC (end-of-conversion pulse) impresses the converter data on the CON/MON bus and strobes the 8156 Port C strobe inputs. These logic operations descriptons follow.

Although there are significant differences between the circuitry for digital and analog monitor data operations, both use the same 8032 firmware; the board logic makes both operations seem identical.

The message ADH MSB is 0; this identifies a monitor data request. Consequently the R/W- control discrete (PB6) is set high. The high state of R/W- conditions the device logic for a monitor data operation.

8156 Ports A and B are set to the input mode by the firmware. Port C is set to the ALT 4 mode (strobed input mode, described in Section 2.2). The A/D EOC generates the Port C strobe pulse as described later on.

8755 Port B is loaded with three control discretes: R/W-, IOR and BFR ENBL. R/W- and BFR ENBL are set first to permit the device logic to set up. R/W- is high so inverter U18-12 makes the 74LS245 CON/MON buffer DIR input low; this steers data from the A/D converter to the 8156 Ports A and B.

When BFR ENBL goes high, inverter U18-2 enables the 74LS245 driver outputs to pass the device data on the CON/MON bus to the 8156 Ports A and B.

18.4 microseconds after R/W- and BFR ENBL are set in PB4 and PB6, IOR is set in PB7. When IOR becomes true, gate U17-12 goes true (low) causing gate U15-11 to output a DEV REQ to the device. DEV REQ initiates the analog monitor data gathering sequence.

The device logic RA decode identifies an analog data gathering operation. DEV REQ causes the device logic to feed back two logic terms to the board logic: ANENB and HI/LO SEL. ANENB is analagous to DEV ACK in that it is the device's handshake response to DEV REQ but it really signals the **start** of the analog multiplexing A/D conversion process, not the **completion** of the process as in the digital data case. HI/LO SEL is used to control the on-board digital multiplexer which selects three RA bits to control channel selection in the on-board analog multiplexer-A/D converter. The use of HI/LO SEL is described below.

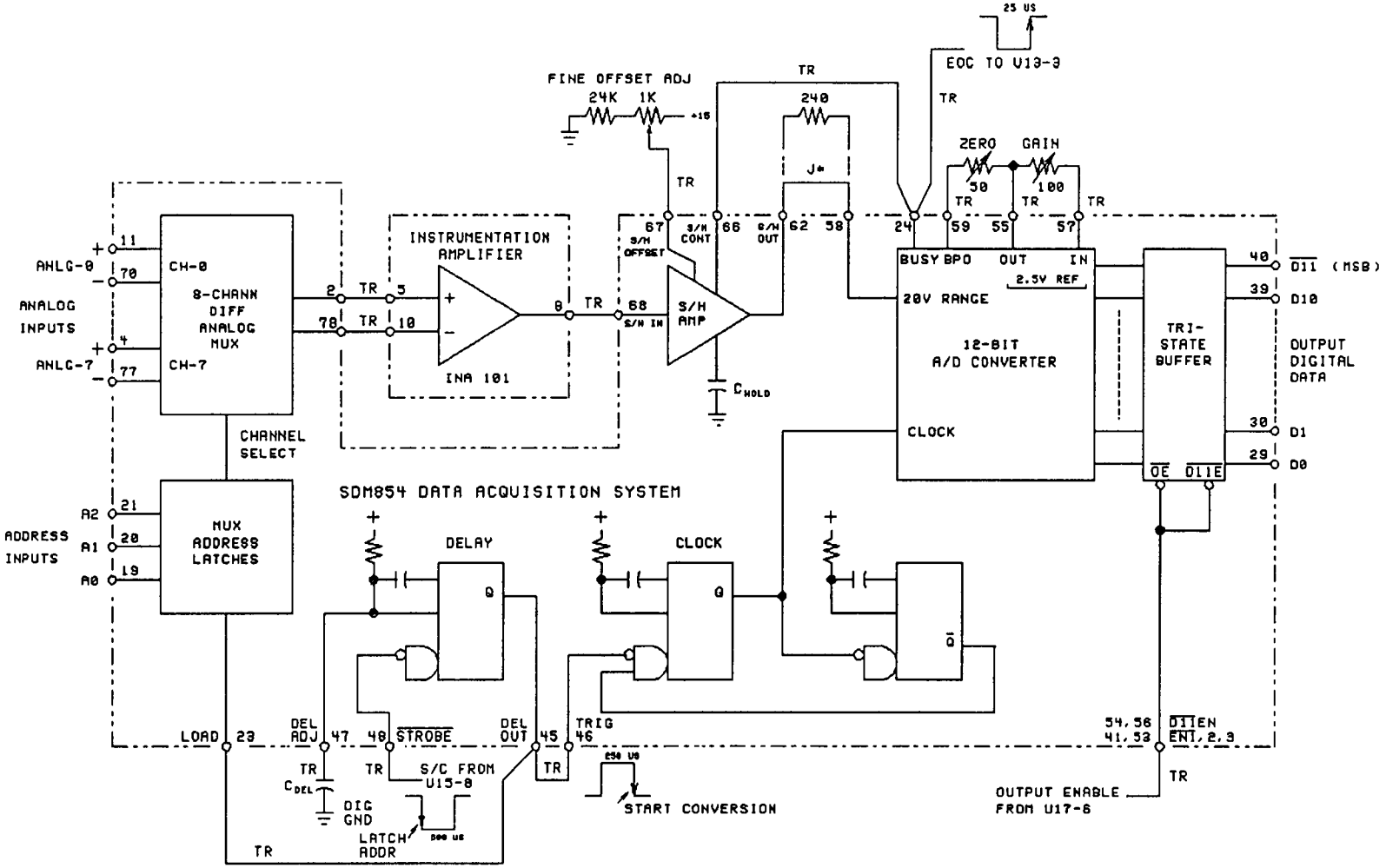
ANENB is typically fed back to the interface board with the delay of the device decode, usually much less than a microsecond. DEV ACK must remain low during this time; a DEV ACK glitch or a sustained high DEV ACK will cause false data to be loaded into the 8156 ports.

There are two versions of the interface board which differ only in the type of multiplexer-A/D converter and A/D interface logic. Version "D" (for differential analog signals) is used in applications in which common-mode noise is imposed upon the analog signals. Version "S" (for single-ended analog signals) is used for applications in which common-mode noise is not a problem. The logic operation of both versions is identical to this point. The logic reference designators are different but this is a minor aspect. The operation of the "D" version is described first.

"D" Version Description

Figure 19, (next page) shows the SDM 854 sequencing logic and the connections to the interface board logic. The SDM 854 module pin numbers are cited below. Section 6 has SDM 854 data sheets.

The 8032 firmware sets IOR in 8755 PB7 which starts the analog multiplexing - A/D conversion sequence. IOR drives gate U15-8 and the other input is ANENB. The output is a 500 usec low true pulse. The pulse leading edge (low-going) triggers the STRB- input (48) of the (Burr-Brown SDM-854) multiplexer - A/D converter. This edge input clocks (via the connection from 45 to 23) the three address bits into the analog multiplexer latches and initiates the Sample/Hold - A/D conversion sequence. The falling edge of U15-8 starts the SDM 854 sequencer shown on Figure 19 (next page). The sequencer consists of three 74123 one-shot multivibrators. The delay of the first one-shot is about 250 usec to permit the analog data to settle in the multiplexer and the Burr-Brown INA 101 instrumentation amplifier. The delay is determined by the 0.01 uf capacitor on the delay adjust pin (47).



NOTES:

1 TR DENOTES A PCB CONNECTION

2 TO SCALE A/D TO 5 MV/COUNT REPLACE J* JUMPER WITH A 240 OHM PRECISION RESISTOR

FIGURE 19, VERSION D MULTIPLEXER - A/D CONVERTER CIRCUITRY

The DELAY one-shot output (45) is connected to the TRIG- (46) pin of the two CLOCK multivibrators. The trailing edge of the DELAY pulse causes the Sample/Hold circuit to hold the analog value on C_{HOLD} and starts the conversion comparison sequence.

Note the jumper which connects SDM 854 signals and control terms to the interface logic. The converter BUSY (42) output is connected to the S/H- CONTROL (66). The ENABLE 1,2,3 and D11 are connected to gate U17-6. The differential output of the multiplexer is connected to the instrumentation amplifier input via pins 2 and 78. The Instrumentation amplifier output (INA 101) is connected to the S/H- IN pin (68). S/H- OUT is connected to the 20V RANGE input (58). The internal +2.5V REF OUT is connected to RV1 gain and RV2 zero potentiometers. The other end of RV1 is connected to the +2.5 IN pin (57) and the other end of RV2 is connected to BPO (59, bipolar offset). These two potentiometers are adjusted during the alignment of the A/D converter.

Note the jumper wire across the 20V Range (58) and S/H OUT pin (62). If this wire jumper is replaced with a 240 ohm precision resistor, the A/D scaling is 5.000 mv/count. The VLBA interface application uses a jumper wire across these two pins; the scaling is 4.8828 mv/count.

The instrumentation amplifier converts the analog multiplexer differential output to a single-ended signal and has unity gain. The differential-input amplifier has a common-mode rejection >100 db at 60 Hz. This high common-mode noise rejection enables accurate conversion of analog signals that are contaminated by common-mode noise. The INA 101 is a high-quality instrumentation amplifier; the reader is urged to review the INA 101 data sheets in Section 6.

The trailing (positive-going) edge of the 25 usec SDM 854 BUSY pulse sets D flip-flop U13-5 high. Gate U17-6 is enabled by U13-5, R/W- and the output of gate U15-8. Gate U15-8 ands IOR and ANENB. When flip-flop U13-5 is set, gate U17-6 output enables the tri-state data outputs of the converter onto the CON/MON bus. The converter D11 enable converts the A/D output to 2's complement format. The trailing edge of gate U17-6 also triggers one-shot U12-7 via gate U15-3. The output of U12-7 is a low true 1 usec pulse which strobes 8156 Port C bits PC2 and PC5. These two port pins load the A/D data into ports A and B and set PA and PB interrupt flags in the 8156 status register. The 8032 polls these flags and when they become true the 8032 reads the A/D data standing in Ports A and B.

On drawing D55001S002, note that the 12 A/D converter data bits are connected to the upper 12 CON/MON bus lines; the lower 4 four bus lines float which makes them indeterminate.

"S" Version Description

Figure 20 (next page) is a block diagram of the HS 9410 A/D converter used in the "S" version. Section 6 has data sheets for this chip. The logic operations of this version are identical to the "D" version up to the point of starting the A/D conversion. ANENB and IOR are anded in gate U7-3 which triggers a concatenated string of one-shot multivibrators that initiate the A/D conversion.

U5A-7 generates a 250 usec low-true pulse; the trailing edge sets D flip-flop U8-5. The rising edge of U8-5 triggers one-shot U4A-7 which generates a low-true 2 usec pulse. The trailing edge of U4A-7 in turn triggers one-shot U4B-9 which generates another 2 usec low-true pulse that is anded with the output of U8A-5 in and gate U7-6. The output of U7A is a high-true signal with a 2 usec wide notch that is 2 usec from the leading edge of U7-6. This signal drives the R/C- input of the A/D converter. The negative-going (i.e. 1 to 0) edge of the 2 usec notch starts the conversion sequence. Sheet 2 of D55002S001 shows this signal adjacent to the STS (i.e. Status) input.

The STS output of the converter goes high when R/-C goes low. When the conversion is completed, STS drops; this signals that the A/D data may be read. The negative-going STS signal triggers

one-shot U5B-9 which generates a 1 usec low-true pulse. The trailing edge of this 1 usec pulse clocks the top 8 bits of A/D data into U20, an octal latch that drives CON/MON-15 ... CON/MON-8. To read the lower 4 A/D converter bits, the A0 input of the A/D must be clocked by a positive-going signal. U8B-9 is set high by the trailing edge of the 1 usec pulse from U9B-9. When U8B-9 rises it clocks A0 which sets the lower 4 data bits on the top A/D output bits that are connected to CON/MON-7 ... CON/MON-4. Flip-flop U8B-8 enables the the 374 outputs onto the CON.MON bus. At this point all 12 A/D data bits are enabled onto the CON/MON bus. The negative-going edge of U8B-8 triggers one-shot U9A-7 through gate U7-8. The trailing (positive-going) edge of the 1 usec pulse from U9A-7 strobes the 8156 PC2 and PC5 inputs. These strobes load the 8156 Ports A and B. Meanwhile, the 8032 has been polling the 8156 status register. When it sees the two interrupt flags, it reads the two 8156 ports and formats the data for output on the RCV bus.

Inverter U10-2 inverts the data MSB which converts it to the 2's complement format.

This A/D converter does not latch the multiplexer address bits; they must be static during the conversion.

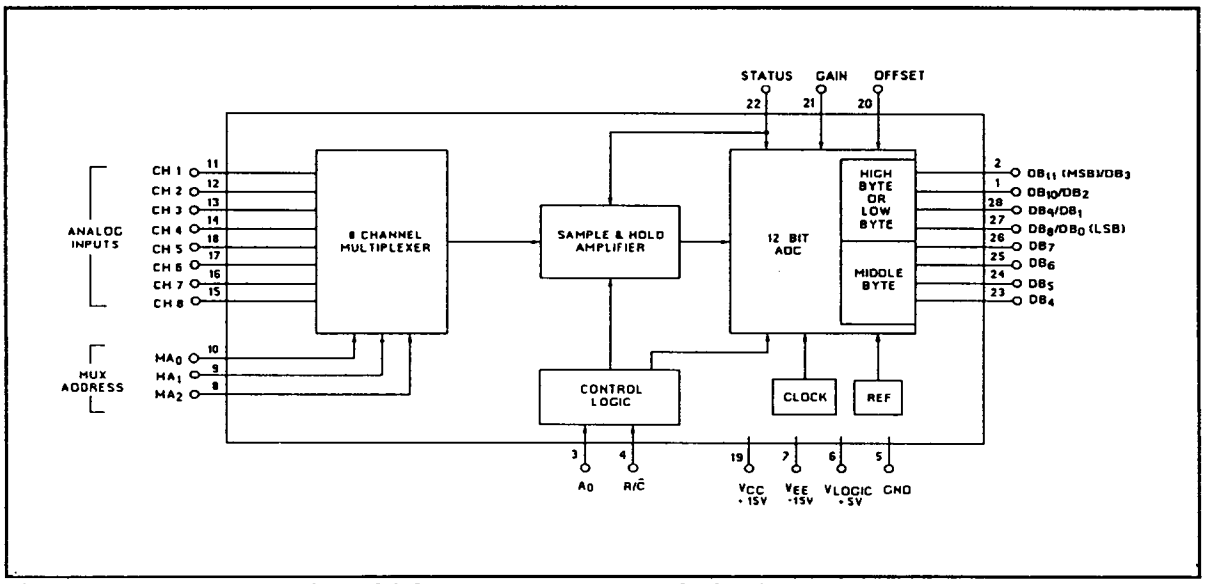


Figure 20, HS 9412 12-Bit Multiplexer-A/D Converter Block Diagram

HI/LO SEL Usage

The use of HI/LO SEL logic makes the analog signal address space compact when device multiplexers are used in conjunction with the on-board analog multiplexer.

If no more than eight analog signals are to be multiplexed and converted to digital values, HI/LO SEL should always be low. To select an input, the on-board multiplexer is controlled by the three lowest bits of RA: RA0, RA1 and RA2.

If the device circuitry has analog multiplexers, their outputs are connected to the inputs of the board multiplexers, ANLG-0 .... ANLG-7. The following is an example of the way these device multiplexers should be controlled, the selection of the on-board multiplexer inputs that are still available, and the resultant analog data address space. Assume that the device has a single 8-channel multiplexer. Channel selection for this device multiplexer is controlled by the three lowest bits of RA: RA0, RA1 and RA2. As

RA0, RA1 and RA2 are sequenced through the eight states, this device multiplexer will select analog signals. Assume that this device multiplexer output is connected to the ANLG-1 input of the board multiplexer. To use the other 7 available on-board multiplexer inputs, channel selection should be controlled by RA0, RA1 and RA2. How can the on-board multiplexer select on ANLG-1 the output of the device multiplexer for all states of RA0, RA1 and RA2 ? Clearly, for this case, the on-board multiplexer must continue to select the ANLG-1 input for all states of RA0, RA1 and RA2. The solution is to use the next three higher bits (RA3, RA4 and RA5) for selection of the ANLG-1 input and use RA0, RA1 and RA2 to control the selection for the other seven available inputs. An on-board digital multiplexer selects either RA0... RA2 or RA3.. RA5 to control the on-board multiplexer. The HI/LO SEL signal controls this address bit selection when one or more device multiplexers are used in conjunction with the available on-board multiplexer inputs.

Figure 21 (at the end of this section) shows suggested logic for the generation of HI/LO SEL when using device multiplexers.

The address bit multiplexer is a quad two-to-one (74LS157) multiplexer with RA0, RA1 and RA2 connected to the A inputs. The next three address bits, RA3, RA4 and RA5, are connected to the B inputs. A low on the SEL line will select the A inputs and a high will select the B inputs. The three outputs of the 74LS157 are connected to the address inputs of the SDM 854. In the case we are considering, the seven available board multiplexer inputs are ANLG-0, ANLG-2 .... ANLG-7 which are selected by RA0 .. RA2. By using some decode logic on RA, we can easily generate a HI/LO SEL term to select ANLG-1. For this case, HI/LO SEL should be high for octal addresses 10, 11, ... 16 and 17 and low for addreses 0, 1 ... 6, 7. With RA3, RA4 and RA5 connected to the A, B and C inputs of a 74LS138 decoder, the Y1 output and an inverter will implement the HI/LO SEL term. With this HI/LO SEL logic, octal addresses 10 through 17 select (via ANLG-1) analog signals connected to the device multiplexer; octal addresses 0, 2, 3, 4, 5, 6 and 7 select analog signals connected to the seven available channels of the on-board multiplexer (i.e. ANLG-0, ANLG-2, ... ANLG-7). Thus, fifteen analog signals may be selected over an octal address space of 0 through 17. The choice of ANLG-1 for the output of the device multiplexer is that it is convenient to associate the 001 state of RA5, RA4 and RA3 with ANLG-1.

If a second 8-channel device multiplexer is added, its output is connected to ANLG-2 and HI/LO SEL is made high for octal addresses 10 through 27. RA0 .. RA2 control the input selection of this second device multiplexer. The inputs for this second multiplexer are selected by octal addresses 20 ... 27. In this case the analog multiplexing capacity is 23 channels with an address space of 0 through 27 octal. With these conventions, up to eight additional device multiplexers can be connected to the on-board multiplexer for a total capacity of 64 channels. The logic for the expansion of analog multiplexing capacity 15 channels is similar to that shown on Figure 21.

Bus and Board Status Display Logic

The interface board logic has circuitry to drive front panel display LED's. The display provides a visual indication of bus and interface board status.

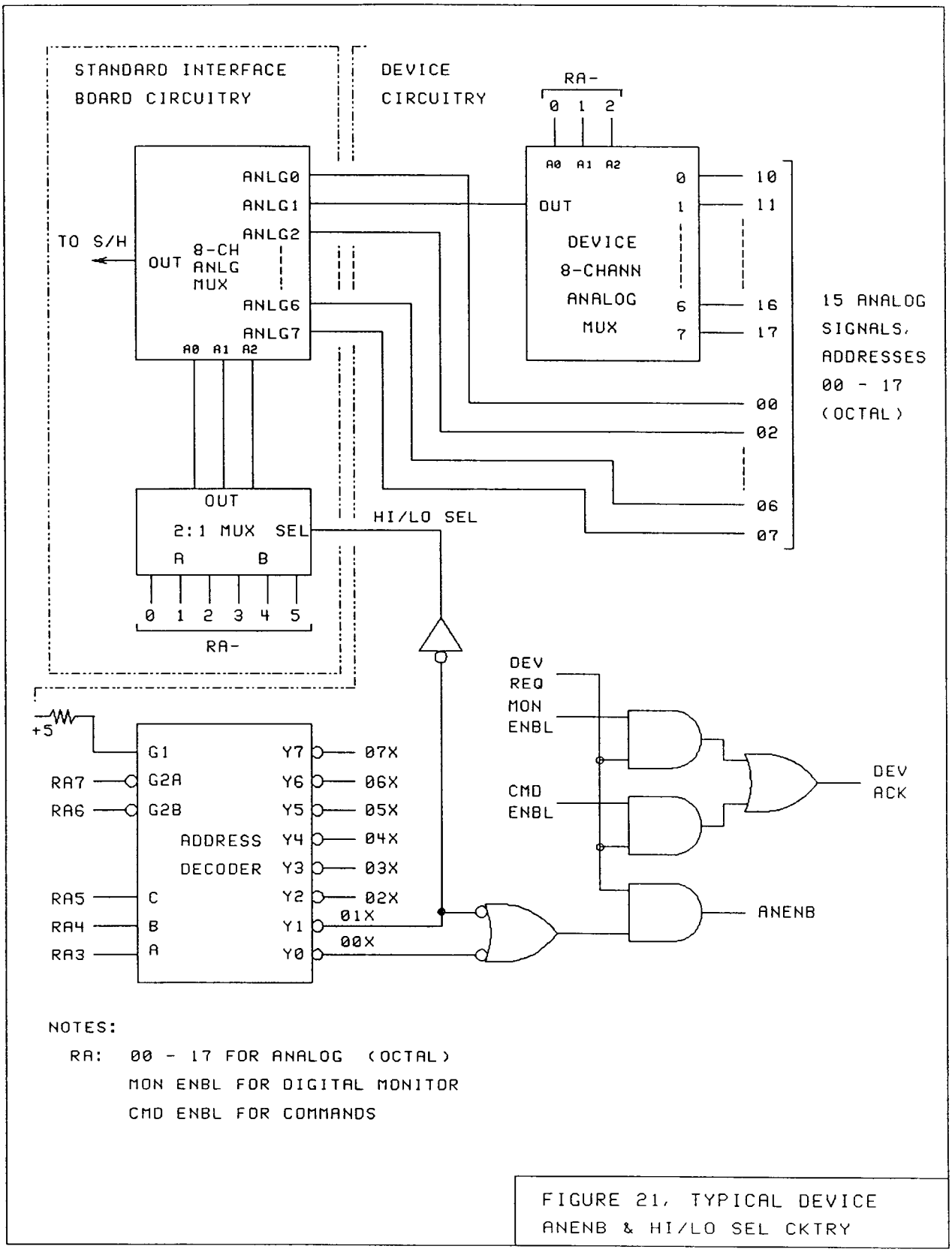
The simplest display is the XMT bus activity status which indicates that the XMT bus is changing state, an indication that there is probably message flow from the Controller. One-shot U12B-9 (96LS02) is triggered by the output of the XMT bus receiver. The one-shot stretches the XMT bus pulses by 27 usec. An edge-triggered, pulsing display of XMT bus activity is used rather than a direct connection. A direct connection might cause the LED drive to be on for a long time between byte transmissions.

The other four LED's are driven by 8032 firmware via 8032 ports P1.1 through P1.4. Ports P1.2, P1.3 and P1.4 drive the D inputs of U3 which is a 74LS173 quad latch. The latch Q outputs sink current through the LED's; to light an LED, the associated Q output is set low and the output enable is pulsed low

by one-shot U7B (described below). A display latch bit is cleared low by a CLR port bit instruction. For example the CLR BUSY instruction will enable the BUSY memory latch bit to be cleared (by P1.2) when the chip is clocked (by P1.1) to load the latches. The SET BUSY instruction will clear the BUSY latch bit when the chip is clocked. The latch chip is edge-triggered when P1.1 makes a low-to-high transition. The four latched display lines are periodically updated by the 8032 firmware.

One-shot U7B (74LS221) is clocked on the falling (trailing) edge of the latch clock. The one-shot period is about 2 millisec. The one-shot Q- output drives the U3 (74LS173) output enables so that the LED's are illuminated only for the duration of the one-shot pulse.

Current limiting resistors should be provided in the display circuitry when the LED display outputs are used.



NOTES:

RA: 00 - 17 FOR ANALOG (OCTAL)
MON ENBL FOR DIGITAL MONITOR
CMD ENBL FOR COMMANDS

FIGURE 21, TYPICAL DEVICE ANENB & HI/LO SEL CKTRY

2.4 TYPICAL DEVICE LOGIC

This section describes the operation of a sample Device Interface shown on drawing C55002S003 (included in Section 5). This generic logic may be copied as shown or adapted to suit the application.

The logic shown will accomodate eight analog signals, eight 16-bit command registers (74LS374's), and eight 16-bit digital monitor registers (74LS373's). Only one command register and one digital monitor register are shown on the drawing.

The logic in this sample design is qualified by the firmware comparison of the message address with the block starting address and block size. RA, R/W- and DEV REQ are only activated if the interface board firmware has found the message address to be within the assigned block.

The RA high order bits are decoded by a 74LS138 decoder on the left side of the drawing. The decoder Y0 and Y1 outputs are $00X_8$ and $01X_8$, respectively, where X is any digit between 0 and 7. The $00X_8$ decoder output enables the ANENB signal and the command address decoder. The $01X_8$ decoder output enables the digital monitor address decoder. If the R/W- signal from the Standard Interface is high, the digital monitor address decoder (upper 74LS138 with the MON CNTL-X outputs) is enabled. If it is low, the command address decoder (lower 74LS138 with the CMD CNTL-X outputs) is enabled. Both decoders are qualified by DEV REQ from the interface.

If all decoded addresses were used, the block would have 24 addresses: 8 analog monitor, 8 digital monitor and 8 command. RA would range from 000 through 017_8. In practice, the address range would probably be much smaller and would be determined by the highest address used.

In addition to the hardware addresses mentioned above, the interface address block has 16 additional addresses for command-monitor functions which are for functions internal to the interface board. These are described in Sections 2.1 and 2.5.

The configuration shown has a full decode of the eight RA address lines and assigns addresses (octal format) 000 through 007_8 to analog signals. The MON CNTL decoder (74LS138) assigns addresses from 010_8 through 017_8 to the digital monitor data register (74LS373's). One of the eight MON CNTL-0 through MON CNTL-7 address decoder signals would be connected to the 74LS373 LE- (low-true load enable) inputs. The CMD CNTL decoder (74LS138) assigns addresses from 000 through 007_8 to the digital command register (74LS374's). One of the eight CMD CNTL-0 through CMD CNTL-7 address decoder signals would be connected to the 74LS374 CLK (clock) inputs.

The best choice for the digital assignments would be to assign address 010_8 to the digital monitor data and address 000 to the command register. In this case RA would range from 000 to 010_8. It is always best to make the device address space as compact as possible.

The RA lines are static at the address used in the most recent message execution. When a new message is received with an address within the interface's address block, RA is set. RA precedes all other control terms. R/W- is low (write state) between message executions and is set to the appropriate state after RA has been set. DEV REQ is the last discrete to be set on the interface board outputs and initiates the device control or monitor data sequence.

Between readouts of digital monitor data, the 74LS373 register outputs follow the "D" inputs. When the monitor register data is read out, the 373 load enable goes low and the latch contents remain static until the readout sequence is completed. The state of the monitor register is enabled onto the CON/Mon bus by the output enable. This register is connected to the bus only when RA specifies the

assigned digital monitor address.

When a command is executed, the leading (rising) edge of the load enable from the command address decoder will clock the state of the CON/MON lines into the command register (74LS374's). The output enable is tied low; the command register outputs are always available.

If R/W- is high and RA specifies an analog signal (RA: 000 through 007_8), when DEV REQ goes true (high) the device logic will return an ANENB signal to the interface because the high order address decoder Y_0 output makes the 74LS02 ANENB output high. This disqualifies the 74LS02 DEV ACK gate.

The eight analog signals are multiplexed by the analog multiplexer on the Standard Interface Board. The $00X_8$ decode and high state of R/W- enable ANENB which is fed back to the interface board. The board analog multiplexer will select the RA-designated analog signal for conversion. Since this version of device logic does not have analog multiplexers, HI/LO SEL is connected to logic common which causes the interface board analog multiplexer to be controlled by the three lowest RA bits. The analog signals may be either differential or single-ended depending upon the common-mode noise environment. In either case the connections to both versions of the standard Interface Board are identical.

If analog multiplexers are used in the device logic, the HI/LO SEL term would be generated from a decode of RA over the address range of 010_8 through the highest analog address. Using HI/LO SEL in this manner keeps the analog monitor data address space compact. See the description of the HI/LO SEL logic in Section 2.3 above.

Although it is not included in this generic design, it is often desireable to add a digital monitor data readout of the state of each command register; this command echo verifies that the interface and device logic is working. This can be done by feeding the command register outputs back into a monitor register.

The digital monitor circuitry of the generic design samples the monitor states at the time of readout. If the device digital monitor circuitry is used to read out the state of transient or intermittent phenomena (such as a phase-locked loop which might flip in and out of lock), the generic monitor circuitry would probably miss most of the transient events. In this situation, the digital monitor circuitry should be designed so that transient events are trapped in a latch for subsequent readout. The last stage of the readout sequence should clear the latch so that it can return to the transient sensing condition.

Another control possiblity is the use of a command address as an address-designated strobe to pulse some device function. In this case the command argument would probably be irrelevant.

The MSB in ADH distinguishes command addresses from monitor data addresses. Since this is a discrete bit, it is not considered a component of the addresses. Since R/W- is used as an enable for the two address decoders, there are no logic conflicts when command and data addresses are identical. If a monitor data readout is an echo of a command, identical addresses are desireable but when there is no clear correspondance between command and monitor functions it is generally best to use unique addresses (or blocks of addresses) to make it easier to distinguish between commands and data addresses.

2.5 INTERFACE BOARD ADDRESS CONVENTIONS

This section describes the conventions followed in the assignment of addresses to interface boards. Each Standard Interface Board in a device is assigned a unique block of addresses in the 32 K of available address space. The block starting address and block size of each interface is assigned by the Controller (i.e. antenna control computer) and the Controller can at any time reassign these addresses, the only constraint being that the blocks be unique and disjoint. Thus (under these two constraints) an interface which had been assigned a low starting address can be reassigned to a high starting address and vice versa. The board firmware stores the assigned address block start address and block size in the 8032 RAM memory in the form of block starting and block ending addresses. The board firmware is indifferent to the actual addreses values; when a new message is received on the XMT bus, the message address is compared with the two addresses stored in RAM. If the message address is within the block it is executed; if not it is ignored.

Each device is assigned a unique, hard-wired Block Identification Number (N) which is read by the interface during the power-up sequence as described above. This number is the vector for the assignment of block start address and block size by the Controller. The interface board firmware assigns RAM memory locations for the storage of 2N+1 (block start address) and 2N (block size). 2N and 2N+1 are addresses which can be written into by a command from the Controller (CDH and CDL contain the the value to be stored). These values in these two addresses can be read out as monitor data by the Controller (MOH and MOL contain the value being read out).

The power-on initialization firmware sets default values of 7FF0H for the block start address and 10H for the block length. At some time later, the Controller assigns the two working values to each interface in the system.

The interface firmware periodically re-reads the Block ID number to detect possible ID code faults.

The last 16 addresses in the block are for parameters peculiar to the Standard Interface board. The bus and interface board specifications require that the occurrance of bus fault conditions (such as parity errors etc.) be accumulated and be made available for monitor data readout by the Controller. The counters must also be capable of being reset by the Controller. These parameters were described in Section 2.1. These addresses are identified by the notation: BE-1, BE-2, etc. where BE designates the Block End (last) address. The assignments are as follows:

Address	Value
BE-15	Reserved for future use
BE-14	" " " "
BE-13	" " " "
BE-12	No Control Response counter (i.e, no DEV ACK from device)
BE-11	No Monitor Response counter (i.e., no DEV ACK or ANENB from device)
BE-10	**Interface Type and revision code (cannot be altered by the Controller)**
BE-9	Address of last control message received. (i.e., ADH and ADL)
BE-8	Control data for last control message received. (i.e., CDH and CDL)
BE-7	Address parity error counter, all messages
BE-6	Control data parity error counter, all messages
BE-5	Invalid SYN character
BE-4	Control data parity error counter, messages in block
BE-3	**N, ID byte value from device logic (cannot be altered by the Controller)**
BE-2	Count of correctly received control messages
BE-1	Count of correctly received monitor data request messages
BE-0*	**Address of beginning of block (cannot be altered by the Controller)**

* BE-0 cannot be directly commanded to another value. It is set to a new value by the action of

the 2N and 2N + 1 firmware described in Section 3.3.

With the exception of the addresses designated by bold print, all of these addresses can be written into by a control message from the Controller. All addresses can be read out by a monitor data request.

The discussion above is focused on the addresses of an individual interface; we now consider the address space for all interfaces. Figure 22 (below) shows the address space for all interfaces with several example devices.

Tabulated below are N, Block Start Address (2N + 1), Block Size (2N) and device for all devices at the VLBA antenna sites (status of 6/90).

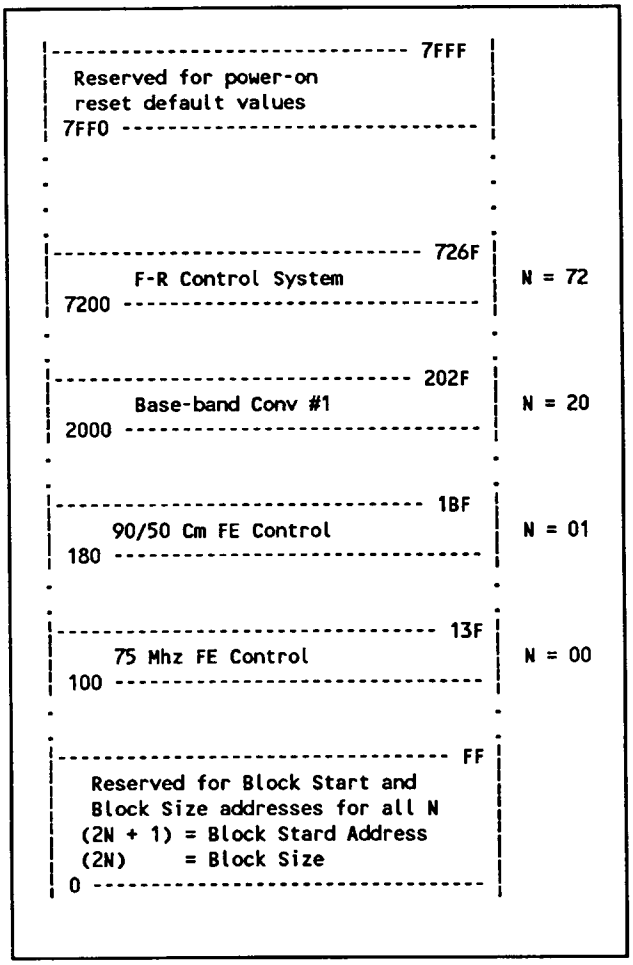


Figure 22, Interface Board Address Space (Hex)

VLBA Antenna Site Block ID (N), Block Start (2N + 1), Block Size (2N) (hex) values, and Devices, status of 6/90

N	2N + 1	2N	Device
00	100	40	75 Mhz FE Interface
01	180	40	90/50 cm FE Interface
02	200	40	20 cm FE Interface
03	280	40	13 cm FE Interface
04	300	40	6 cm FE Interface
05	380	40	4 cm FE Interface
06	400	40	3 cm FE Interface
07	480	40	2 cm FE Interface
08	500	40	1.3 cm FE Interface
09	580	40	7 mm FE Interface
0A	600	40	4 mm FE Interface
10	1000	20	2-16 Ghz Synthesizer #1
11	1100	20	2-16 Ghz Synthesizer #2
12	1200	20	2-16 Ghz Synthesizer #3
14	1400	30	LO/IF/RF Switch Controller
15	1500	30	Round Trip Phase Monitor
16	1600	40	Maser Interface
18	1800	80	Weather Station
20	2000	30	DAR-1 Baseband Converter #1
21	2040	30	DAR-1 Baseband Converter #2
22	2080	30	DAR-1 Baseband Converter #3
23	20C0	30	DAR-1 Baseband Converter #4
24	5000	30	DAR-1 Baseband Converter #5
25	5040	30	DAR-1 Baseband Converter #6
26	5080	30	DAR-1 Baseband Converter #7
27	50C0	30	DAR-1 Baseband Converter #8
28	2100	30	DAR-1 IF Distributer #1
29	2180	30	DAR-1 IF Distributer #2
2A	2200	100	Recorder Transport #1
2C	2300	100	DAR-1 Formatter
30	2400	30	DAR-2 Baseband Converter #1
31	2440	30	DAR-2 Baseband Converter #2
32	2480	30	DAR-2 Baseband Converter #3
33	24C0	30	DAR-2 Baseband Converter #4
34	5100	30	DAR-2 Baseband Converter #5
35	5140	30	DAR-2 Baseband Converter #6
36	5180	30	DAR-2 Baseband Converter #7
37	51C0	30	DAR-2 Baseband Converter #8
38	2600	30	DAR-2 IF Distributor #1
39	2680	30	DAR-2 IF Distributor #2
3A	2800	100	Recorder Transport #2
3C	3C00	100	DAR-2 Formatter
40	4000	40	Rack B Interface
42	4200	80	Building Utility Module
41	4100	80	Pedestal Room Utility Module
70	7000	40	Antenna Control Unit
72	7200	70	Focus Rotation Drive
43	4300	80	Inclinometer Interface (PT only)
44	4400	90	MKII Interface

2.6 INTERFACE - DEVICE TIMING

This section describes the interface - device timing which is depicted in Figure 23 (next page). The three types of interface - device message timing operations are shown. The firmware operations are identical for the execution of data request messages for digital and analog data but the board circuitry operations are quite different. Both cases are shown in the figure. Device non-response conditions are also shown. To simplify the figure, the timing diagrams are not drawn to scale.

The most important term in these timing signals is DEV REQ which the device logic uses to enable device logic operations. DEV REQ goes true many microseconds after RA and R/W- are set and is cleared many microseconds before R/W- is reset. The device logic can use microprocessors and hard-wired digital logic because of the very large timing margins.

For simplicity, the BUFR ENBL signal is not shown on Figure 23 but has the same timing as R/W-.

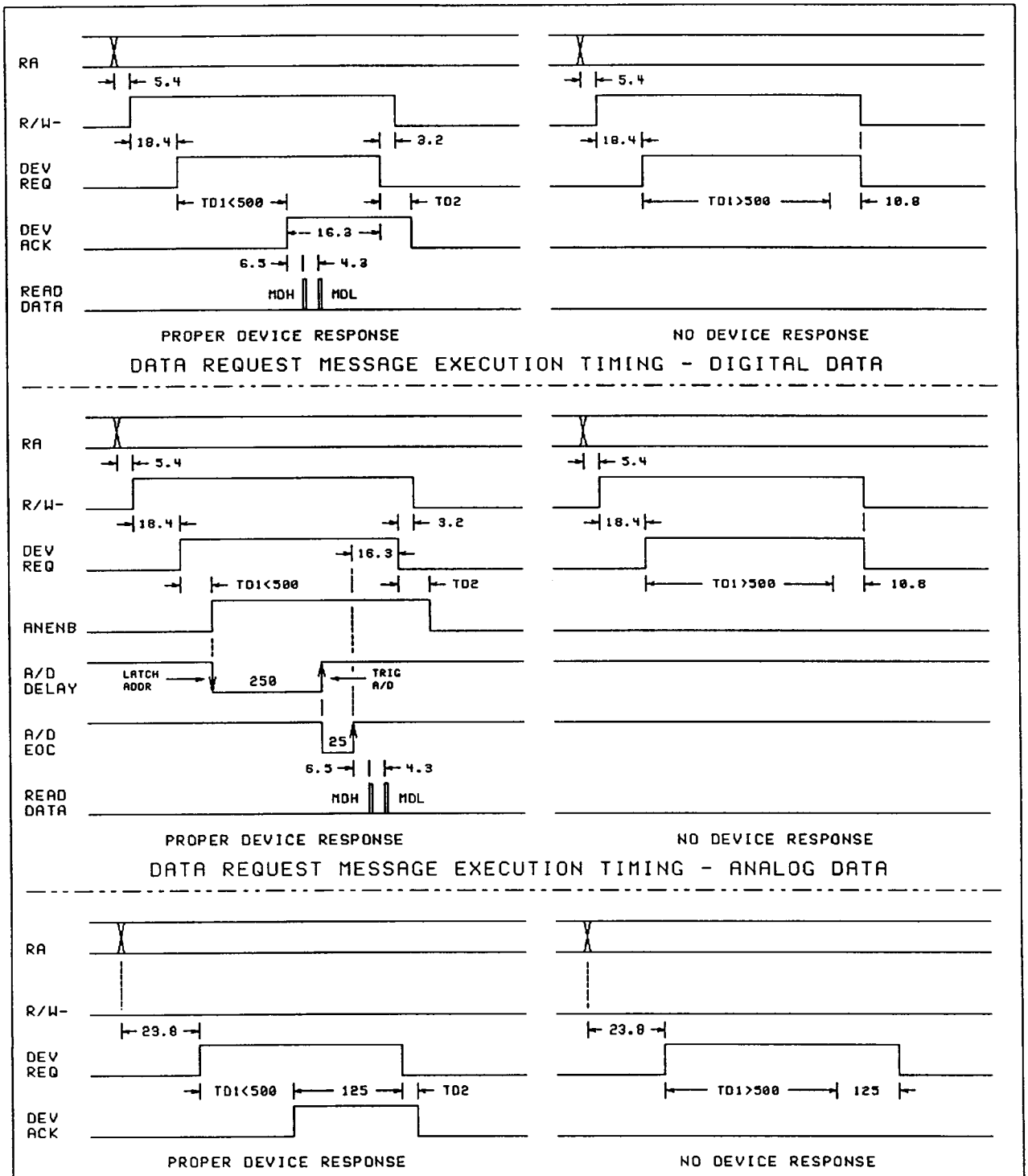
The interface timing is primarily determined by the control firmware which is described in Section 3.3. The I/O operations performed by the firmware are fully detailed in this section. The R/W-, BFR ENBL, IOR and CMD REL signals are set into the 8755 Port B latches. These latches have glitch-free transitions. DEV REQ is derived from IOR and CMD REL. DEV REQ is set and cleared by instructions which follow and precede (respectively) the set up and clear of R/W-, BFFR ENBL, CMD REL and IOR (see the logic schematics in Section 5). The DEV ACK and ANENB signals are sensed by 8156 Port C strobe inputs which set flags that are tested by the firmware. Sections 2.3 Interface Board Logic and 2.4 Typical Device Logic detail the operation of these signals and usage in typical device logic. The operation of the 8755 and 8156 I/O ports are described in Section 2.2.

Note that in the case of device non-response, all of the 8755 Port signals are cleared at the same time. Since the device did not respond (it was probably dead) within the allotted 500 microseconds, there is no need to clear DEV REQ before the other signals.

Note that RA remains set at the message value until the next message is serviced. The quiescent (i.e. between messages) state for R/W- is low.

Two device logic response times are shown: TD1 and TD2. TD1 is the DEV ACK or ANENB response time when the DEV ACK line goes true. The firmware permits device responses within 500 microseconds; the device is considered non-responsive if TD1 exceeds this period. TD2 is the device logic response time when DEV REQ returns low.

The ID code read operation is not shown on Figure 23 because it is very simple and does not involve handshaking. The ID value is sampled twice by the firmware and the total period for the Device ID code read operation is about 6.5 microseconds. For the first sample, the device logic must impress the ID value upon the CMD/MON bus within approximately 2 microseconds after the interface drops the ID REQ line low. The second sample is taken about 2 microseconds after the first sample. Two microseconds is many times greater than the typical TTL device ID logic response time. The device logic should disconnect the ID Code drivers within a few microseconds after the ID REQ line returns high.



NOTES:

1 TIME IN MICROSECONDS
2 NOT TO SCALE
3 TD1 AND TD2 ARE DEVICE LOGIC RESPONSE TIMES

FIGURE 23, INTERFACE SIGNAL TIMING DIAGRAMS

2.7 BAUD RATE GENERATION

In the VLBA Standard Interface application, Timer 1 is used in Mode 2 to generate the 57.6 Kilo-baud clock rate. The SMOD and TH1 parameters are adjusted in the following equation to reprogram the EPROM to a lower baud rate. Baud rate = $(2^{SMOD}/32)$ x (Oscillator Frequency)/(12 x [256 - (TH1)]) The oscillator frequency is 11.0592 Mhz.

To program the baud rate to the lower standard rates the SMOD (in PCON), C/Tbar (in TMOD) and TH1 (in SFR address 8DH) parameters should be set to the values shown in the table below. The assembly language program addresses where these parameters are set are 037EH through 038AH. The program listing is contained in Section 3.4. The operation of this section of the program is described in Section 3.3.

	Timer 1				SPO01 & SPE01
Baud Rate	SMOD	C/Tbar	Mode	TH1	DELAY COUNT
19.2 K	1	0	2	FDH	150*
9.6 K	0	0	2	FDH	300**
4.8 K	0	0	2	FAH	600**
2.4 K	0	0	2	F4H	1200**
1.2 K	0	0	2	E8H	2400**

* See the description of the SPO01 and SPE01 routines in Section 3.3. Adaption to the 19.2 K rate requires only modest changes to the firmware.

** The 8-bit counter in the SPO01 and SPE01 routines must be changed to a 16-bit counter, a more extensive change than for the 19.2 K case above.

2.8 BOARD ALIGNMENT AND TEST

This section describes the alignment and functional tests performed on the VLBA Standard Interface Board. The following is a brief description of the test fixture (MAXTESTER) and outline of the tests; a more extensive description of MAXTESTER and the tests is beyond the scope of this manual.

MAXTESTER is a microprocessor-controlled test fixture which contains a keyboard and CRT terminal; the operator inputs values and commands via the keyboard and MAXTESTER values are displayed on the CRT screen. During the course of construction, MAXTESTER is used to test the interface boards to verify performance and to align and test the analog multiplexer and A/D converter.

MAXTESTER functions as both a Controller and Device. Functioning as a Controller, MAXTESTER synthesizes valid and invalid XMT bus messages. Invalid bus messages are composed of error-tainted SYN, ADL, ADH, CDH and CDL message components. The Controller compares the CDH and CDL values read into the Device with the value sent by the Controller. The interface's response to these messages is evaluated by the Controller as it reads the RCV bus messages. The parity of the RCV bus ACK, NACK, DC1, DC2, MOH, MOL and the content of MOH and MOL are evaluated by the Controller. In the event of a RCV bus parity error, the tester displays a DC2DC2! message on the CRT screen.

Functioning as a Device, MAXTESTER reads control message RA and the CMD/MON bus, interacts with the control and handshake lines and simulates Device non-response by inhibiting DEV ACK.

In the Device role, MAXTESTER simulates data request message functions (for digital monitor data) by reading the RA bus states, impressing a programmed state on the CMD/MON bus, interacting with the control and handshake lines and simulating Device non-response by inhibiting DEV ACK.

Also in the Device role, MAXTESTER simulates data request message functions (for analog monitor data) by reading the RA bus states, injecting an analog signal into the analog multiplexer for conversion, interacting with the control and handshake lines and simulating Device non-response by inhibiting ANENB. Analog switches connected to the board's analog inputs can inject an analog signal or power supply voltages into any of the analog multiplexer inputs. Using keyboad-input values, a 12-bit DAC synthesizes analog signals for input to any of the multiplexer channels.

The Device circuitry also contains Controller-commandable ID code logic which may be set to any value. The Controller can also control the Device's interaction with the interface's discrete control-handshake signals. The interface's DEV REQ and R/W- can be evaluated and the interface's response to Device-inhibited DEV ACK and ANENB can also be evaluated.

Since an external bus reset is one of the interface board features, MAXTESTER can test the board response to a bus reset signal.

MAXTESTER uses both digital state and bit walk tests. Bit walk tests are "walking bits" in which a single "1" sequences through parallel data input or output lines. All other bits are set to zero and the response lines are tested for a spurious response to the single one. This test is applied to the RA, CMD/MON and ID byte buses.

The interface board under test is the only board on the two buses; bus signal levels and loading are not tested.

MAXTESTER does not test interface board logic threshold levels and currents. Device logic loading is one LS TTL load.

A test menu permits the selection of single or recurrent tests of any of the specified types. Tests may be executed 1, 100, 1000 or 10,000 times.

The test menu has the following tests in the order shown:

1) Response of the interface to a bus reset command.

2) Response of the interface to a changed ID byte. ID byte parity is tested.

3) Analog tests in response to stimulus values input from the keyboard. These values can be injected into to any multiplexer input. Test signals are: (analog) GND, a keyboard-specified value, -15/2, +15/2, +5, -10 and +10.

4) Response of the interface to an ID byte bit walk. The ID parity bit is tested for proper parity.

5) Response of the interface to a change in address block size.

6) Response of the interface to a change in the address block starting address.

7) Response of the interface to control messages with walking 1's in RA and CDH, CDL.

8) Standard internal address test sequences (for the 16 internal interface control and monitor data functions).

    a) Normal message sequence consisting of 1 control message and 17 data request messages per cycle.
    b) Bad sync parity, a test of a message with an invalid SYN code, BE-5 address.
    c) A test of a message with a valid SYN but ADH or ADL tainted by a parity error, BE-7 address.
    d) A test of a message with valid SYN and ADH, ADL but CDH or CDL tainted by a parity error, BE-6 address.
    e) Block Address control and monitor data request messages and the interface response. Addresses 2N and 2N+1.
    f) Out of bounds address messages, i.e. response of the interface to messages outside the assigned address block.
    g) Response of the interface to Device non-response to a control message, BE-12 address.
    h) Response of the interface to Device non-response to a data request message, BE-11 address.

9) MW processor - a sequence of all RA and CMD/MON bus states and walking 1's on the RA and CMD/MON bus.

10) Select control message or data request message modes for the Controller. ADH, ADL and CDH, CDL values are input from the keyboard. MOH, MOL values read from the Device are displayed on the terminal CRT screen.

11) Abnormal (similar to Internals above)

12) ANENB check

13) DEV ACK check

14) R/W- check

15) DEV REQ check

16) Check power. A recheck of +/- 15, +5 and GND.

2.9 POWER REQUIREMENTS

The 5 Volt and +/- 15 Volt power requirements (in milliamps) are tabulated below.

Power	Version "S"	Version "D"	Either Version, No analog chips
5 Volts	700	700	650
+15 Volts	30	30	0
-15 Volts	32	60	0

3.0 FIRMWARE DESCRIPTION, ALGORITHMS AND PROGRAM LISTINGS

3.1 INTRODUCTION AND FIRMWARE OVERVIEW

The VLBA Standard Interface firmware described below is the August 2, 1988 version which was written by Wayne Koski.

After power-up initialization, there are only three major functions performed by the firmware: CMD (command to execute control messages), MON (to execute data request messages) and STATUS, to execute the 2N and 2N + 1 address block assignment control and data request messages. The main control program LOOP calls these intermediate routines as it decodes the incoming messages. The firmware description is a narrative commentary on these five code functions.

Users desiring to make changes to the code or adapt it to non-VLBA tasks should find it fairly easy to do so because of this topical organization. One example of such a change is to reduce the baud rate to lower values. Another example is to simplify the address block logic to use the ID code as a simple address reference value rather than the complicated 2N and 2N + 1 scheme used in the VLBA application. Another example is to simplify the bus protocols by reducing the number and types of function code feedbacks on the RCV bus.

Readers who have studied the interface board specification (Section 6.0) may have noted a requirement for provisions to link to a background task which operates when the interface is not servicing bus messages. This capability has not been implemented (although a very limited background task probably could be added to the VLBA firmware). There are two reasons for this omission. First, the basic concept of this interface is that it must be interchangeable throughout the VLBA antenna site without alteration or special adaptation. Specially-programmed versions would militate against this interchangeability and they would certainly end up in the wrong module. Secondly, the CPU time available for background tasks is so small that it probably would not be very useable and there is a real risk that the background code could interact with the tightly time-constrained bus service routines. Very careful programming would be required to add a background task capability to the VLBA firmware and the resultant code would have to be carefully checked in the worst case of VLBA bus activity. If it is really necessary to add a background task capability to this VLBA firmware, a new physically distinct board should be made with additional program memory. An 8752 version with 8 K of on-chip EPROM might be more appropriate for the processor. Another very desirable feature would be the incorporation of recently-developed, 5-volt power threshold comparators that detect power glitches and issue reset strobes before the power excursion exceeds the operating voltage range of the CPU and memory chips. In non-VLBA applications, operation at lower baud rates would ease bus service time constraints so that a background task capability would be much more feasible. The ability to jumper-select either RS-485 or RS-232 line drivers and receivers would increase the board utility but the RS-232 applications would be restricted to single-interface board applications.

The reader should have at hand the program listing (Section 3.4) and a copy of Intel's 8-bit Embedded Controllers data book, the 1991 version preferably, but an older version will do. Since this manual does not contain the instruction definitions, the Intel data book is vital for a proper understanding of the operation of the firmware. This manual is not a tutorial on MCS-51 programming; the reader must be capable of interpreting the instruction operations. However, as an aid to understanding the operation of the firmware, interesting examples of instruction usages are briefly described. Throughout this commentary, the reader should constantly compare the program listing with the description and at places of interest, decode the numeric op codes to verify the description.

The firmware was assembled using a Version 2.0a, AD2500 cross-assembler for an 8051 in a CP/M Northstar PC. The 8032 has capabilities not included in the 8031 so it was necessary to "patch" the code by macros to use these features. Although this patching may look awkward in the program listing, the performance of the 8032 is not impaired by this stratagem.

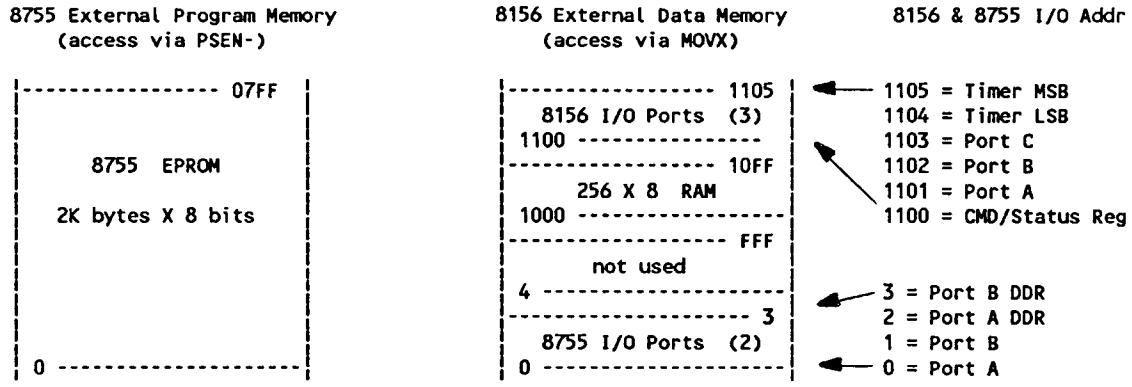
The firmware description first describes the initialization code and subroutines because they perform very important but specialized operations. With the insight gained by study of the subroutine descriptions, the reader will find the higher level code functions, CMD (command), MON (monitor) and STATUS to be very straight-forward.

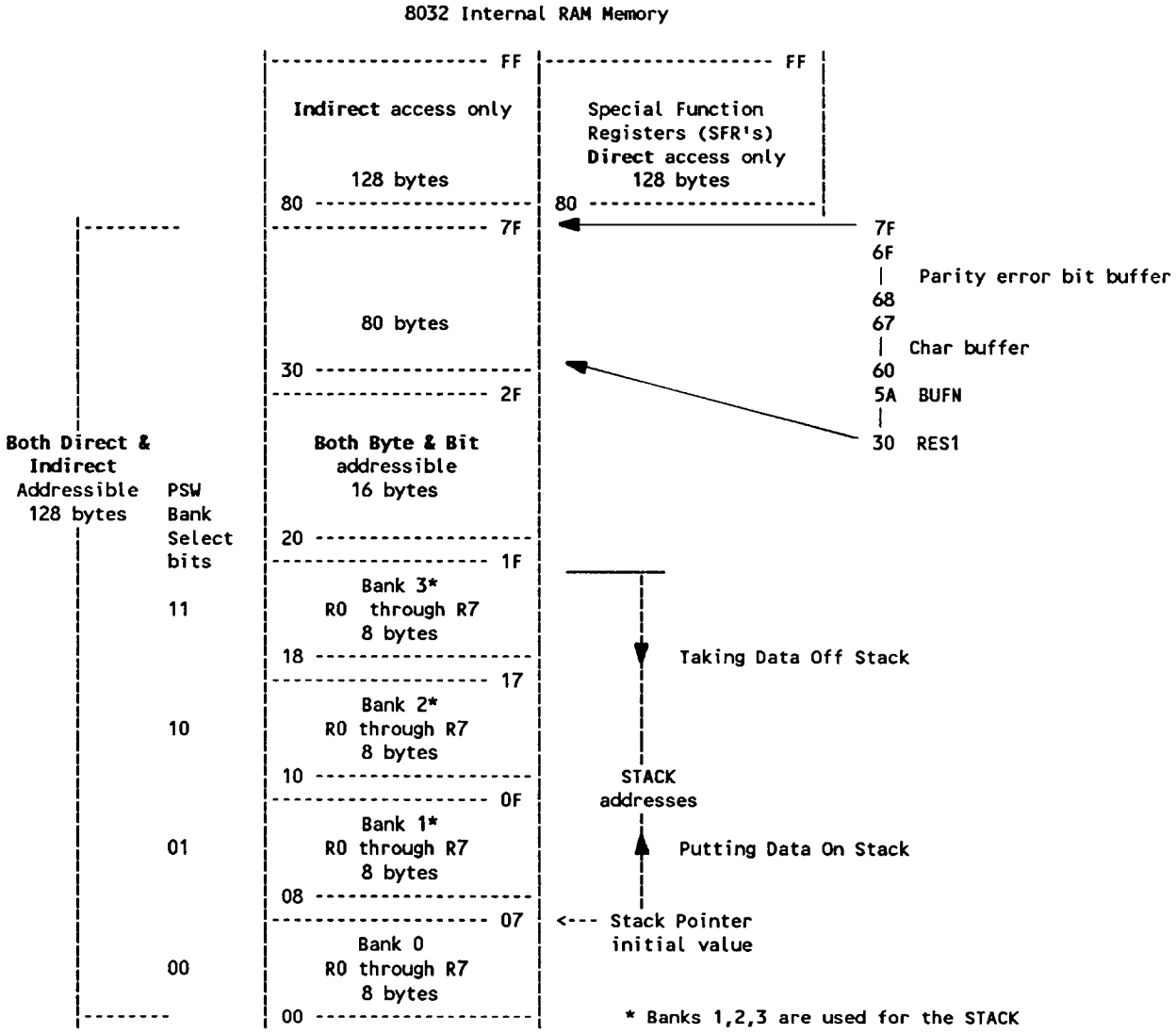
The description is organized into topical sections consisting of four parts:

1) A general functional description of the operations performed by the code section under discussion. ***
2) A detailed description of the code operations.
3) A subroutine or code Algorithm.
4) A detailed explanation of an interesting or important instruction seen in the code section. The description is focused on one or more instructions. The purpose of this description is to show the details of CPU instruction execution. The explanations are not a programming tutorial.

In Section 3.3 below, address references are hex values set in brackets, e.g. START1 [0030].

3.2 PROGRAM MEMORY MAP





3.3 FIRMWARE AND ALGORITHMS DESCRIPTION

MACRO Instruction Description

The firmware uses several macro instructions to implement code. In some cases the macros synthesize instructions for 8032 features (e.g. Timer 2) for which there are no provisions in the 8051 assembler. In another case the XORB macro implements an exclusive-or function for two boolean variables. The macro instructions are injected into the source code when the macro name is invoked. Comment lines may be included in the macro if they are preceded by a semicolon.

Structurally, the macro is a macro declaration (MACRO NAME:MACRO) followed by one or more lines of code and terminated by the ENDM instruction. For example, the SETPCON macro is as follows:

```
SETPCON:MACRO
;THE SETPCON MACRO IS USED TO DOUBLE THE BAUD RATE OF THE
;SERIAL PORT
;ENTRY          NONE
;AFFECTS        REGISTERS: PCON

DB   75H,87H,80H
ENDM
```

In this macro the DB (define byte) operator is used to synthesize instructions to set the SMOD bit in the PCON register. Clearly, the synthesized instructions must be correct in every respect or the program will not operate as expected.

A macro can use conventional mnemonic instructions or a mixture of synthetic (DB) and mnemonic instructions. For example the XORB macro uses only mnemonic instructions. Variables peculiar to the macro may be declared following the macro declaration. For example the XORB macro declaration and variables are: XORB:MACRO ARG1,ARG2,ARG3 where the three variables are peculiar to the XORB function only and cannot be referenced outside the macro. The operation of the XORB macro is described below.

Data Definition Area Description

The program DATA DEFINITION AREA on program listing pages 2 and 3 tabulates internal RAM address assignments, equates, and Special Function Register (SFR) register-label and bit-label assignments. These are described below.

Reference Values

Equates MODEL and REV define the Standard Interface Board model and revision level. These parameters are injected into the table of Standard Interface Board parameters described in Sections 2.1 and 2.5.

Equate BASE is a reference value (7FF0H) that is the default block start address which is used by the firmware until set to the assigned value by the Controller. POINTS is the default block size (16D) until set by the Controller. The BASE value is also used as a reference value in INTIAL [034C].

Internal RAM Assignments

The internal RAM is used for storage of a number of byte and discrete parameters which are accessed by the firmware during the course of program execution. The RAM address-label assignments are shown on pages 2 and 3 of the program listing in Section 3.4. The assignments and functions of this table are as follows.

RAM locations 30H through 47H are dedicated to the table of 16 Standard Interface Board parameters (described in Sections 2.1 and 2.5). The first 6 internal RAM locations in this table are reserved for future use. In this table, locations 3AH and 3BH are set to Model and Revision values. This table is cleared by INTIAL [034C], which also sets the Model and Revision values.

Locations 4EH through 55H are dedicated to address block parameters (e.g., 2N + 1, N values) that are set by the NUMBER subroutine [052E] described below.

RAM locations 56H and 57H are used for timer parameters TIME1 and TIME2 which are set by SYNC [03B7].

RAM locations 59H and 5AH are input message character pointers BUFFC and BUFFN that (respectively) point to the current and next addresses in the RAM character buffer. The character buffer is RAM addresses 60H through 67H. XMT message bytes are stored in this buffer by SERP [0483] as they are input from the serial RCV port. The parity state of these bytes is stored (as bytes) in the character parity error buffer in addresses 68H through 6FH.

RAM address locations 20H through 2FH may be used for both bit and byte storage of parameters. The program uses locations 20.1 through 21.1 for storage of discretes and flags. The usage of these discretes is distributed through the firmware description below as they are encountered.

Register Assignments

Bank 0 registers are assigned functions and storage as follows:

R0 = general use, RA LSB storage, indirect MOVX instruction index register
R1 = general use, RA MSB storage
R2 = ADL, MDL storage
R3 = ADH, MDH storage
R4 = CDL storage
R5 = CDH storage
R6 = not used
R7 = not used

SFR Register-Label, Bit-Label and Flag Mnemonic Assignments

SFR registers PSW (address D0H), SCON (address 98H) and TCON (address 88H) are both bit and byte addressable. The program frequently accesses many of these discretes during the course of execution. The REG assignments from D0H through 9BH assign mnemonic labels to these functions. For example: the PSW address is D0H, bit 0 of PSW is the Accumulator (register A) parity bit; it is assigned the label P for location PSW.0. This mnemonic reference is a convenience because the label P (for parity) is less abstract than PSW.0. In a similar manner, bits in SCON and TCON are given similar labels: TI for Transmit Interrupt flag location (SCON.1), RI for Receive Interrupt flag location (SCON.0), TR1 for Timer 1 Run Control bit location, RB8 for Receive Data Bit 8 (parity bit) location, and TB8 for Transmit Data Bit 8 (parity bit) location.

Seven Port 1 bits are given similar assignments. MSG, BUSY, DOUT and PARX (bits P1.1 through P1.4 respectively) drive the Bus and Board Status Display Logic (described in Section 2.3). RCVEN (P1.0) enables the RCV bus driver chip. ADDEN (P1.5) is the discrete that activates the device ID REQ logic. This logic is described in Section 2.3.

INTIAL and Program Start

INTIAL [034C] is the subroutine called by an ACALL (absolute call) from the beginning of the program, START1. Before the firmware can begin to execute the tasks of executing control and monitor data request messages, the processor, registers, serial ports, timers, interrupt system, and RAM memory must initialized. The board reset logic resets the processor and registers as described in Section 2.3 but the conditions established by the processor reset must be altered to suit the mode and operation requirements of the firmware. INTIAL establishes the correct states and modes in the processor and I/O devices.

Note from the Reset description in Section 2.2 that register IE, the interrupt control register is set to 0; this disables all interrupts so the firmware is not vulnerable to a chance interrupt from any source. (This register is designated IEC by the assembler).

An overview of the INTIAL operations is presented below. This will be followed by a detailed description and the algorithm.

When the processor emerges from the reset state program execution starts at START [0000] and the first instruction is an AJMP START1 (op code 01 30) which is an absolute jump to START1. START1 is at address 30H. The jump destination must be within the same 2K code segment as the instruction. In the op code, the three high-order destination address bits are the top three bits of the leading 0. The 1 designates the jump and the second byte is the lower component of the destination address, in this case 30H. Thus the jump destination address is 030H.

START1 at location 30H is 5 memory locations above the highest interrupt vector address 2B (for TF2 and EXF2). This destination leaves space for interrupt execution code (described in Interrupt Code below).

Two vital functions are first performed at START1 [0030]. At START1 the Stack Pointer (SP) is set to address 07H. (The processor reset also set it to 7 but for protection from power perturbations, SP should always be intialized in INTIAL.). Next, the PSW is set to 0. This selects Register Bank 0 (PSW bits 3 and 4 are the bank select bits RS0 and RRS1 respectively) and clears PSW 5 which is a user-assignable status flag. PSW 5 is used as a glitch detector flag to provide some protection from the effects of power glitches on program execution - an important function in this firmware. The usage of PSW 5 is described in LOST below.

At this point, the processor is protected from perturbations but is not ready for program execution; the absolute call instruction, ACALL INTIAL, calls INTIAL to complete the initialization.

The first operation in INTIAL is to initialize the block of addresses (30H through 49H) which contain the Standard Interface Internal parameters described in Sections 2.1 and 2.5. These parameters can be read out as monitor data or loaded by a control message from the Controller. The first 10 internal RAM locations in this table (30H to 39H) are cleared to 0. RAM locations 3AH and 3BH are set to Model and Revision values. Next, RAM addresses 3CH through 4DH are cleared to 0; this is the upper part of the table. RAM locations 4EH through 55H are set to block start and block count values by NUMBER (described above/below). RAM locations 56H through 58H are used for timer parameters that are set by SYNC [039F] and TIMER [04B5].

Next, RAM addresses 3CH through 4DH are cleared to 0. This clears six reserved RAM locations: 4EH through 55H are set to block start and block count values by NUMBER (described above/below).

The default block start address (7FF0) is mathematically formulated and stored in BLOCK1,2. The default address count of 10H (16D) is stored in COUNT1,2. These values will remain in effect until set by the Controller.

Several simple but important states are set up: the interrupts are disabled, the RCV bus output and RCV bus enable are set to zero, and the 8755 DDR's are set to the output mode.

The subroutine NUMBER is called to read and store the value N (the interface Block ID code value).

Next, in the SETPCON [037E] section, the SETPCON macro sets the double baud rate bit (SMOD in PCON). Port 3 is set to alternate function. This sets the following: P3.0 to function as RXD, P3.1 to function as TXD, P3.2 to function as INT0-, P3.3 to function as INT1-, P3.4 to function as T0, P3.5 to function as T1, P3.6 to function as WR- and P3.6 to function as RD-. Note that before program execution starts, the 8032 power reset sets all port latches to the FFH state. This FFH state sets the alternate function port pins (P1.0, P1.1, P3.0 ..P3.7) to the alternate function mode. The commands (immediately above) to set the 8755 I/O port DDR latches to the output mode would not have worked if port had not set to the WR- function. The command to set the port 3 bits to the alternate function is redundant with the power reset but is still desirable to make the interface less vulnerable to power glitch effects.

Continuing the operations in the SETPCON section, SCON, the serial port control register is set to Mode 3 and serial reception is enabled. TMOD, the Timer 1 and 2 mode control register, sets Timer 1 to the timer function and to Mode 2. A count of 255 is set in TH1 which in conjunction with SMOD = 1, sets the serial port baud rate to 57.6 Kilo-baud. Timer 1 is started. The serial port transmit enable flag (BIT7) is set to permit transmissions on the RCV bus. The four board activity bits which set the front panel display LED memory are cleared by setting the bits true (the memory sinks current from the LED's so the bits are set to turn them off). The final operation in the SETPCON section is to set the XMT message character buffer indexes, BUFFC and BUFFN, to 60H. This is the address of the start of the character buffer.

INTIAL continues by entering the SET2CON [039F] section. The SET2CON macro clears the Timer 2 registers and turns on the timer. The serial port interrupt is set to the highest priority level and all interrupts are enabled. This is the last initializing operation of INTIAL. The microcontroller and support chips have been initialized, timers set up and running, the serial port baud rate generator is in operation and the port is enabled. The subroutine returns to the next instruction following the call which is the main control loop, labeled LOOP.

INTIAL program operations are simple and do not require extensive commentary. The initialized conditions must be correctly set for the program operations and the order of set-up is important.

RAM setting/clearing instructions use indirect move-immediate data instructions in which R0 is the pointer to the RAM address.

The RAM-clearing loops [034C and 035B] set the initial RAM address in R0. R1 is set to a loop count and an indirect move instruction sets the value 0 into the address pointed to by the contents of R0. After the output, R0 is incremented and R1 is decremented and tested for a zero value by a DJNZ instruction. If R1 is not zero, a jump to output another zero value is executed. If R1 is zero, control falls through the DJNZ to the next instruction.

R0 is incremented twice and the MODEL and REV values are stored in the two addresses pointed to by the address in R0.

R0 is incremented and at [0362], the default block start address (7FF0H) is calculated by assembler arithmetic. The value BASE (defined by the BASE: EQU 7FF0H psuedo-instruction on listing page 2) is divided by 100H (256D). The result is 7FH (the MSB of the default value) which is stored in RAM location 4EH (BLOCK1) by an indirect move to the RAM address pointed to by the address in R0. R0 is incremented again an the LSB of the default address is formed by assembler logic which and's BASE (7FF0H) with FFH. The result is F0H which is stored in the address pointed to by the address in R0 (BLOCK2).

A number of discrete control bits and flags (such as RCVEN, TR1, BIT7, etc.) are set or cleared by SETB or CLR instructions. The symbolic character of these bits and flags were briefly described above in the description of SFR Register, Bit and Flag Mnemonics. Since the sequence of operations is described above, additional detailed commentary is not required.

R0 is incremented again and the MSB (00H) of the default block size is set in the address pointed to by the address in R0 (COUNT1). R0 is incremented again and the LSB of the default block size is set in the address pointed to by R0. This value is 10H (16D) and is defined by the POINTS: EQU 16 assembler psuedo-instrution.

The setting of the 8755 I/O port DDR's are simple outputs of the value FFH. The 8755 properties were described in Section 2.2 and do not require additional commentary here.

The subroutine NUMBER [052E] is called to read and store the device ID code and to set up address block parameters. NUMBER is described below.

The SETPCON macro (see the General MACRO Instruction Description above) is a sequence of hex values which exactly replicate a MOV direct, #data instruction. The replicated op code is 75H, 87H, 80H. This sets the 80H bit in PCON which is the SMOD bit that doubles the serial port baud rate.

SCON and TMOD are set to the required state by MOV direct,immediate instructions in which the data is in binary format.

The SET2CON macro replicates a sequence of MOV direct, #data instructions which set the Timer 2 control register states. The Timer 2 SFR's RCAP2H address is CBH and the RCAP2L address is CAH. The T2CON address is C8H. The value 4 that is set in T2CON is T2CON.3, the EXEN2 bit, the external enable flag. (See the Timer 2 description in Section 2.2).

The assembler assigns the IEC label to the Interrupt Enable register.

The last instruction, RETI is an interrupt return that returns control to the address following the calling location, LOOP.

In the COUL loop in INTIAL [0350], the MOV @R0,#0 instruction that clears internal RAM memory locations, is an indirect memory reference with immediate data instruction that is used with R0 or R1. In the Intel 8032 data book, the opcode format is: 0111 011i immediate data where i designates register R0 or R1. The instruction moves the immediate data to the RAM address pointed to by the contents of Ri. In this case the opcode is 76 00, in which Ri is R0 and 00 is the immediate data. In the COUL loop, R0 is initialized to an address of 30H and R1 is loaded with a loop count of 10D. On each pass through the loop, after setting the value 00 into the address pointed to by R0, the address in R0 is incremented and the loop count in R1 is decremented and tested for the zero state by the DJNZ

R1,COUL instruction. If R1 is not zero, the jump to COUL is taken to repeat the loop operations. If R1 contents are 00, control falls through to the next instruction.

The INTIAL algorithm is a linear sequence of simple operations; there are no program branches.

```
BEGIN  INTIAL
        CLEAR RAM ADDRESSES  30H TO 39
        SET MODEL/REV LEVEL IN RAM 3AH, 3BH
        CLEAR RAM ADDRESSES 3CH TO 4DH
        SET RAM ADDRESS 4EH TO 7FH
        SET RAM ADDRESS 4FH TO F0H
        SET RAM ADDRESS 50H TO 0
        SET RAM ADDRESS 51H TO 10
        DISABLE ALL INTERUPTS
        TURN OFF XMT OUTPUT AND DISABLE XMT DRIVER
        SET 8755 PORTS A AND B TO OUTPUT MODE
        READ BLOCK ID CODE
        SET DOUBLE BAUD RATE BIT
        SET PORT 3 TO ALTERNATE FUNCTION
        SET BAUD RATE TO 57.6 K-BAUD
        START TIMER 1
        CLEAR LED DISPLAY MEMORY
        SET BUFFC AND BUFFN TO 60H
        CLEAR TIMER 2 REGISTERS AND START TIMER
        SET SERIAL PORT TO HIGHEST INTERRUPT PRIORITY
        ENABLE ALL INTERRUPTS
END
```

Interrupt Code

Two interrupts are used by the firmware: the serial port interrupt and the Timer 2 interrupt. The properties of the 8032 interrupt control system was described in Section 2.2. The reader is urged to briefly review this section before proceeding further.

The processor acknowledges an interrupt request by executing a hardware-generated LCALL to the appropriate service routine. The hardware-generated LCALL pushes the contents of the Program Counter onto the STACK (but it does not save the PSW or any other registers) and it reloads the Program Counter with an address that depends upon the source of the interrupt being vectored to. In the Standard Interface application the only interrupts ever enabled are the serial port and Timer 2; these are enabled by IEC.4 and IEC.5, respectivley in the IE Register. These two interrupts are enabled and disabled at several places in the firmware, notably MON [01A6], SERP [0483], READ [04E6], SYNC [03B7], TIMER [04B5] and other places.

The serial port interrupt vector is at 0023H and is an AJMP to SERP [0483] which reads an XMT bus character from SBUF in response to the RI flag. SERP also tests the TI flag to detect the completion of transmission of the character output on the RCV bus. SERP is described below.

The Timer 2 interrupt vector is at 002BH and is an AJMP to TIMER [04B5] which drives the firmware timers TIME3 (a 5-second ID code flag timer) and TIME1,2 which is a 1-minute SYNC timer. TIMER is described below.

Interrupt service routines are terminated by an RETI instruction which first pops the high and then the low order bytes off the STACK onto the Program Counter and decrements the Stack Pointer by two. Program execution continues at the resultant address, generally the address following an ACALL or LCALL.

In MON [01A6] the interrupt enable disable instruction CLR IEC.4 disables bit 4 of the IE register. In the Intel data book this is a CLR bit instruction with an opcode format of: 1100 0010 bit

address. The instruction opcode is C2 AC in which the C2 is obvious. What address is designated by AC ? The Interrupt Enable (IE) register is a bit-addressible SFR (see the SFR Memory Map in Section 2.3) in which IE.0 has an address of A8, IE.1 has an address of A9, IE.2 has an address of AA, ... and IE.4 has an address of AC. The counterpart to this instruction is SETB IE.4.

NUMBER

NUMBER [052E] is a subroutine that is called from INTIAL [037C], SYNC [03CC] and MON3 [0292] in to periodically read and store the Block ID code (N, labeled NUMB1,2) from the device logic. The first two calls service the operations of message detection and decoding, the third call is to obtain N whenever interface data function, BE-3 (ID value) is requested by a monitor data request message. Using the RAM values for Block Start address (2N + 1, labeled BLOCK1,2) and Block Size (2N, labeled COUNT1,2), NUMBER calculates the Block End address (EBASE1,2) and **device logic hardware block size** (IBASE1,2). (Remember that there are 16 addresses dedicated for interface internal functions above the device addresses). These are all 16-bit values where the 1 suffix denotes the MSByte and suffix 2 designates the LSByte.

Number is called by SYNC [03B7] when the ID Request flag (bit 6) is a 1. This flag is set by the 5-second timer, TIME3 in TIMER [04D0] which is driven by the Timer 2 interrupt. This flag is cleared after most of the operations in NUMBER have been completed.

The 8-bit Block ID code is read from the device logic by making P1.5 (labeled ADDEN) low which causes the device logic to impress the 7-bit + parity code on bits CMD/MON 0 .... CMD/MON7. The upper CMD/MON byte is also read but has no (presently) assigned function so it may have an indeterminate value. The two bytes are stored in RAM locations NUMB1 (MSB) and NUMB2.

The Block ID code is read into the 8156 ports in the simple input (non-strobed, Alt 1) mode. This requires that the 8156 command register be set to designate this mode. The two bytes are read and then the 8156 is reset to the strobed (ALT 4) mode.

After resetting the 8156 to the ALT 4 mode, NUMBER calculates the EBASE and IBASE values. EBASE is the **address of the start of the next block** (assuming that the address block of the next higher address interface block is contiguous). To illustrate this point, consider the example of a start address of 1000D (decimal) and a block size of 20D. The twenty block addresses range from 1000D to 1019D, address 1020 is **outside** the block. The calculated EBASE value is checked for out-of-range values and if found wrong (i.e., MS bit of EBASE = 1, all other bits not = 0), NUMBER stops processing and exits via a jump to LOST (described in LOST below). If EBASE is within range, control returns to the calling location.

The firmware must test EBASE for errors for both the **usual** case and **special** case values. The **usual** case for this EBASE comparison is to test that the MS bit of EBASE is not 1 and if it is, to test that all other bits are equal to zero. If the MS bit is a 1 and the other bits are not zero, there is an error and the interface must be re-initialized. In this **usual** case, EBASE is an assigned **interface address** in the range of 100H through 7FEFH. This **usual** case applies to the situation after the Controller has loaded the block start address (BLOCK) into address 2N + 1 and the block size (COUNT) into N. The **special** case is the default address situation during the interval between the time of interface power-up initialization and the time that the Controller loads the working block start address and block size into 2N + 1 and 2N, respectively. Power-up initialization firmware sets the default block start address to 7FF0H and the block size to 10H (**16 decimal**); the last address in this block is 7FFFH. One more count added to 7FFFH makes a value of 8000H which is an acceptable value for EBASE in this error-testing logic. If the MS bit of EBASE is a 1, the error case is EBASE > 8000H which indicates that EBASE is erroneous and the interface must be re-initialized. The firmware must assume that there is no particular time for the

Controller to set the working values into 2N + 1 and 2N. It could happen at any time and there could be a long wait for the address assignments. During this interval, the interface firmware must continue to operate although it cannot execute control or monitor data messages addressed to the interface's normal device addresses. The interface is in effect anonymous during this default-address interval (many other interfaces could also be set to the default values). An EBASE value greater than 8000H is numerically realizable in the microcontroller but is above the defined interface address range. This error-testing logic must operate in both the normal EBASE address range of 100H to 7FEFH and the default address range of 7FF0H through 7FFFH. 8000H is a plausible EBASE value in the special case because EBASE is the start address for the start of the next (apparent) contiguous block.

The algorithm for NUMBER is as follows:

```
BEGIN  NUMBER
         SET 8156 PA AND PB TO ALT 1, INPUT MODE
         ENABLE CON/MON BUS AND SET DIR TO INPUT
         READ AND STORE BLOCK ID CODE N, IN NUMB1 (MSB) AND NUMB2 (LSB)
         DISABLE CON/MON BUS AND SET DIR TO OUTPUT
         SAVE ID LSB IN R2, MSB IN R3
         BLOCK START ADDR (2N + 1 IN BLOCK1,2) + ADDR COUNT (N COUNT1,2) = BLOCK END ADDR (EBASE1,2)
                 IF  EBASE1 MS BIT EQ. 32768  THEN  GOTO NUM3
                         IF EBASE2  NEQ. 0  THEN  GOTO NUM3
                         ELSE  GOTO  NUM2
                 ELSE  GOTO  NUM2
NUM2     COUNT1,2 - 16  EQ. IBASE1,2
         SET ID REQ FLAG (BIT 6)  EQ. 0
         SET 4CH IN TIM3
NUM3     IF  PSW.5 EQ. 0  THEN  GOTO  LOST1
         ELSE  PSW.5  EQ. 1,  GOTO START, RE-INITIALIZE INTERFACE
LOST1    PSW.5  EQ. 1,  RETURN TO SYNC
END
```

NUMBER subroutine program operations are as follows.

The first operation in NUMBER [052E] is to set up the 8156 A and B I/O ports for simple (ALT 1 mode) input; this is done by the first two instructions. (See Section 2.2 for a description of the 8156 and 8755 I/O ports; note that the 8156 and 8755 are External Memory, which is only accessible by MOVX instructions. Section 3.2 depicts the 8156 and 8755 memory maps). The 8156 RAM and I/O ports are accessed by indirect addressing using the Data Pointer (DPTR) register (DPH and DPL) that is loaded with the value 1100H, the address of the 8156 control register. MOV DPTR,#1100H, op code 90 11 00 is a load data pointer with a 16-bit immediate data instruction. In the next instruction, A is loaded by a MOV A,#00H instruction (op code 74 00), which is an immediate data instruction. 00H is the value to be output to the 8156 control register to set Ports A and B to the ALT 1, input mode. The instruction MOVX @DPTR,A ([0539], op code F0) is a move external transfer of the A register contents (the command state for the 8156 control register) to the address pointed to by the contents of DPTR (the address of the 8156 I/O ports control register). As indicated by the @ symbol, this is an indirect move instruction because DPTR holds the address of the destination and A holds the data value.

This sequence is typical of the 8156-related operations in NUMBER. An 8156 register address is set (or incremented or decremented) in DPTR. A is loaded with a value to be output or A is a value read from the address specified by the contents of DPTR.

To cause the CON/MON bus data flow to be from the device to the 8156 A and B ports, the 245 bus buffers must be enabled and the direction specified. This is accomplished by setting the 8755 Port B ports BUFR ENBLE (PB4) and R/-W (PB6) bits. The 8755 I/O addresses are shown on the memory map and the Port B address is 0001H. To output these bits to the port (the 8755 ports are always set to output), an indirect address MOVX (move external) instruction is used. The value 50H is loaded into A,

the DPTR is set to address 0001H, and MOVX @DPTR,A loads 8755 Port B with these two bits. This operation is very similar to the 8156 port and I/O operations.

Having set the 8156 I/O ports mode, enabled the CON/MON bus and set the data flow direction, it is necessary to set the 8156 Port A address into the DPTR; this is done by MOV DPTR,#1101H. The ID code value (LSB) is read by the MOVX A,@DPTR, which loads A from the location DPTR address. The value is read twice (to be sure) and stored in NUMB2 by a MOV NUMB2,A instruction, a direct address move A instruction. NUMB2 is (8032 internal) RAM address 0049H. To read the MSB of the ID code value, the address in DPTR is incremented to point to 8156 Port B and another MOVX instruction reads Port B into A and the value is saved in NUMB1.

The last step in the ID code read process is to return the ID Request signal (P1.5) to the high state. This is done by the SETB ADDEN (op code D2 95) instruction. ADDEN: REG P1.5 (in the data definition area, page 2) equates ADDEN to 8032 I/O port P1.5.

Having read and stored the ID code, the 8156 I/O ports need to be reset to the ALT 4 mode for normal (i.e., ALT 4, strobed I/O) mode. The ID code read operations above left DPTR set to a value of 1102H; it must be reset to 1100H, the address of the 8156 I/O ports control register. Two DEC DPTR instructions set the DPTR to 1100H (Command Register address). A is loaded with the value 3BH which sets the ports to ALT 4 mode, output mode.

The 245 buffer is disconnected by outputting a value of 00 to the 8755 Port B via the now-familiar sequence of setting DPTR to 01H (the address of 8755 Port B) and outputting a value of 00H (in A) by a MOVX @DPTR,A instruction.

NUMB1 and NUMB2 (the ID code MSB and LSB, respectively) are set into Bank 0 registers R3 and R2 for possible use when BE-3 is requested by a monitor data request message. The instructions are MOV Rn, direct instructions, (op codes AB 48 and AA 49 respectively). The three lower bits in the B and A designate the register number. B is 1011; thus the 011 designates Register 3 and A is 1010 in which the 010 designates Register 2. 48 and 49 are the addresses of NUMB1 and NUMB2 in the 8032 internal RAM memory.

Using the block start address (BLOCK1,2) and address count (COUNT1,2), NUMBER calculates the block end address (EBASE1,2) and the last device address (IBASE1,2). Remember that the last 16 addresses in the block are interface internal functions that are included in the block count. After the calculation, the value of EBASE is checked for magnitude consistency since no address can exceed the value of 32,768D.

At [055B] the instruction ADD A,COUNT2 (op code 25 51) adds the LSB of the block count to the LSB of the block start address (BLOCK2) in A. COUNT2 is in 8032 RAM location 51. The Carry Flag (CY, PSW 7) could be set as a result of this addition. The sum is set in the lower byte (DPL) of DPTR. In a similar manner the MSB of the address count (COUNT1) is added to the block start address (BLOCK1), but the add instruction is the ADDC which adds both operands and the CY flag from the previous addition. The addition results are in A; the sum is loaded into DPH, the upper byte of DPTR. DPTR is used as a temporary storage register for the impending consistency tests below.

We now consider the logic that tests the EBASE value for the usual and special cases described above. At [0565], the JNB A.7,NUM2 conditional branch instruction (op code 30 E7 07) causes a jump to the relative address offset (in this case 07, to NUM2) if the bit (bit A.7 in this case) is not set. A still contains the upper byte of the address sum formed in the previous paragraph (i.e., sum of block start address and address count MSB's). The interface address conventions are that no address can exceed 7FEFH, the address one count less than the power-on reset default block start. Bit 7 has the weight of

32768D or 8000H. If A.7 is set, either the stored block start address or address count (maybe both) or something else is erroneous. In the event that the bit is set and the jump to NUM3 is taken, control is transferred to LOST (described in LOST below). Calculation of EBASE1,2 and IBASE1,2 is bypassed and the previously set values will be used when ADDCK is called to compare the incoming message addresses with the block start (BLOCK1,2) and block end (EBASE1,2) addresses.

If A.7 is not set, control falls through to the next consistency test, conditional jump instruction CJNE A,#80H,NUM3 (op code B4 80 1C), which compares the contents of A (from above) with the value 80H. This is a compare, A, data value, relative address jump in which the data value is 80H and the (positive) relative address offset is 1C. The CJNE instruction is at [0568] and NUM3 is at [0587], 0568 + 1C = 0587, the address of the NUM3 instruction if the jump is required. The logic of this test is to again verify that the address byte = 80H.

If the CJNE jump is not taken, the next test is a comparison of the LSB of the sum still standing in DPL. The conditional jump instruction JNZ NUM3 (op code 70 18) transfers control to NUM3 if the contents of A are not 0. The jump address, NUM3 is a relative (positive) address displacement of 18 from the address of the JNZ NUM3 instruction.

If the consistency tests above are passed, the calculated values of block end address in DPTR are stored in EBASE1,2. The carry flag (PSW 7) is cleared; it would have remained set if the sums had set it. It is best to clear this bit so that it does not confuse the forthcoming SUBB instruction.

At [057A], 16D is subtracted from COUNT2 (the LSB of address count); this may set the borrow (carry flag) if a borrow is required for bit 7. The instruction is SUBB A,#16, op code 94 10. The instruction is of the form: SUBB A,#data where data is immediate data and the hex 10 value is 16D. The instruction arithmetic is: (A) <- (A) - (CY) - #data. The result is put in IBASE2. COUNT1, the MSB of address count, is put in A. Another subtract instruction, SUBB A,#0 subtracts the value 0 and permits the borrow from the first subtract to operate on the MSB. The result of this subtraction is the highest device logic address which is stored in IBASE1,2.

The next instruction is CLR BIT6. BIT6 is the ID BYTE REQUEST FLAG that caused SYNC to call NUMBER. Since the ID Byte has been read and caused the other operations to be performed, it is appropriate to clear the flag.

Before returning, the value 4CH (76D) is loaded into the 8032 RAM location TIME3 and address 58H. This value is loaded into Timer 2 to form a 5 second time out in subroutine TIMER.

NUM3 is the exit from NUMBER and the last operation is to test the state of the "Lost" flag, PSW.5. The JNB PSW.5,LOST1 instruction tests this bit. If set, it indicates that the program execution has been perturbed (perhaps by a power glitch) and control falls through to the LJMP START instruction to return control to the START location, which reinitializes the interface. If the bit is not set, a normal subroutine RET is executed to return control to the location at which it was interrupted by the Timer 2 interrupt.

The operation of the LOST flag code is described below.

In NUMBER the instruction MOV TIME3,#4CH is a MOV direct,#immediate data instruction. The Intel data book instruction format is: 0111 0101 direct address immediate data. In this case the opcode is 75 58 4C where 75 designates the MOV direct,#immediate data instruction, 58 is the (hex) address of (symbolic label) TIME3 (in 8032 RAM) and 4C is the immediate data hex value that is loaded into 58H (TIME3).

NACK, DC2, ACKN, DC1, SPEOUT, and SPEXIT

This set of routines can be described as a single entity since they perform the function of feeding back one of the four single-byte function codes to the Controller via the RCV bus. Logic in MON and CMD (the monitor and command executors, described below) does an absolute call (ACALL) to these locations to output a function code byte as a function of the device-interface interactions. The function codes formats and usage are described in Section 2.1.

This set of routines fit into a tree-like structure with four entry points which converge to a single exit. The entry points are: NACK [041A], DC2 [041E], ACKN [0422] and DC1 [0426]. The four function code entry points output even parity codes via the SPEOUT [0428] routine. If there is a serial port TX fault, a 50-count time-out loop in SPEOUT causes a jump to SPEXIT [0457] which does a mini-initialize and returns control to LOOP [0038], the entry point of the program control loop (described below). The 50-count time-out loop provides protection from the effects of a power glitch which might perturb the state of SCON. If SCON is altered, the serial port modes would change and the TI flag could be inhibited which could hang up the processor in this loop indefinitely.

NACK [041A], DC2 [041E], ACKN [0422] and DC1 [0426] are two-instruction function code entry points which load A with the code values to be output. All four end with an absolute jump (AJMP) to SPEOUT.

The SPEOUT time-out is implemented by a 50 cycle loop. In the loop, BIT7 (the Transmit Flag) is tested to see if the transmit port is ready to accept another character. The serial port is controlled by SERP (described above) which controls the BIT7 flag. The time-out period is about 380 microseconds; at a baud rate of 57.6 kilo-baud, this is about the unload time for two characters - a generous time margin. If the TX port does not become ready within this period, there is something wrong. In this event, the jump to SPEXIT re-initializes the processor, 8156 and 8755 chips. The re-initialization is done by an absolute call to COUL2 which is an entry point to INTIAL that was described above. In this initialization, the 8032 RAM memory is not cleared. The 50-count time-out loop provides protection from the effects of a power glitch which might perturb the state of SCON. If SCON is altered, the serial port modes would change and the TI flag could be inhibited which could hang up the processor in this loop indefinitely.

Note: An important aspect of this 50-count loop is that it is keyed to the 57.6 K-byte baud rate. If the baud rate is reduced the time-out count must be increased in proportion to the baud rate change. For example, if the baud rate is dropped to 19.2 K-baud, the time-out count must be 150 counts. If the baud rates are reduced below 19.2 K-baud, this counter must be changed from an 8-bit to a 16-bit counter to accomodate the larger loop count values.

SPEOUT [0428] is entered with the character to be output in A. Upon entry to SPEOUT, the Accumulator parity bit P (PSW 0) is loaded into the C bit (Carry bit, bit PSW 7). Remember that P will be 0 if A contains an even number of 1's. The function codes are transmitted with even parity so the serial port parity bit (TB8) is set to the state of P.

A is pushed onto the STACK for temporary storage and is loaded with a count of 50 for the loop count. At SPEO1, the JB BIT7,SPEO2 instruction tests the state of BIT7 - the serial port transmit ready flag (set by SERP). If the port is not ready, A is decremented and tested for a count of zero by the JZ SPEXIT instruction. If the count in A is not zero, the AJMP SPEO1 instruction returns control to the start of the loop. If upon entry to the loop or at some time within the 50 count period the transmit flag (BIT7) becomes a 1, the JB BIT7, SPEO2 instruction causes a jump to SPEO2 which pops the saved character off the STACK onto A. A is loaded into SBUF which immediately initiates the serial transmission on the XMT bus. After the transmission is initiated, the RET instruction returns control to the calling location which could be in either the CMD or MON routines. The logic for inducing the calls is described

in these routines below.

The algorithm for this set of routines is shown below. Note that most of the SPEXIT initialization is effected by a call to COUL1 in INTIAL (described above). These operations are repeated here to show the initialization actions.

```
BEGIN  NACK, DC2, ACKN, DC1, SPEOUT
        NACK:    CHAR EQ. 15H,  GOTO  SPEOUT
        DC2:     CHAR EQ. 12H,  GOTO  SPEOUT
        ACKN:    CHAR EQ. 06H,  GOTO  SPEOUT
        DC1:     CHAR EQ. 11H,  GOTO  SPEOUT
SPEOUT   SET CHAR EVEN PARITY BIT IN XMT PORT
SPEO1   IF PORT NOT READY  THEN  COUNT EQ. COUNT + 1
                IF COUNT EQ. 50  THEN  GOTO  SPEXIT
                ELSE  GOTO  SPEO1
        ELSE  SET XMT PORT NOT READY
        SET SBUF EQ. CHAR
END

BEGIN  SPEXIT
        CLEAR LOST FLAG
        DISABLE INTERRUPTS
        CALL COUL1
END

BEGIN  COUL1
        TURN OFF XMT, DISABLE DRIVER
        SET 8755 I/O PORTS TO OUTPUT
        READ ID BYTE
        SET BAUD RATE EQ. 57.6 KB
        SET SERIAL PORT TO MODE 3
        SET TIMER 1 TO MODE 2
        SET XMT READY FLAG
        CLEAR LED DISPLAY MEMORY
        SET BUFFER POINTER EQ. 60
        TURN ON TIMER 2
        SET SERIAL PORT INTERRUPT TO HIGHEST PRIORITY
        ENABLE INTERRUPT SYSTEM
END
```

In SPEOUT at 0430H, the instruction SPRO1: JB BIT7,SPEO2 is a conditional branch instruction in which the jump is taken if the designated bit is set. The Intel data book format is: 0010 0000 bit address rel address. The instruction op code is 20 07 05, in which the 20 designates the jump if bit set instruction, 07 designates the bit address in bit-addressible RAM and 05 is the relative address displacement. The bit address is 20.7 in the 8032 RAM (BIT7 REG 20.7 on page 3, defines the variable BIT7 as RAM bit 20.7, Data Definition area in the program listing). The jump destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. The address of the next instruction is 0433H. Adding 5 to this address produces an address of 0438H, the address of the symbolic label SPEO2 which is seen in the instruction mnemonics.

SPOOUT and SPEXIT

The SPOOUT [043F] routine performs the function of outputting monitor data to the Controller via the RCV bus. Logic in MON [01A6] (the monitor executor described below) does an absolute call (ACALL) to SPOOUT [043F] to output odd-parity monitor data to the Controller after it is acquired by the interface board. If the device is non-responsive, the DC2 function code (see DC2, SPEOUT [0428], etc. above) is called by MON to output this even-parity device no-response code. The monitor data and DC2 function code formats are described in Section 2.1.

If there is a serial port TX fault, a time-out loop in SPOOUT causes a jump to SPEXIT [0457] which does a mini-initialize and returns control to LOOP [0038], the entry point of the program control loop (described below).

The 16-bit monitor data value is formatted into two bytes. SPOOUT is called twice from MON1 [022F]; the first call outputs the MSB, the second the LSB.

The SPOOUT time-out is implemented by a 50 cycle loop. In the loop, BIT7 (the Transmit Flag) is tested to see if the transmit port is ready to accept another character. The serial port is controlled by SERP (described above) which controls the BIT7 flag. The time-out period is about 380 microseconds; at a baud rate of 57.6 kilo-baud, this is about the unload time for two characters - a generous time margin. If the TX port does not become ready within this period there is something wrong. In this event, the jump to SPEXIT re-initializes the processor, 8156 and 8755 chips. The re-initialization is done by an absolute call to COUL2 which is an entry point to INTIAL, described above. In this initialization, the 8032 RAM memory is not cleared. The 50-count time-out loop provides protection from the effects of a power glitch which might perturb the state of SCON. If SCON is altered, the serial port modes would change and the TI flag could be inhibited which could hang up the processor in this loop indefinitely.

Note: If the baud rate is reduced, the 50-cycle loop count must be increased in proportion to the baud rate change. See the note above in the SPEOUT description.

SPOOUT [043F] is entered with the data byte to be output in A. Upon entry to SPOOUT, the Accumulator parity bit P (PSW 0) is loaded into the C bit (Carry bit, bit PSW 7). Remember that P will be 1 if A contains an odd number of 1's. The data values (MOH, MOL) are transmitted with odd parity so the serial port parity bit (TB8) is set opposite to the state of P. The CPL C instruction complements the C bit to form the odd parity used for data values.

A is pushed onto the STACK for temporary storage and is loaded with a count of 50 for the loop count. At SPEO1, the JB BIT7,SPOO2 instruction tests the state of BIT7 - the serial port transmit ready flag (set by SERP). If the port is not ready, A is decremented and tested for a count of zero by the JZ SPEXIT instruction. If the count in A is not zero, the AJMP SPOO1 instruction returns control to the start of the loop. If upon entry to the loop, or at some time within the 50 count period, the transmit flag (BIT7) becomes a 1, the JB BIT7, SPOO2 instruction causes a jump to SPOO2 which pops the saved data value off the STACK onto A. A is loaded into SBUF which immediately initiates the serial transmission on the XMT bus. After the transmission is initiated, the RET instruction returns control to the calling location in the MON routine. The logic for inducing the calls is described in the MON routine described below.

In SPOOUT at location 044EH is the instruction AJMP SPOO1. This is an absolute jump to location SPOO1. The Intel data book AJMP format is: a10 a9 a8 0 0001 a7 a6 a5 a4 a3 a2 a1. AJMP transfers program execution to the indicated address, which is formed at run-time by concatenating the high order five bits of the PC (after incrementing the PC twice), opcode bits 7-5, and the second byte of the instruction. The destination must be in the same 2K block of program memory as the first byte of the instruction following AJMP. The instruction opcode is: 81 48 and the instruction is at address 044EH. The destination SPOO1 is at address 0448H. In this instruction, from the instruction definition, the destination address is: 100 0100 1000 or 0448H which is the address with the symbolic label SPOO1.

The algorithm for SPOOUT is shown on the next page. Note that most of the SPEXIT initialization is effected by a call to COUL1 in INTIAL (described above). These operations are repeated here to show the initialization actions.

```
BEGIN  SPOOUT
         SET DATA BYTE ODD PARITY BIT IN XMT PORT
SPOO1    IF PORT NOT READY  THEN  COUNT EQ. COUNT + 1
                 IF COUNT EQ. 50  THEN  GOTO SPEXIT
                 ELSE  GOTO  SPOO1
         ELSE  SET XMT PORT NOT READY
         SET SBUF EQ. CHAR
END

BEGIN  SPEXIT
         CLEAR LOST FLAG
         DISABLE INTERRUPTS
         CALL  COUL1
END

BEGIN  COUL1
         TURN OFF XMT, DISABLE DRIVER
         SET 8755 I/O PORTS TO OUTPUT
         READ ID BYTE
         SET BAUD RATE EQ. 57.6 KB
         SET SERIAL PORT TO MODE 3
         SET TIMER 1 TO MODE 2
         SET XMT READY FLAG
         CLEAR LED DISPLAY MEMORY
         SET BUFFER POINTER EQ. 60
         TURN ON TIMER 2
         SET SERIAL PORT INTERRUPT TO HIGHEST PRIORITY
         ENABLE INTERRUPT SYSTEM
END
```

XMT Bus Character Handling

The XMT bus character stream analysis routines must detect the SYNC character, decode the address and determine if it is within the assigned address block, distinguish between control and data request messages, load the message argument, analyze parity over the byte stream, etc. - all complex operations that sequentially decode the incoming data in conformance with the rules of the message format. This analysis is performed by the higher level LOOP, CMD, MON and STATUS executors (described below), which call intermediate level routines SPIN and SYNC to acquire the XMT bus data. SPIN and SYNC call READ, which polls the receive port RI flag to catch and store the XMT bus data on the fly. READ also clears the XMT buffer if it is full. The higher level routines CMD, MON and STATUS store address and argument values in RAM.

When other operations dominate CPU operations so that READ RI-polling is inhibited, the serial port interrupt subroutine SERP acquires and stores the data in the XMT data buffer. READ does not analyze parity; this is done by SYNC and SPIN.

At this point, the reader should review the serial port description in Section 2.2.

For a perspective on the serial character transmission and reception, consider the following digital logic operations in the serial port.

At the transmit and receive baud rate of 57.6 Kilobaud, the bit period is about 17.4 microseconds. This is about equivalent to 16 CPU machine cycles. The byte transmission period (including start bit, eight data bits, parity bit and stop bit) is 191 microseconds.

In Serial Port Mode 3, the transmit interrupt is generated by the serial port TX control logic at the leading edge of the stop bit transmit shift clock; this clock shifts out the stop bit. The stop bit is the last bit of the serial frame. Thus TI signals the firmware that the transmission has been almost

completed and the transmit portion of the serial port will shortly be available for a new transmission. The TI flag (in SCON) must be cleared by firmware. Note from Figure 9 (the serial port logic) that the transmit logic has a transmit buffer; the CPU can load the buffer and then do other tasks. After the byte has been output, the TI flag interrupts the CPU so that it can load the transmit buffer with the next character to be output.

In Serial Port Mode 3, the receive interrupt is generated by the serial port RX control logic halfway through the stop bit, about one CPU machine cycle earlier in the frame than the comparable TI interrupt described above. The RI flag is set at the leading edge of the last receive shift clock. This is about one bit-time earlier than the stop bit which conveys no information - the stop bit is a framing element. Note from Figure 9 that the receive logic is buffered, which provides the interrupt service routine time to read SBUF and store the data in the XMT data buffer (in RAM). The buffer is loaded by the LOAD SBUF signal which is generated by the same logic that generates RI so the data is immediately available to the interrupt service routine. Since SBUF is loaded at the same time as RI, the serial input shift register becomes immediately available for reception of the next character. The Mode 3 frame consists of 11 bits; this period is about 191 microseconds. SERP or READ operations must be completed within this period or data will be lost. The RI flag must be cleared by firmware.

SERP

SERP [0483] is the serial port interrupt subroutine that is called when the serial port interrupt is activated. This interrupt is set by either a Receive (RI) or Transmit (TI) flag (in SCON) and is generated by the serial port (see Figure 9 in Section 2.2). These SCON flags are set by the serial port and cleared by firmware. The serial port interrupt vector located at address 23H is an AJMP SERP - an absolute jump to the subroutine. INTIAL set the serial port interrupt to the highest priority and enabled the interrupt system. When a serial port interrupt occurs, the program counter is set to 23H and the AJMP instruction at that location is executed. The next higher interrupt vector address is 2BH, 8 locations higher. If the serial port interrupt subroutine had been seven instructions or less, it could have been executed in the 23H to 2BH sector.

SERP is a low-level subroutine that has two functions. In servicing Transmit Interrupts (TI), SERP clears the interrupt, sets a transmit interrupt flag (BIT7) and tests for the occurence of a pending Receive Interrupt (RI). If there is no RI pending, control is returned to the location at which the serial port interrupt occurred. Since the TI portion of SERP does not alter register contents, it does not push any registers onto the Stack.

As mentioned above, **SERP is not the only means of reading character data from the serial port.** The READ subroutine also reads data from SBUF (the serial port buffer register) by polling the RI flag. This is faster than the interrupt-driven SERP subroutine - preferable for the high-speed bus service function of the interface. If the device response time is long, or other processing operations dominate CPU operations, SERP may store XMT bus characters. The reader should compare the operations of SERP and READ.

A second inportant point is that **only SERP loads data into the XMT bus character buffer.** Since READ can capture and process the XMT bus character on the fly, there is no need for temporary storage of the character in the buffer.

In servicing Receive Interrupts, SERP tests the parity bit and loads the received character and parity state into eight-word XMT character and eight-word parity state buffers. If the buffers are full, the character buffer pointer BUFFN (address), is reset to the beginning of the buffer. Since some registers are altered during the execution of the receive portion of SERP, the Accumulator, R0 and PSW are pushed onto the stack. Before leaving SERP, the stack is popped to restore these three registers. Finally, the RI

flag is cleared before the interrupt return.

Note that BUFFN, the SERP XMT buffer pointer, is only updated in SERP. BUFFN is set to the address for the next character to be stored by SERP. The BUFFN value is compared with the BUFFC pointer value to determine if a new XMT bus character has been input.

The SERP algorithm is as follows.

```
BEGIN   SERP
        IF  TI  EQ. 1   THEN SET  BIT7,   CLEAR  TI
        ELSE     IF  NOT  RI  THEN   GOTO  SERP4
                 ELSE  PUSH PSW,  PUSH A,  PUSH R0,  SAVE CHAR IN BUFFN
                 IF  PARITY BIT  EQ. 0  THEN  BUFFN + 8  EQ. 00H
                 ELSE  BUFFN + 8 EQ. FFH,  POP  R0,  POP  A,  POP PSW,  CLEAR RI
                 IF  BUFFN  EQ.  68H  THEN  BUFFN  EQ.  60H
                 ELSE  CONTINUE
SERP4   INTERRUPT RETURN
END
```

Since SERP is serial port interrupt-driven, it is only entered if a transmit or recieve interrupt occurs. The first operation in SERP [0483] is a test of the TI flag. If set, the discrete BIT7 (transmit interrupt flag) is set, TI is cleared and control falls through to SERP1 which tests the state of the RI flag.

SERP1 [048A] tests the RI flag to see if it is the source of the interrupt. If the Controller is operating the XMT bus at high speed and the device response is slow, a receive interrupt is possible at about the same time as a transmit interrupt. Referring to Figure 9 in Section 2.2, the reader will note that the RI and TI flags are inputs to the OR gate which activates the vector to PC = 23H. There is no exclusion logic so either a TI or RI flag can induce an interrupt or simultaneous occurrance of both flags can induce the interrupt.

If RI is not the source of the interrupt (it must have been TI which was tested above), control is returned to the location at which the interrupt occurred by the JNB to SERP4, an RETI instruction.

If RI is the source of the interrupt, the PSW, A and R0 registers are pushed onto the STACK because they are used in servicing the interrupt. The listing shows 00H in place of R0; this is a quirk of the assembler. Because this R0 is in Bank 0, it has an address of 00H so the assembler indicates R0 in this obscure manner.

BUFFN is the character buffer pointer. It is loaded into R0 and the contents of SBUF (the character just read into the receive port) is saved in the buffer location pointed to by the contents of R0 (BUFFN).

Remembering that BUFFN in R0 is the current buffer address, the code next forms the address of the parity buffer. R0 and A are exchanged and 8 counts are added to A. (We don't know what was in A before the exchange and don't care; A will be loaded with its previous contents from the STACK before leaving SERP). The result of adding 8 to A is to form the value BUFFN + 8 in A. This is the location for storage of the character parity state. A and R0 are exchanged again so R0 can be used to load the parity state. A is cleared to 0 for the impending parity comparison.

RB8 is the receive parity bit in SCON; the JNB RB8, SERP2 instruction tests this bit. (Remember from Section 2.1, that data on the XMT bus can have either parity. The SYNC function code is transmitted with even parity, data with odd parity. The parity error evaluation occurs in SYNC [03B7] and SPIN [03E8]). If the parity bit RB8 is a 0, the jump is taken to SERP2 which stores the 00H value in A in the address pointed to by the contents of R0. This location is BUFFN + 8.

If RB8 is a 1, control falls through the jump to the CPL A instruction which sets A to the value FFH. At SERP2, this value is saved in location BUFFN + 8.

BUFFN is next incremented to prepare it for the storage of the next receive character. Since the XMT character buffer could overflow, it must be tested for this condition after the increment BUFFN. A is loaded with the value 68H which is the last character buffer address + 1. The BUFFN address value is tested by the CJNE A,BUFFN,SERP3 instruction which compares the overflow address in A with the incremented BUFFN value. If the incremented BUFFN value is less than 68H, the jump to SERP3 is executed. If BUFFN = 68H, BUFFN is reset to 60H.

At SERP3 [04AC], the RI flag is cleared and R0, A and PSW are popped off the STACK onto the respective registers. Note that the pops **must** be in inverse order to the pushes above. Having restored the registers, the RETI instruction returns control to the location at which it was interrupted by the TI or RI flags.

SERP terminates with an RETI, return from interrupt instruction, in this case returning from a response to the serial port interrupt. The Intel data book opcode format is: 0011 0010 or the 32 seen in the program listing. RETI pops the high- and low-order bytes off the PC sucessively from the STACK, and restores the interrupt logic to accept additional interrupts at the same priority level as the one just processed. The STACK pointer (SP) is decremented by two. No other registers are affected; the PSW is not automatically restored to its pre-interrupt status. Program execution continues at the resulting address which is generally the one immediately after the point at which the interrupt request was detected.

READ

READ [04E6] is a subroutine called by SPIN [03E8] (for serial port in) and SYNC [03D2] (for SYN character detection) to read a character from the serial port receive buffer SBUF, or from the XMT bus character buffer. BUFFN (next address) and BUFFC (current address) are the two XMT buffer address pointers. An important point is that **most XMT bus characters are acquired and processed by READ; SERP serves as a backup to READ in the event of heavy CPU activity.**

The interrupt-driven routine (SERP [0483]) described above detects the arrival of characters by the occurance of a serial port receive interrupt flag (RI). In responding to the interrupt, SERP stores the character in the XMT character buffer, increments the BUFFN buffer pointer, and if necessary, resets the buffer pointer. READ **principally** acquires XMT bus characters by polling the RI flag. This on-the-fly detection is much faster than the RI interrupt-driven SERP subroutine described above. The **secondary** character acquisition results from the operation of SERP. In the event of heavy demands for CPU time, READ can slip behind in polling the RI flag so that it must read SERP-loaded data from the XMT bus character buffer.

Since time is a dominant factor in XMT bus service, READ calls TIMER ([04B5] described below) which times-out apparent XMT bus inactivity. The assumption is made that the Controller will always transmit at least a few messages every 100 milliseconds. Therefore SYN function codes should occur at the same rate. TIMER has a 1-minute timer, initialized by SYNC when SYNC is called. In the event that a SYN function code is not detected within one minute, the best (and only possible) action is to re-initialize the interface by calling COUL1. This sets the appropriate states in the RAM and EPROM ports, the SFR's which control the baud rate generator (Timer 1) and the serial port. The disappearance of messages could be attributed to a power glitch-induced change to the state of these important control registers.

Since describing TIMER here would unnecessarily complicate the READ description, TIMER is described below.

READ is called from SYNC [03B7] by the higher level code LOOP [003E] when it is searching for the start of a new message. As a result of detection of a new message, SPIN [03E8] calls READ to acquire and store message data. The outputs of READ are the XMT character that is carried in A and the parity bit of the data byte carried in C. During the course of execution of READ, TIMER is called to detect bus time-out faults (mentioned above). READ continously loops while testing for the detection of an RI flag in SCON. READ also tests to see if SERP has put a character into the XMT bus character buffer. (This could have happened at some stage of READ when the serial port interrupt was enabled).

Important parameters in READ (as well as in SYNC and SERP) are BUFFN (new character) and BUFFC (current character). BUFFN is the XMT bus buffer address pointer set by SERP when it stores a character in the buffer. BUFFN is only incremented by SERP. BUFFC is another character buffer address pointer which is only incremented by READ. During the course of execution of READ, BUFFN and BUFFC are compared; if different, it means that SERP had stored a character when READ was unable to do so. If this is the case, READ obtains the character and parity state from the buffers and returns them to the calling routine just as if it had caught the characters on the fly. In the course of retrieving the character, BUFFC is incremented so that the BUFFC pointer tracks the BUFFN pointer; BUFFN is (hopefully) never more than one address higher than BUFFC upon entry to READ. If device logic response is very slow, BUFFC could be more than one address behind BUFFC. The maximum lag is six addresses; lags greater than this would probably cause serious interface malfunctions. This effect has not been measured.

The T2 overflow interrupt enable (TF2, T2CON.7) and T2 software start/stop bit (TR2, T2CON.2) may have been set by SPIN when it called READ.

The READ algorithm is as follows:

```
        BEGIN  READ
        DISABLE SERIAL PORT INTERRUPT
        IF  BUFFN EQ. BUFFC  THEN
READ1                              ENABLE INTERRUPT SYST,  TIMER 2 OVFLW FLAG,  SET TIMER 2 RUN
              IF  TIMER 2 OVERFLOW  THEN  CALL TIMER
              ELSE     IF  RI  EQ. 1  THEN  DISABLE TIMER 2 INTERRUPT, SET RI EQ 0,
                                            LOAD CHAR & PARITY, ENABLE SERIAL PORT,
                                            GO TO  LOST
                       ELSE  GO TO READ1
        ELSE  DISABLE TIMER 2 INTERRUPT, PUSH B, R0 ONTO STACK, READ CHAR FROM BUFFC,
              READ PAR FROM BUFFC + 8, INC BUFFC
        IF  BUFFC  EQ. 68H  THEN  BUFFC EQ.  60H
        ELSE  CONTINUE
        END
```

The first operation upon entry to READ is to disable the serial port which is done by CLR IEC.4. The next operation is a comparison of BUFFN and BUFFC; if equal, it means that SERP has not stored a new character in the XMT buffer. If unequal, control is transferred to READ2 which retrieves the character from the buffer.

If BUFFN and BUFFC are equal, the instruction ORL IEC,#A0H enables the interrupt system and and Timer 2; the serial port interrupt (IE.4) is left disabled. The Timer 2 interrupt is enabled because the 1-minute and 5-second timers in TIMER [04B5] must function during the operation of READ. TIMER is described below.

The next three instructions [04F0] are DB (define byte) instructions that make executable three sets of data which synthesize Timer 2 instructions. This awkward-looking construct was necessary because the AD2500 assembler version did not accomodate the 8032 microcontroller. The synthesized instructions are: ANL T2CON,84H [04F0], ORL T2CON,4 [04F3], and JNB T2CON.7,2 [04F6]. The ANL T2CON,84H is a direct address, logical AND of the contents of location C8H (T2CON) and the bit pattern

84H. This selects the TF2 bit (T2CON.7, Timer 2 overflow flag) and the TR2 bit (T2CON.2, Timer 2 start/stop bit; a 1 starts T2). If either of these two bits in T2CON are set, this instruction will set these bits into the Accumulator. The next instruction, ORL T2CON,4 will set the start/stop bit (to insure that T2 is started). The JNB T2CON.7,2 tests the T2 overflow bit and if not set, control is transferred to the RI flag polling loop. The state of the RI flag is tested by the JNB,READ1 and if not set, control is transferred to READ1 [04ED]. This repeats the loop until a T2 overflow is detected which calls TIMER by an absolute call (ACALL TIMER).

When the RI flag goes true in the loop above, it signifies that the serial port receive register (SBUF) has a new character to be read. The Timer 2 interrupt is disabled (we don't want a T2 interrupt at this time; the character acquisition process is running smoothly). The contents of SBUF are read into A and the RI flag is cleared so that it can be set upon reception of the next XMT bus character. The state of RB8 (the received character parity bit) is set into C (PSW.7, the Carry Flag). Control is transferred to READ5 by an absolute jump. At READ5 the serial port interrupt is enabled and a jump is taken to LOST to check for perturbations to the program execution resulting from power glitches. The operation of LOST is described below.

If on the entry to READ in the comparison of BUFFN and BUFFC it is determined that they are not equal, it indicates that SERP has put a character into the XMT character buffer which must be retrieved and processed. At READ2, the T2 overflow interrupt is disabled and the B and R0 registers are pushed onto the stack. (Remember the assembler shows the R0 address: 00H). The code then retrieves the character and associated parity state from the two buffers in a manner similar (but not identical) to that used in SERP. At the entry to READ, A was loaded with BUFFC, READ's buffer pointer. This address is put into R0 and an indirect read loads the XMT buffer character pointed to by R0 into B. The address of the associated parity state in the parity buffer is formed by adding 8 counts to A, which is then loaded into R0 to read the parity data (at READ3, below). BUFFC is incremented to prepare it for comparison with BUFFN on the next pass through READ.

After preparing R0 to read the parity state, BUFFC is compared with 68H to see if the end of the buffer has been reached (after being incremented as described above). If so BUFFC is reset to 60H. If not, at READ3 the parity state is read into A from the parity buffer. The C flag is cleared and the parity state (in A) is tested; if a zero, it signifies RB8 was zero and the JZ READ4 leaves C cleared. If the parity state is not zero (SERP sets the state to FFH for RB8 = 1), C is set. The character is loaded into A from B and the Stack is popped onto R0 and B.

At READ5 the serial port interrupt enable bit is OR-ed into IE (the Interrupt Enable register) and an AJMP (absolute jump) to LOST completes the operation of READ.

It should be noted that either the SYNC or the SPIN routine performs the first evaluation of the character and associated parity bit, READ simply passes these parameters to SYNC or SPIN without analysis.

In READ at address 04F0H, one of the curious-looking constructs mentioned above is: DB 53H,0C8H,84H. This synthesizes ANL T2CON,84H, a logical AND instruction for bytes. This is an ANL direct,#data instruction and the Intel data book opcode format is: 0101 0011 direct address immediate data. The first byte is the 53 opcode, the second is the direct address (0C8H) and 84H is the immediate data. C8H, the direct address, is T2CON in the SFR's. (See the SFR memory map in Section 2.3). In this case the instruction data is 84H or 1000 0100B which sets the desired state for T2CON. After execution all bits in T2CON will be a 0 except for bits T2CON.7 and T2CON.2, which will be a zero if they were a one before the instruction. T2CON.7 is TF2 (the overflow flag) and is permitted to be set and T2CON.2 is the TR2 software START/STOP bit for Timer 2.

SPIN

SPIN [03E8] (for serial port in) is an important subroutine called to obtain a character from the serial port XMT bus and to determine if the character was tainted by a parity error. Only single-bit parity errors are detected (but not corrected); the probability of multiple parity errors in a byte is vanishingly small under normal conditions. If multiple bit parity errors occur, they will be greatly outnumbered by single bit parity errors - an indication of a hardware problem such as a sick chip or a loose jiggling connection somewhere in the XMT bus path. Parity error occurrances are recorded at the calling location in the higher level code.

SPIN is called from a number of high-level program locations: LOOP [0048 and 0057], PARC1 [0087 and 008D], CMD [00B3 and 014C], MON [01DC, 01E4, 01AB and 02B3] and STATUS [02DB and 033F]. All these calls obtain message values required to decode the message in the context of the format stucture.

The SPIN algorithm is as follows:

```
BEGIN  SPIN
IF  BUFFN  EQ.  BUFFC   THEN   ENABLE TIMER 2 INTERRUPT
ELSE  CONTINUE
CALL  READ,  OBTAIN CHAR, PARITY
EXCLUSIVE  OF  PARITY AND  PSW.0
RETURN
END
```

To obtain the character, SPIN calls READ which returns with the character in A and the parity bit in C. C will be a 1 for RB8 = 1 and a 0 for RB8 = 0.

The parity bit of the XMT character obtained from SBUF is put into RB8 (in SCON). In evaluating the character parity, SPIN compares this bit with the P bit (PSW 0) which forms even parity over the contents of A. During the transmission over the XMT bus, either RB8 or the character could have been contaminated by an error. Since the XMT bus data is transmitted with odd parity and the P is formed over A to produce even parity, the state of RB8 is complemented before the analysis.

SPIN and READ have some complementary properties since SPIN alway calls READ. Only READ increments BUFFC; at the entry to SPIN, BUFFC is compared with BUFFN to determine if SERP has stored a character in the XMT buffer. In storing the character, SERP increments BUFFN after the character is stored. If BUFFN is not equal to BUFFC, BUFFN will be one address count higher than BUFFC. If they are equal, Timer 2's overflow interrupt is enabled and the timer is started by the ORL IEC,#A0H instruction. If unequal, Timer 2's overflow interrupt and Timer 2 start is deferred to READ as described above.

SPIN calls READ, and upon return, A contains the character and C contains the parity state, read from RB8. Remember that data is transmitted with odd parity and SPIN is only concerned with data. C is complemented for the impending comparison with P (PSW 0). In contrast, P is formed with even parity over A, i.e. when the number of 1's in A is an odd number, P is set to 1 so that the sum of 1's in A and P is always an even number. Thus the parity error analysis is based on even parity comparisons of P and the complemented RB8 bit, RB8-. During transmission either RB8 or the character could be contaminated by a single bit error, but the error probability for the character is (obviously) eight times that of RB8.

C (RB8-) is stored in BIT1, one of the arguments for the XORB (exclusive or) macro which will compare RB8 and P. P is set into C and then stored in BIT2, the other XORB operand. BIT0 contains the results of the XORB macro. Let the complement of RB8 be designated RB8'. The XORB macro performs the following logic product-sum: C = (RB8' x P-) + (RB8'- x P) where the - (minus sign) suffix

denotes the logic complement of the bit, x denotes a logic product and + denotes a logic sum. If RB8 is erroneous or A has a single bit error, C will be set indicating a parity error. The reader is urged to verify this logic operation by assuming values for TB8, RB8, the character, and P.

The XORB [03F9] macro instruction: XORB BIT1,BIT2,BIT0 causes the XORB Macro instructions (described below) to be injected into the instruction stream ([03F9 through 0405]. The XORB macro is described below so the macro operations will not be repeated here.

After completing the XORB macro instructions, control is returned to the calling locations (mentioned above) with the character value in A and the parity state in C.

In SPIN at 03EDH, the ORL IEC,#0A0H instruction performs a logical OR of the immediate data into the IE (Interrupt Enable) register. The instruction opcode is 43 A8 A0. The Intel data book shows this to be an ORL direct,#data instruction with the opcode format: 0100 0011 direct address immediate data. In this application, the 43 designates the type of ORL. A8 is the address of the IE register in the SFR memory bank and 0A0H is the immediate data.

SYNC

SYNC [03B7] is called from LOOP [003E] to detect the SYN function code which prefixes XMT bus messages. SYNC calls READ to obtain a bus character and the associated parity state. If the character value is 16H and the parity is even, the MSG flag is set indicating that a message has been detected. If the SYN character is not detected, program control remains in SYNC with periodic jumps to START to reinitialize the system.

SYNC must detect the SYN character for all messages on the XMT bus. The interface must assume that it is the target of every message until it has read ADH and ADL and compared the message address with the address block range.

SYNC is roughly analogous to SPIN but unlike SPIN, it does not return a value (the SYN function code is a discrete signal character, not a value). If SYNC is unable to detect the SYN character, it continues to loop until the character is detected. No other possible operations are possible until a message is detected. If the SYN character is detected but the parity is odd, an erroneous sync counter is incremented. Time is also a concern in SYNC. Although TIMER is not called, the one-minute timer paramenters in TIMER are initialized in SYNC upon entry.

SYNC also reads the ID code value by calling NUMBER in response to the BIT6 flag set by the operations of TIMER (discussed below).

On entry to SYNC, the 1-minute, 16-bit timer parameters TIME1 (LSB) and TIME2 (MSB) are initialized to 4CH and 04H, respectively. The composite argument is 44CH. These parameters are used by TIMER (described below) to periodically reinitialize the system. The assumption is that a power glitch could have perturbed the operation of the interface by spuriously setting discrete control flags in RAM. The operation of the 1-minute timer is described in TIMER.

SYNC1 [03BF] is the start of the SYN character test loop which begins by setting the 1-minute timer parameters. R1 (Register 1 in Bank 0) is reset to 0. R1 is used as an 8-pass character counter through unsucessful SYN character detection paths in SYNC.

Next follows the now-familiar equality comparison of BUFFN and BUFFC. This comparison is also used in SERP and READ. Remember that BUFFN is only incremented by SERP, and BUFFC is only incremented by READ when it has slipped out of the RI-flag polling mode. If they differ, it is assumed

that a character has been stored in the XMT character buffer by SERP. On the first entry to SYNC at the beginning of the message detection process, the only way that the interface could detect an XMT bus character is by the serial port interrupt which calls SERP. READ could not have been polling the RI flag since it would not have been called. There could be many passes through SYNC and READ before the SYN function code is detected and SERP could store the SYN code in the XMT buffer and increment BUFFN. If BUFFN is not equal to BUFFC, the CJNE A,BUFFN,SYNC2 instruction transfers control to SYNC2 to obtain the character.

If BUFFN and BUFFC are equal, the Timer 2 overflow interrupt (TF2 in TCON) is enabled and the timer is started (if it is not already running) by setting the TR2 bit (in TCON). The LOST (PSW.5) flag is cleared and if BIT6 is set, NUMBER is called to obtain the ID code value. (The reason NUMBER is called at this point is to update N for problem detection). Upon return from NUMBER and not having detected the SYN characater, control is returned to the start of the SYN character detection loop SYNC1 by an absolute jump.

At SYNC2 [03D0], the LOST flag (PSW.5) is cleared and an absolute call to READ obtains the character and associated parity state. Upon return from READ, A contains the character and C contains the parity bit: if C = 1, the character parity is even; if a 0, it is odd. The parity state is saved in BIT8, the Sync Carry flag which is tested at [03D9] (two instructions below). The CJNE A,#16H,SYNC3 instruction compares the value in A (returned by READ) with 16H. If the comparison is not equal, the character is not the SYN function code and the jump to SYNC3 is taken. If the character returned by READ matches the SYN function code, it must have even parity. The JNB BIT8,DSYNC instruction tests the parity state temporarily saved (above) in BIT8. If it is a 0, the parity is odd, an unacceptable condition for the start of a message. In this event, control is transferred to DSYNC [03AE].

If BIT 8 is a 1, the SYN function code has been detected and control falls through the JNB to first activate the LED state memory. This is done by setting the MSG bit (Port 1.1) and clearing it in the next instruction. This creates a clock which loads the LED display register (U3, 74LS173) with the states standing on the "D" inputs. In this case only the "MSG" LED will be illuminated. (See the Bus and Board Status Display description in Section 2.4). Having set the display, the RET instruction returns control to the calling location in LOOP [003E]. This sucessful detection exit permits the message decoding sequence in LOOP to proceed. See the description of this process in the LOOP description below. Note that none of the operations in SYNC produces an exit parameter; it is strictly a logical comparison subroutine.

At SYNC3 [03E1], the unsucessful comparison counter (register R1, it was initialized to 0 upon entry to SYNC) is incremented and is compared with the value 8. If the count is less than 8, control is transferred back to SYNC1 to repeat the character test loop. If the R1 count equals 8, an absolute jump to START returns control to the beginning of the program, which reinitializes the processor and processor peripheral chips by the absolute call to INTIAL. After initialization, SYNC will be called again to continue the search. Reinitialization after 8 SYNC1 cycles insures that correct program conditions are re-established in the event that some power glitch has perturbed RAM memory or the processor SFR's. For example, if the serial port mode control bits set in SCON or the Timer 1 (the baud rate generator) states were erroneous, it could be impossible to detect the SYN function code.

DSYNC [03AE] increments the "BAD SYNC" counter (invalid SYN character counter, BE-5) which is one of the interface board internal monitor and control functions described in Section 2.1. The counter is a 16-bit counter in RAM at locations 44H and 45H. Location 44H is the MSB and is assigned the mnemonic BADSY1; 45H is the LSB and is BADSY2. This counter is cleared during the RAM initialization in INTIAL. In the event of a continuous failure to detect the SYN function code, the count will never exceed 8 counts. The 8 count cycle is the result of the test of the state of R1 in SYNC4 below. If the Syn function code is occasionally missed, the BADSY counter could exceed 8 counts.

The "BAD SYNC" counting operation uses the INCRE subroutine [040F] (described below) which uses the contents of R0 as an index to the LSB, in this case 45H. INCRE just increments this counter and does not return a parameter to the calling location but makes an absolute jump to LOST [0587] where PSW.5 is tested for a perturbation to program execution as evidenced by the state of PSW.5. (See the description of the "LOST" strategy below). Upon return from LOST, R1 is incremented and an absolute jump is made to SYNC4 to compare the R1 count with the value 8 as described above in the SYNC3 paragraph.

The SYNC algorithm is as follows:

```
        BEGIN  SYNC
        TIME1  EQ. 4CH,  TIME2  EQ. 04H,   R1  EQ.  0
SYNC1   IF  BUFFN  EQ. BUFFC  THEN  ENABLE TIMER 2 INTERRUPT
                IF   ID REQ  EQ. 1   THEN  PSW.5  EQ. 0,  CALL NUMBER & READ ID VALUE
                ELSE   CONTINUE
SYNC2   ELSE  PSW.5  EQ 0,  CALL READ AND OBTAIN CHAR & PARITY
        IF  CHAR  EQ.  SYN  THEN
                                IF  PARITY  EQ. EVEN  THEN  SET BOARD STATUS IN DISP MEMORY,  RETURN
                                ELSE  INCREMENT  BAD SYNC COUNTER,  GO TO SYNC4
        ELSE  INCREMENT  R1
SYNC4   IF  R1  EQ. 8  THEN  GO TO START
        ELSE  GO TO SYNC1
        END
```

In SYNC at 03D2H, the instruction ACALL READ is an absolute call to the READ subroutine and has the opcode 91 E6. In the Intel data book the ACALL addr11 instruction has the opcode format: a10,a9,a8,1 0001 a7,a6,a5,a4,a2,a1,a0. This resembles the AJMP format discissed in SOOUT above. The ACALL unconditionally calls a subroutine located at the indicated address. The instruction increments the PC twice to obtain the address of the following instruction, then pushes the 16-bit result onto the STACK (low order byte first) and increments the STACK POINTER twice. The destination address is obtained by sucessively concatenating the five high-order bits of the incremented PC, opcode bits 7 to 5, and the second byte of the instruction. The subroutine called must start within the same 2K block of program memory as the first byte of the instruction following the ACALL. No flags are affected.

In this example, the ACALL READ instruction opcode is 91 E6. The numeric components which identify the ACALL instruction code are: aaa1 0001 aaaa aaaa and the address components are 100n nnnn 1110 0110. This address is 4E6H which is the address of the READ subroutine.

TIMER

TIMER [04B5] is a very important subroutine that uses Timer 2 as a clock to drive two firmware timers. TIME1-TIME2 is a 16-bit, 1-minute SYNC timer and TIME3 is an 8-bit, 5-second ID request flag timer. All three timing values are in 8032 RAM at addresses 56H, 57H and 58H. TIMER is called by the Timer 2 interrupt [002B] and by READ [04E6] when the Timer 2 Overflow flag TF2 goes true. The ID request flag timer periodically requests a re-read of the ID value by setting the ID request flag (BIT6). SYNC [03C7] tests this bit; if set, it reads and stores the ID value in NUMB1 and NUMB2. The 1-minute timer is used to cause a partial interface re-initializion by calling COUL1 [035D in INTIAL]. This is followed by a jump to LOOP [0038] to test for the detection of a new message on the XMT bus. Prior to the jump, the "LOST" flag, PSW.5 is cleared.

TIMER does not perform a time-out test of the XMT bus character interval. This period is solely under control of the Controller and must conform to the timing specified in Section 2.1.

The TIMER algorithm is as follows:

```
        BEGIN  TIMER
        PUSH PSW, A ONTO STACK
        IF  TIME1  EQ. 0  THEN  DECREMENT  TIME2
                IF  TIME2  EQ. 0  THEN
TIMER1                                      PSW EQ. 0,  CALL COUL2,  GO TO LOOP
                ELSE  CONTINUE
                IF  TIME2 - 5  LT.  0  THEN  GO TO  TIMER1
                ELSE  CONTINUE
        ELSE  DECREMENT  TIME3
                IF  TIME3  EQ.  0  THEN SET ID REQ FLAG
                ELSE  CONTINUE
        SET:  RCAP2L EQ. 0,  RCAP2H EQ. 0,  TIMER 2 TO RUN
        POP  PSW, A  OFF STACK
        INTERRUPT RETURN
        END
```

At this point the reader should review the operation of Timer 2 in Section 2.3. Timer 2 is clocked at a rate of 931,350 Hz (11.0592 Mhz/12) and operates in the Auto-Reload mode in which the 16-bit value in RCAP2L and RCAP2H is reloaded into TL2 and TH2 when TH2 overflows. The processor reset routine INTIAL [034C], SET2CON [04D8] routine and TIMER all clear RCAP2L and RCAP2H to 00H so that Timer 2 counts from 0 to 65,535. At overflow, RCAP2L and RCAP2H are reloaded into TL2 and TH2 for the next counting cycle. Overflow Flag TF2 (T2CON.7) sets the TIMER 2 interrupt. As mentioned above, READ may also call TIMER.

The TIMER entry rate is 14.211 Hz which is determined by the clock rate (931,350 Hz) and the Timer 2 radix (65536), 931,350/65536 = 14.211 Hz. This value pertains to both interrupt-driven and READ-driven entries. The 1-minute timer, TIME2 (MSB), and TIME2 (LSB) are initialized to a count of 044CH (1100 decimal) by SYNC [03B7]. This value in TIME1,2 is decremented on each pass through TIMER (at the 14.21 sec rate); when TIME1 and TIME2 equals zero, COUL1 is called to perform a partial re-initialization of the interface. In normal XMT bus operation, SYNC resets the 1-minute timer so that it never times out.

If the XMT bus message flow terminates, there may be a problem with the interface. It could have been perturbed by a power glitch which might make it impossible to detect XMT bus messages via SYNC. Examples of such are errors in the states of mode control functions such as the Timer 1 radix or mode or the serial port mode. The bus specifications require interface operation with maximum and minimum message rates of 955 us and 1145 us, respectively. The interface meets these specifications. In normal operation, the XMT bus is in constant use although (at the time this manual was written) there are no available values for average, maximum and minimum message rates.

The 1-minute timer period is: 1100/14.21 = 77.4 seconds.

A trap at TIMER2 [04CB] detects perturbations of the processor (possibly by power glitch effects) by testing the states of TIME2. TIME2 is the MSB of the 1-minute division value and is initialized to 04H by SYNC. As it is decremented to 00, it goes through the states 4,3,2,1 and 0; no other states are valid. The instruction SUBB A,#5 subtracts 5 from TIME2 and sets C if there is a borrow. Thus C is always set in the normal sequence. If TIME2 is any value other than the 0 to 4 operating range, C will be zero, indicating an erroneous state in TIME2. This condition is used to call COUL1 for the partial re-initialization.

In TIMER at 04B7H is the instruction PUSH A and the instruction opcode is C0 E0. The Intel data book shows only one type of push onto the STACK. In executing this instruction the STACK POINTER is incremented by one. The contents of the indicated variable is then copied into the internal RAM

location addresses by the STACK POINTER. Otherwise no flags are affected. In this case the Accumulator address in the SFR memory is E0.

ADDCK

ADDCK (address check) [045F] is called partway down LOOP [005A] to compare the just-acquired message address with the address block start address in BLOCK1,2 and the block end address, EBASE1,2 to determine if the interface is the target of the XMT bus message being processed. Remember that the address in EBASE1,2 is actually one count more than the last working address of the block. These address values are calculated by NUMBER [052E] after it reads the Block ID value from the device. The block size and block start address are provided by the Controller which loads them into addresses pointed to by the Block ID code. The NUMBER description details the operations of determining EBASE1,2.

In determining if the message address is within the bounds of the address block, two limit comparisons are necessary. If the message address is greater than or equal to the block start address (BLOCK1,2) and less than the block end address (EBASE1,2) then the interface is the target of the message being decoded. These limit comparisons are the first two operations of ADDCK. The third operation determines the Relative Address which will be set on the RA bus. ADDCK is called with ADH loaded into R3 and ADL loaded into R2. On return to the calling location (i.e. LOOP) from ADDCK, the Relative Address is contained in R1 (MSB) and R0 (LSB) and two discrete flags, BIT4 and BIT5 carry the result of the limit comparison. The most significant bit of ADH (the R/W- bit) is stripped off before ADDCK is called. Address parity is not analyzed in ADDCK; this function is performed by SPIN (described above).

ADDCK consists of two parts. The first part is a set of code with two sequential calls to the SUBT subroutine to perform the low and high limit comparisons. The second part (SUBT) is a 16-bit subtract subroutine called by the first part. The arguments for SUBT are in DPH and DPL. After calling SUBT to perform the two limit comparisons, the BLOCK1,2 (block start address) is loaded into DPL and DPH and control falls into SUBT [0479] for a third 16-bit subtraction. The result of the third entry to SUBT is the Relative Address, which is returned to LOOP in R1 (MSB) and R0 (LSB).

The 16-bit subtraction in SUBT consists of two 8-bit subtracts with borrow (SUBB); the borrow is set into C. C is cleared before the subtractions. The first (LSB) SUBB instruction subtracts DPL from A; the result is in A and if there is a borrow, C is a 1. The second (MSB) SUBB instruction subtracts DPH and C (from the first subtraction) from A and the results are in A and C. R2 contains ADL and R3 contains ADH for all three passes through SUBT. DPH and DPL are loaded with the two sets of block limit addresses before SUBT is called.

On the first pass through SUBT, DPL and DPH are loaded with BLOCK1 and BLOCK2 (respectively) and A is loaded with ADH and ADL (respectively) for the two subtractions. If the message address is greater than the Block Start address, C = 0, so the address is greater than or equal to the Block Start address. After this first pass through SUBT, C is saved in Bit4. The subtraction results in A are ignored on this first pass.

On the second pass through SUBT, DPL and DPH are loaded with EBASE1 and EBASE2 and A is loaded with ADL and ADH for the two subtractions. If the message address is less than the Block End address, C = 1, so the address is less than the block end address. After this second pass through SUBT, C is saved in BIT5. The subtraction results in A are ignored on this second pass.

On the third pass through SUBT, DPL and DPH are again loaded with BLOCK1 and BLOCK2. After completion of the two subtractions, R1 contains the MSB of the Relative Address and R0 contains the LSB of the Relative Address. C is not saved upon return to LOOP.

The ADDCK algorithm is as follows:

```
BEGIN   ADDCK
IF  MSG ADDR  GT. OR  EQ. BLOCK START ADDR   THEN
        IF  MSG ADDR  LT.  BLOCK END ADDR  THEN  MESSAGE IS WITHIN BLOCK,
                RELATIVE ADDRESS  EQ.  MSG ADDR - BLOCK START ADDR
        ELSE  CONTINUE
ELSE  MESSAGE OUTSIDE BLOCK
END
```

At 047BH in the SUBT subroutine of ADDCK is the instruction SUBB A,DPL with the opcode 95 82. This is a SUBB A,Rn instruction in which the contents of the register and C are subtracted from A and the results are left in A. The Intel data book shows this instruction opcode as 1001 1rrr where rrr is a register address. In this case the 95 8x identifies the SUBB instruction and rrr is the address of the register containing the operand. In the SFR memory bank the address of DPL is 82H.

SUBB subtracts the indicated variable and the carry flag together from the Accumulator, leaving the result in the Accumulator. SUBB sets the carry (borrow) flag if a borrow is needed for bit 7, and clears C otherwise. (If C was set before executing a SUBB instruction, this indicates that a borrow was needed for the previous step in a multiple precision subtraction, so the carry is subtracted from the Accumulator along with the source operand).

XORB Macro

The XORB [03F9] (exclusive or) macro is used to perform an exclusive or of two Boolean states. Strangely enough, the Boolean features of the MCS-51 series microcontrollers do not have a built-in exclusive or function so it was necessary to implement it in the form of a macro.

When XORB is invoked, the code operations of the macro definition (Page 1 on the program listing) is injected into the assembled code and executed in the normal manner. (See the macro code at [03F9 ... 0405] in SPIN). XORB is (only) called by SPIN [03E8] to perform an analysis of the parity states of the XMT bus character. This comparison uses the serial port receive parity bit (RB8) and the state of the Accumulator parity bit, PSW.0. The SYNC subroutine does not use the XORB function for parity analysis; it is done locally by analysis of the state of C after READ has been called to obtain the character. XORB is used frequently because SPIN invokes it for every XMT bus character it processes.

If the Boolean variables A and B are subjected to an exclusive or analysis, the operation is: C = A x B' + A' x B where C is the result, A' an B' are false ("0") states of the variables, x denotes the logic product and + denotes the logic sum. C is true (a "1" state) only if A x B' or A' x B is true. Note that C is false ("0" state) if A and B are both true or are both false. This is the algorithm implemented by the XORB macro. The XORB variables are set into 8032 RAM as BIT0, BIT1 and BIT2 where BIT1 and BIT2 are the two variables and BIT0 is the result. After the return from SPIN, the parity analysis results (still resident in C) are stored in BIT3 for subsequent logical decisions. BIT3 is not an XORB variable.

The XORB macro is passed two parameters in 8032 bit-addressable RAM: BIT1 and BIT2. The macro results are placed in BIT0. Two variables, MBIT1 and MBIT2 are defined to be the initial states of BIT1 and BIT2. MBIT1 and MBIT2 are complemented by the / assembly operator. For example, /MBIT1 is the logic complement of MBIT1. The macro operation is quite simple; in the first comparison, BIT1 is loaded into C and the ANL C,/MBIT2 forms the logic product of BIT1 (in C) and /MBIT2. The logic product result in C is saved in BIT0. In a similar manner, BIT2 is anded with /MBIT1 with the logic product results in C. The two results are logically OR-ed by the ORL C,BIT0 and the results are in C. Finally, C is loaded into BIT0 to complete the macro code operation.

In XORB at location 03FD is the instruction MOV C,BIT0 with the opcode 92 00. This is a MOV bit,C instruction in which the designated bit is loaded into C. The bit address is 20.0 in the 8032 RAM. In Page 3 in the Data Definition area of the program listing, BIT0 REG 20.0 defines the variable BIT0 as RAM bit 20.0.

INCRE

INCRE (for increment) [040F] is a simple subroutine called to increment a 16-bit variable. The last 16 addresses in the address block are dedicated to interface board and bus control-data functions. In this block there are 8 counters that accumulate various errors; these are described in Section 2.1. The INCRE subroutine is a convenient way to increment these counters with a minimum of in-line code. INCRE is called from: [0079, 009D, 0125, 0135, 01F2, 0214, 023B and 023C], all locations in which the XMT message is analyzed or device response evaluated.

The RAM locations that are incremented are 36H through 4DH; 48H and 49H (N, ID value storage) and 3DH, 3EH (command data of most recent message) are not changed by INCRE. The 16-bit counters are all ordered MSB-first and the MSB is designated by a 1 suffix, e.g. ADDPE1.

The address of the counter LSB is loaded into R0 before the call. R0 is used as an index register for the indirect access of the two counter locations. The location pointed to by the contents of R0 is read into A and 1 is added to it. The incremented value is saved again at the same address. Adding 1 to A may produce a carry (i.e. C = 1) if the original counter value is 255. R0 is incremented to point to the counter MSB and the counter value is read into A where it is incremented and saved at the same address. Finally, INCRE is exited by an absolute jump to LOST [0587], which returns control to the calling location.

Since INCRE is so simple, the algorithmic description is omitted.

In INCRE at address 0410H is the instruction ADD A,#1 with the opcode 24 01. This is an ADD A,#data instruction in which the the immediate data (the value 1) is added to the Accumulator. ADD adds the byte variable indicated to the Accumulator. The carry (C) and auxiliary carry (AC) are set, respectively if there is a carry-out from bit 7 or bit 3, and cleared otherwise. When adding unsigned integers, the carry flag indicates that an overflow occured.

OUTIT

OUTIT [0408] is a small subroutine called by an absolute call, ACALL, from two places [00EC and 00F6] in CMD and one place in MON [01D2]. Thus this subroutine reduces the program size by using one set of instructions in three places. CMD and MON are the command and monitor executors, (described below). OUTIT performs the simple function of outputting two bytes of data to either the 8156 or 8755 A and B ports. In the application of OUTIT for the 8755, Port A is loaded with RA, the relative address and Port B is loaded with control enables to activate the device-interface output control signals. In the application of OUTIT for the 8156, Ports A and B are loaded with the command state that drives the CON/MON bus. 8156 Port A is the LSB and Port B is the MSB.

OUTIT is called with the data values to be output in registers A (LSB) and B (MSB) and the port address in DPTR. The first byte is output by an indirect move to the address pointed to by the contents of DPTR (Port A). After the first byte output, DPTR is incremented to point to the second port address (the 8156 and 8755 port addresses are contiguous). A is loaded with the contents of B and the second byte is output to Port B by a second indirect move.

After outputting the two data (or control information) bytes, OUTIT exits to LOST by an absolute jump AJMP. After executing the operations in LOST, control is returned to the calling location by a RET instruction. The jump to LOST is to verify that the program has not been perturbed by a power glitch.

Since OUTIT is very simple, the algorithmic description has been omitted.

In OUTIT at location 0409H is the instruction INC DPTR with the opcode A3. This is an INC DPTR instruction in which a 16-bit increment (modulo 2^{16}) is performed; an overflow of the low-order byte of the DPL from 0FFH to 00H will increment the high-order byte (DPH). No flags are affected.

LOOP

Now that the XMT bus character service and other subroutines have been described, it is time to deal with the high-level message service routines such as LOOP [0038], PARCI [0073], STATUS [02C5], NOTUS [007B], CMD [00A1] and MON [01A6]. LOOP is the top-level routine which decodes and interprets an incoming message and branches to the mid-level routines listed above as a function of the address and the parity states of the message. These mid-level routines manage the various tasks which the interface must perform. For example, CMD acquires the command arguments and outputs it to the device logic; MON acquires the requested monitor data and outputs it to the Controller. STATUS configures the address block parameters as commanded by the Controller. PARCI responds to address parity errors. NOTUS handles cases in which the message destination is another interface. When these mid-level tasks have been completed, control is returned to LOOP to service the next message. In the longest path, LOOP is 30 instructions long.

In the process of decoding and interpreting the incoming messages, LOOP invokes many of the lower-level subroutines described above. For example, SYNC is called to detect the SYN function code which prefixes all XMT bus messages. After a message has been detected, SPIN is called on a character-by-character basis to read the message data values ADH, ADL, CDH, and CDL. ADDCK determines if the incoming message address is within the assigned address block. Other familiar routines are called as required by the message decode and interpretion logic of LOOP.

The LOOP algorithm is as follows:

```
BEGIN   LOOP
RESET STACK POINTER, SELECT BANK 0
CALL SYNC
CLEAR  BUSY, DOUT & PARX LED DISPLAY MEMORY
CLEAR  LOST
CALL  SPIN FOR ADH, SAVE PARITY ERROR IN BIT3
CLEAR  ADH  MSBIT
IF  ADH  LT.  255  THEN  GOTO  STATUS
ELSE  SAVE  ADH  IN  R3
CLEAR  LOST
CALL  SPIN FOR ADL,  PARITY ERROR IN BITO
SAVE ADL IN R2
IF  ADH HAS A PARITY ERROR  THEN  GOTO  PARCI1
        IF  ADL HAS A PARITY ERROR  THEN  GOTO  PARCI1
        ELSE  CONTINUE
ELSE  CONTINUE
IF  BLOCK1,2  EQ. OR GT. ADH,ADL  OR  LT.  EBASE1,2  THEN  ENABLE RCV BUS DRIVER, SET BUSY LED MEMORY
ELSE  GOTO  NOTUS
IF  MESSAGE IS  COMMAND  THEN  GOTO  CMD
ELSE  GOTO  MON
END
```

LOOP begins by initializing the STACK and bank registers. Although set up by INTIAL on power-up, from power-glitch considerations, it is best to re-establish these important registers at the outset. Having done this initialization, SYNC [03B7] is called to detect the arrival of a new message. Program control will remain in SYNC and the associated character-handling routines READ and TIMER until the SYN function code is detected. In the event that the SYN character is detected but is tainted by a parity error, control remains in the SYNC routine as described above. A parity-tainted SYNC function code is not accepted as the start of a message. The invalid SYN character counter (BE-5) is incremented when the SYN character has a parity error.

On return from SYNC (indicating that a new XMT bus message has been detected), the LED display memory bits BUSY, DOUT and PARX are set; the next clock on P1.1 will clock these states into the memory. (See the description of the Bus and Board Status Display Logic in Section 2.4). Next, the LOST flag, (PSW.5) is cleared. (See the LOST flag logic description above).

The top address byte ADH (and associated parity state) is acquired next by a call to SPIN. The top bit (the R/W-, MON/CMD- designator) is masked out of ADH. The parity state is saved in BIT3; it will be dealt with after ADL is acquired. If all other bits of ADH are zero, a Status command has been detected. Remember that the message address (ADH and ADL) is 16 bits. Also remember that message addresses under 100H are reserved for the Block Start (2N+1) and Block Size (2N) parameters for all interfaces in the system. See Section 2.6, the Interface Board Address Conventions and the address Space Map for a description of the usage of the N (Block ID value), 2N+1 and 2N parameters. In the event that all bits of ADH are zero (after bit 7 has been masked out), an absolute jump to STATUS is taken to either store or read out the 2N+1 and 2N values. STATUS operations are described below.

If ADH (with the R/W- bit masked out) is not zero, a device-related command or data request message to some interface has been detected. At this point LOOP has not determined if this interface is the target of the message. ADH (in A as it comes from SPIN) is saved in R3. (This point, label START2 [0054] is an entry point from STAT1 described below). PSW.5 (the LOST flag) is cleared and SPIN called to read ADL and the associated parity state which was set in BIT0 by the XORB in SPIN. ADL is saved in R2. At this point, both ADH and ADL are on hand, so ADDCK is called to determine if the message address falls within the address range assigned by the Controller by a STATUS command. (See the STATUS description below). ADDCK (described above) performs the comparisons and sets BIT4 or BIT5 if the address is out of range. In addition upon return, R0 and R1 contain the Relative Address, RA. (The digital logic usage of RA was described in Sections 2.4 and 2.5; the firmware usage of RA is described in CMD and MON below).

Upon return from ADDCK, a series of four comparisons is made. The first two test the parity of the stored parity states of ADH (in BIT3) and ADL (in BIT0). If either component is error-tainted, control is transferred to PARCI ([0073] below) to increment the address parity error counter. After this, control is returned to LOOP to search for the next message; the error-tainted message is rejected.

Two address range comparisons follow the address parity comparisons. These test the state of BIT4 and BIT5, set by ADDCK if the message address is out of range. Remember that ADDCK sets BIT4 and clears BIT5 if the address is outside the address range. If out of range, control is transferred to NOTUS [007B] below.

If the message address passes the above comparisons, the interface is the target of a valid, error-free (so far) control or data request message. Next, RCVEX (Port 1.7, not used, a vestige of a former special application) and RCVEN (Port 1.1, the enable for the RCV bus driver) are set. The interface can now return function codes (e.g., ACK, NAK, etc.) and data to the Controller on the RCV bus. The BUSY LED status bit (P 1.2) is cleared to illuminate the BUSY LED when clocked into the latch. This BUSY LED indicates that the interface is executing a control or data request message.

The next test is to distinguish between control and monitor data request messages. The raw ADH byte was saved in B (at [004C]) above. Bit 7 is the R/W bit; if a 0, the message is a monitor data request message and if a 1, it is a control message. (Note that this is the opposite sense of the R/W- device interface logic signal). If this bit is set, control is transferred to CMD [00A1]; if not set, control is transferred to MON by an absolute jump. All of the high-level message decisions have been made; after execution of the MON, CMD, STATUS, PARCI and NOTUS routines, control is returned to LOOP to service the next message.

In LOOP at address 0050H is the instruction JNZ START2 with the opcode 70 02. This is a Jump if Accumulator Not Zero (JNZ rel) instruction. From the Intel data book we see: If any bit of the Accumulator is a one, branch to the indicated address; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC after incrementing the PC twice. The Accumulator is not modified. No flags are affected. The opcode format is 0111 0000 rel address. In this case the relative address is 02. Incrementing the PC twice and adding 02 yields the address 054H, which is the address of the START2 instruction.

PARCI

LOOP [0038] branches to PARCI [0073] if ADH or ADL is tainted by parity errors. The LED display parity error memory bit is cleared (P1.4) when clocked (later). This lights the PARX display LED. The address (41H) of the address parity error counter (BE-7) is loaded into R0, the PSW.5 bit is cleared and INCRE (described above) is called to increment this 16-bit counter. Note that this counter is incremented for all messages, not just the messages addressed to this interface. Upon return from INCRE, control falls into NOTUS.

In PARCI the CLR PARX instruction, opcode C2 94 clears the port P1.4 latch. On Page 3 of the program listing, PARX REG P1.4 defines the symbol PARX to be P1.4. This is a CLR bit instruction and the opcode format is: 1100 0010 bit address. Port 1 is an SFR with an address of 90H. Port bit 4 has the bit address 94H.

NOTUS

NOTUS [007B] is called when LOOP (using ADDCK) determines that the interface is not the target of the message. As described above, control also falls into NOTUS if the message address components have a parity error. NOTUS is also called from STAT4A [02F1] when the 2N +1 comparison shows that another interface is being addressed for a STATUS commnd (see STATUS below). NOTUS is used to record parity errors and reset the bus enable before returning to LOOP. NOTUS acquires (but does not save) the message argument CDH and CDL, analyzes parity over the argument, increments the command argument parity error counter (BE-6), disconnects the RCV bus driver and returns control to LOOP.

Upon entry to NOTUS, a 40 count delay loop consumes about 87 usec, about 5 serial bit periods. The RCV bus may have been made active in processing a previous message. CMD, MON and STATUS make the RCV bus stay active for 87 microseconds after the bus transmissions have been completed. Since this interface will not use the RCV bus (some other interface might need to use it). NOTUS provides a turn-off after the small delay. CLR RCVEN and CLR RCVEX disconnects the RCV bus driver and clears P1.7.

The LOST flag (PSW.5) is cleared and SPIN is called to read CDH; the parity state is saved in BIT3 and the LOST flag is cleared again. Another call to SPIN reads CDL and the associated parity state which remains in BIT0 (from the XORB macro in SPIN). In SPIN, CDL over-writes CDH in A, both arguments are of no interest now but the CDH, CDL parity must be checked.

Upon return from the second SPIN, the states of BIT3 and BIT0 (parity errors for CDH and CDL) are tested by the JB BIT0,PARCI1, JB BIT3,PARCI1 instructions. If either bit is set, the jump to PARCI1 is taken and the PARX output (P1.4) is driven low; the state will be clocked into the memory latches later. 43H, the address of the command parity error counter (BE-6), is loaded into R0, LOST is cleared again and INCRE [040F] is called to increment the counter. Upon return from INCRE, an absolute jump to LOOP starts the search for the next message.

Since PARCI and NOTUS are so closely linked, the joint PARCI-NOTUS algorithm is shown below:

```
BEGIN  PARCI-NOTUS
SET  PARX LED MEMORY
CLEAR  LOST
INCREMENT ADDRESS PARITY ERROR COUNTER,  BE-7
DELAY  87 USEC
DISCONNECT  RCV  BUS DRIVER
CLEAR  LOST
CALL  SPIN  FOR  CDH,  SAVE PARITY ERROR IN BIT3
CLEAR  LOST
CALL  SPIN  FOR  CDL,  PARITY ERROR IN BIT0
IF  PARITY ERROR IN  CDH,CDL  THEN  SET  PARX  LED MEMORY,
                        INCREMENT CDH-CDL PARITY ERROR COUNTER, BE-6,
                        GOTO  LOOP
ELSE  GOTO  LOOP
END
```

In NOTUS at 007EH is the instruction DJNZ B,$ with an opcode of D5 F0 FD. This is a Decrement and Jump if Not Zero instruction. The Intel data book shows this to be a DJNZ direct,rel instruction with an opcode format of: 1101 0101 direct address relative address. DJNZ decrements the location (register B) by 1, and branches to the address indicated by the second operand if the resulting value is not zero. An original value 00H will overflow to 0FFH. No flags are affected. The branch destination is computed by adding the signed relative-displacement value in the last instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. In this case the address of the B register (in the SFR's) is F0H. The $ is an assembler code designating the address of the instruction 007EH. The relative displacement is FDH, a negative value of 3. The instruction is at 007EH and the address of the first byte of the next instruction is 81H. Adding minus 3 to 81H yields 007EH, the instruction address.

STATUS

STATUS [02C5] is entered by an absolute jump from LOOP [0052] when the message address is less than 100H (255 decimal). The address range of 0 to FFH is assigned to storage of interface address block start addresses and address block sizes. (See Section 2.5 for a description of the address conventions).

STATUS [02C5] is the routine which deals with the address block parameters N (device ID value), 2N (the vector to the address block size parameter, COUNT1,2) and 2N + 1, the vector to the address block start parameter, BLOCK1,2. On power-up, the interface reads an ID value N from the device logic; numerical manipulations of N provide the address vectors for the storage of the two parameters. The value of N is unique to each device and is a hard-wired device function but the values pointed to by 2N and 2N + 1 are set into the interface by control messages from the Controller. The Controller can reassign these parameters at any time and STATUS controls the loading of these parameters into the two sets of RAM addresses.

The STATUS algorithm is as follows:

```
        BEGIN  STATUS
        R3  EQ.  ADH
        IF  ADH  HAS A PARITY ERROR  THEN  GOTO  START2
        ELSE  CALL NUMBER, READ  N_DEV
        IF  N_DEV  HAS A PARITY ERROR  THEN  GOTO START2
        ELSE  CLEAR  N_DEV  PARITY BIT,  2N_DEV  EQ.  2*N_DEV,  PUSH B,  B EQ. 2N_DEV
        CLR  LOST,  CALL SPIN  FOR  ADL,  SPIN RETURN:  A  EQ. ADL,  R2 EQ.  ADL
        IF  ADL  HAS A PARITY ERROR  THEN  GOTO PARCI
        ELSE  CONTINUE
STAT5   IF  A  EQ.  B  THEN  SET RCVEX,  SET RCVEN,  B  EQ.  ADH
                IF  ADH  MSBIT  EQ. 1  THEN  GOTO  STCMD
                ELSE  GOTO  STMON
        ELSE  B  EQ.  B + 1
        IF  A  EQ. B  THEN  GOTO  STAT5
        ELSE  POP B,  GOTO  NOTUS

STMON   IF  A.0  EQ. 0  THEN  R3 EQ. COUNT1,  R2 EQ. COUNT2,  GOTO  MON4
        ELSE  R3  EQ. BLOCK1,  R2  EQ. BLOCK2,  GOTO  MON4

STCMD   IF  A EQ. 0  THEN  R0  EQ.  COUNT1 ADDRESS
                CALL  STCMD5
                STCMD RETURN:  R5 EQ. CDL, BIT9 EQ. 0 IF CDH  EQ. 0
                COUNT1  EQ. R5,  INCR  R0,  COUNT2  EQ. R4
                CLR  LOST,  CALL NUM1,  NUM1 RETURN: NO CALCULATION OF EBASE OR IBASE
                GOTO  CMD3
        ELSE    R0  EQ.  BLOCK1 ADDRESS
                CALL  STCMD5
                STCMD5 RETURN:  R5  EQ. CDH,  R4 EQ. CDL,  BIT9 EQ. 0  IF CDQ  EQ. 0
                BLOCK1  EQ. R5,  INCR  R0,  BLOCK2  EQ. R4
        IF  BIT9  EQ. 1  THEN  GOTO  CMD3
        ELSE  CONTINUE
        IF  CDH  GT. 80H  THEN  GOTO  CMD3
        ELSE  R0  EQ.  BLOCK1 ADDRESS
                BLOCK1  EQ.  R5,  BLOCK2  EQ.  R4
                CLR  LOST
                CALL  NUM1
                NUM1 RETURN: EBASE, IBASE CALCULATED
                GOTO  CMD3
        END

STCMD5  CLR LOST
        CALL SPIN
        SPIN RETURN:  A EQ. CDH,  C EQ. PARITY ERROR
        R5  EQ. CDH, BIT3  EQ. C
        BIT9  EQ. 0
        IF  CDH  EQ. 0  THEN  BIT9  EQ. 1
        ELSE  CONTINUE
        CALL  ACKN,  ACKN RETURN
        CALL  SPIN
        SPIN RETURN:  A EQ. CDL,  C EQ. PARITY ERROR
        IF  CDH  OR  CDL  PARITY ERROR  THEN  POP B, GOTO  CMD1
        ELSE  R4 EQ. CDL
        RETURN TO CALLING LOCATION + 1
```

N (the device ID value) is read by NUMBER [052E] when called by INTIAL [034C] during power-up intialization or when the program is re-initialized by a jump to START [0000]. N is also periodically re-read by SYNC [03B7] when the ID request flag (BIT6) is set by TIMER3 [04D0] in the TIMER [04B5] subroutine. Although N is acquired quickly in the power-up initialization, the interface cannot service device-related control or data request messages until the block size and block start address are supplied by the Controller. Until this is done, the interface has no relevant device address information. INTIAL assigns a default block start address of 7FF0H and a block size of 10H. No device address is permitted

to exceed 7FEFH; the default and device addresses are disjoint.

The Controller can read the value of N (in BE-3) by a monitor data request to this address but cannot alter N. The readout of N from BE-3 is a response to a conventional monitor data request and is handled by MON.

The Controller can also read the 2N (COUNT1,2) and 2N + 1 (BLOCK1,2) parameters by monitor data request messages using 2N and 2N + 1 as addresses. STATUS handles the readout of these parameters.

The Controller first outputs the block size control message using 2N as the address; the message argument, CDH, CDL is COUNT1,2. Some time later the Controller outputs the block start address control message using 2N + 1 as the address; the message argument in CDH, CDL is BLOCK1,2. Other than this content and order, there are no other specifications for these two control messages. For example, the arrival time or interval between the two messages is not specified and the messages are not required to be consecutive. STATUS will not permit device control or monitoring until **both** parameters have been loaded. In addition, the Controller does not output control messages or monitor request messages with default block addresses. As a result of this address assignment scheme, STATUS is a rather complicated routine - much more so than would have been the case if the interface could read a simple block start address from the device.

As described in NUMBER [052E] above, using the BLOCK1,2 and COUNT1,2 values, NUMBER calculates EBASE1,2 and IBASE1,2. If the Controller has not output the two values, NUMBER uses the default values (BLOCK1,2 = 7FF0H, COUNT1,2 = 10H) in these calculations. Arithmetically, an address value of 8000H is acceptable (using the default settings) for EBASE1,2 but any value greater than 8000H is rejected by NUMBER because no address can exceed this value. If NUMBER calculates an EBASE1,2 greater than 8000H, the assumption is made that the BLOCK and/or COUNT values have been contaminated, perhaps by a power glitch or the address range has been only partially set up.

An important consequence of the use of Controller-assigned address parameters is that a jump to START will re-initialize the interface and INTIAL will set the default values 7FF0H and 10H into BLOCK1,2 and COUNT1,2. If the interface is re-initialized, the Controller must recognize the absence of acknowledgement function codes, lack of device control and absence of device monitor data. If noticed, the Controller can verify the re-initialized condition by reading out the two parameters using data request messages addressed to 2N + 1 and 2N. When the Controller recognizes the re-initialized condition of an interface, it must re-transmit new block start and block size messages to the interface. There are two jumps to START in the firmware: the first is in SYNC [03B7], the second is in LOST [0587]).

STATUS is called when LOOP [at 0052] has determined that the message address is less than 100H (i.e., ADH = 0) but before ADH parity has been tested, ADL read or the call to ADDCK. Since the address is less than 100H, ADL values are either 2N or 2N + 1 values for **some** interface; at this point, it is not clear that the message is for this interface. The logic below will relate the device N to the message 2N or 2N + 1 values to determine if the message is targeted to this interface. Before the jump to STATUS, B was loaded with ADH which is used below in STAT5.

ADH is also saved in R3 and BIT3 is tested for a parity error. If there is a parity error, control is returned to START2 [0054] in LOOP, which is the instruction that followed the jump to STATUS. The reason for the return to START2 is that the subsequent operations in LOOP will increment the address parity error count by the jump to PARCI [0073].

At STAT1 [02CB], if ADH is not tainted by a parity error, the device ID value N is read from NUMB2, the LSB of the 16-bit value read from the device by NUMBER [052E]. This is a 7-bit value with the MSbit set to odd parity by the device logic. The parity is evaluated by testing the state of P (PSW.0). P is even parity over the 8 bits of A. That is, if the number of 1's in A is odd, P will be 1 to make the total set of 1's an even number. In the case of the 7-bit + odd parity value for N, P will be 0 if there is no parity error in N. If there is a parity error in N, control is transferred to START2 just as in the case of a parity error in ADH just above.

The parity bit is masked off the device N value and N is multiplied by two to obtain $2N_{DEV}$. The contents of B are pushed onto the STACK to free B for comparisons and the $2N_{DEV}$ value is set in B for the forthcoming comparison. The LOST flag is cleared and SPIN [03E8] is called to read ADL which will be either $2N_{CNTRLR}$ or $2N + 1_{CNTRLR}$. Upon return from SPIN, A will have either of these two values and BIT0 will be 1 if there is a parity error. The value in A is saved in R2 for subsequent use and BIT0 is tested for a parity error. A will continue to hold the ADL value read by SPIN for the forthcoming comparison (described below). If there is no parity error, JNB BIT0,STAT3 will transfer control to STAT3. If there is an error, B is popped off the STACK and an AJMP to PARCI will record the error and return control to LOOP; an erroneous value cannot be used for this vital function.

At STAT3 [02E5], if the $2N_{CNTRLR}$ (it could also be the $2N + 1_{CNTRLR}$ value or values for other interfaces) in B is equal to the $2N_{DEV}$ value in A (from above), the message is addressed to this interface but it is not yet clear whether the message is a control message with a $2N_{CNTRLR}$ value in CDH and CDL or a data request message to read the contents of the $2N_{DEV}$ address stored in RAM. If the two sets of 2N values match, the AJMP to STAT5 will start the action of responding to the message. If the 2N values compared above do not match, the $2N_{DEV}$ value in B is incremented to form $2N + 1_{DEV}$ which is compared with the ADL value in A. (Remember that this could be addressed to any interface so the firmware must also test this 2N + 1 parameter to see if it is a message targeted to this interface). If the 2N + 1 values match, the AJMP to STAT5 is taken to start the action of responding to the message.

At STAT5 [02F5], the RCV bus is enabled and the RCVEX line (P1.7, no longer used) is set high. The STACK is popped to put ADH (B held ADH before the jump to STATUS; it was pushed above) back into B. The JB.7,STCMD instruction tests the ADH MS bit to determine if the message is a control message or a data request message for either of the 2N or 2N + 1 parameters. If the bit is a 1, the message is a control message to store CDH, CDL. CDH and CDL holds either the COUNT1,2 value (pointed to by $2N_{CNTRLR}$) or the BLOCK1,2 address (pointed to by $2N + 1_{CNTRLR}$) in RAM. At this point the interface has not yet obtained CDH and CDL from the bus; this will be done in STCMD5 below.

Upon entry to STCMD [030D], A holds either $2N_{CNTRLR}$ or $2N + 1_{CNTRLR}$. The LSbit of A is tested by the JB A.0,STCMD4 instruction. If the A.0 bit is a 1, the contents of A is $2N+1_{CNTRLR}$, CDH and CDL will hold BLOCK1,2 and the jump to STCMD4 is taken. (See STCMD4 below).

If the A.0 bit is a 0, the contents of A is $2N_{CNTRLR}$, CDH and CDL will hold COUNT1,2, and the jump is not taken. The COUNT1,2 value in CDH, CDL will be stored in RAM in addresses 51H and 50H. 50H, the RAM address for storage of COUNT1 (the MSB of COUNT), is loaded into R0 and the STCMD5 subroutine is called to read and parity test CDH and CDL. After returning from STCMD5, the CDH, CDL will be stored in the COUNT1, COUNT2 RAM addresses.

STCMD4 [031F] is peculiar to the block start address parameter pointed to by the 2N + 1 vector. Upon entry to STCMD4 (from STCMD above), R0 is loaded with 4EH (the BLOCK1 RAM address) and the STCMD5 subroutine is called to read and parity test CDH and CDL. After returning from STCMD5, CDH and CDL will be stored in R5 and R4, respectively. The parity errors will be in BIT3 (CDH) and BIT0 (CDL) and an under-range address error flag in BIT9. The under-range address error test is performed in STCMD5, (described below).

In STCMD4, immediately upon returning from STCMD5, the under-range address flag BIT9 is tested by the JNB BIT9,STCMD3 instruction. If BIT9 is 0, the CDH block address is less than 100H (determined by STCMD5); this is an unacceptable address because all device addresses are greater than 100H. If BIT9 is a 1, control passes on to the upper limit test.

In STCMD4, a second upper limit test is performed on CDH read into A from R5. C is cleared and then 80H is subtracted from A. If CDH < 80H, C = 1 so CDH is less than 80H, an acceptable address. In this case, control falls through the JNC STCMD3 instruction and the AJMP STCMD2 is taken. (STCMD2 will store the BLOCK1,2 address in RAM). If C = 0, CDH exceeds 80H; clearly the CDH value is in error so the JNC STCMD3 instruction will return control to CMD3, a cleanup area described in CMD.

STCMD5 [032E] is a subroutine called to read CDH and CDL and to evaluate parity for the two bytes. Upon return to the calling locations, R5 will hold CDH and BIT3 will be set if there is a CDH parity error. R4 will hold CDL and BIT3 will be set if there is a CDL parity error. STCMD5 also tests CDH to see if the value = 0. If so, BIT9 is set. (This test is only relevant to the BLOCK1,2 address; BIT9 is ignored in evaluating the COUNT1,2 value).

Upon entry to STCMD5, the LOST flag is cleared and SPIN is called to read CDH. CDH is stored in R5 and the parity error in BIT3. BIT9 is cleared before the zero test of the JZ STCMD6 instruction. Upon return from SPIN if CDH (in A) is 0, the address is an error. CDH should be greater than 0 since all device addresses exceed 100H. The JZ STCMD6 instruction bypasses the next instruction which sets BIT9.

At STCMD6 [033B], the ACKN [0422] subroutine is called to transmit the ACK function code to the Controller on the RCV bus. This subroutine and the associated delay routine is described above.

Upon return from ACK/SPEOUT [0422, 0428], the LOST bit is cleared and SPIN is called to obtain CDL and the associated parity bit which is stored in BIT0 by SPIN and remains in C. If CDH has a parity error the JB BIT3,STCMD instruction pops the STACK onto B and control is transferred to CMD1 which clears the PARX LED port (P1.4) so that the PARX LED will be illuminated when the memory is clocked. This error exit leaves without storing the COUNT1,2 or BLOCK1,2 parameters.

If there are no parity errors in CDH and CDL, the JNC STCMD8 instruction tests the state of C which would have been set by XORB if CDL had a parity error. If C is not set, CDL is set in R4 and the RET instruction returns control to the calling locations in STCMD or STCMD4. Remember that STATUS calls STCMD5 twice: once to read BLOCK1,2 and the second time to read COUNT1,2.

Having described the functions of identifying the $2N_{CNTRLR}$ and $2N + 1_{CNTRLR}$ control messages and testing them for parity and address limits, the final step in executing the command is to store the two address block parameters in RAM memory and restructure the address parameters EBASE1,2 and IBASE1,2.

The return from the STCMD5 subroutine will be (for both paths) to STCMD2 [0314], which is a pair of indirect MOV instructions using the contents of R0 as the index. In storing the BLOCK1,2 addresses, the BLOCK1 address 4EH was loaded into R0 in STCMD4. At STCMD2 [0314] the MOV @R0,05 instruction stores CDH from R5 into RAM location 4EH. R0 is incremented and the next instruction stores CDL from R4 in location 4FH. Note the register references to R5 and R4 by the assembler, a quirk noted earlier in referencing R0.

The storage of the COUNT1,2 parameters also uses STCMD2 but in this case 50H, the address of the COUNT1 parameter, was loaded into R0 in STCMD. In executing the STCMD2 instructions with 50H

loaded into R0, the two count parameters are loaded into RAM addresses 50H and 51H in an identical manner to that described immediately above.

After each pass through STCMD2, the LOST bit is cleared and NUM1 [0559] in NUMBER is called to calculate EBASE1,2 and IBASE1,2 as described above in the NUMBER description. Remember that the Controller outputs the 2N (COUNT1,2) value first so the first pass through NUM1 will fail because the address block start is the default value 7FF0H. The failure will cause a bypass of the storage of the calculated values. On the second pass, after reception of the 2N +1 value for BLOCK1,2, the calculated values of EBASE1,2 and IBASE1,2 are set into RAM storage which will enable the interface to execute device control and data request messages. This completes the control message operations of STATUS.

The other case of STATUS operations is the response to data request messages which have the now familiar 2N and 2N + 1 addresses. Remember that at [02FB] bit B.7 of ADH was tested. If a 1, the message is a command message and control was transferred to STCMD as described above. If bit 7 is a 0, control falls through to STMON [02FE] where bit A.0 is tested to see if it is odd or even (just as it was done in interpreting the 2N or 2N + 1 commands above). If odd, ADL is a 2N + 1 value and control falls through the JB A.0, STMON1 instruction to load the COUNT1 value (from RAM) into R3 and the COUNT2 value into R2. If ADL is a 2N value, the jump is taken and the BLOCK1 and BLOCK2 values are loaded into R3 and R2. Both paths exit to the MON4 [02A6] entry point in MON [01A6], which outputs the values to the Controller.

In STATUS at address 02E1H is the instruction POP B with an opcode of D0 F0. This is a POP direct (from STACK) instruction. The contents of the internal RAM location addressed by the STACK POINTER is read, and the STACK POINTER is decremented by one. The value read is then transferred to the directly-addressed byte indicated. No flags are affected. In the Intel data book the opcode format is: 1101 0000 direct address. The B register is an SFR with an address of F0H, so the STACK value in RAM (pointed to by the STACK POINTER) is loaded into B by this instruction.

CMD

CMD [00A1] is the routine that outputs the command argument of a control message to the device; this is called a device command below. Control messages also set command arguments (CDH, CDL) into certain locations in the interface RAM memory; these are called internal commands below. Internal commands do not interact with the device; they are used to set values into most of the upper 16 internal addresses (see the exceptions noted in "Interface Internal Monitor and Control Functions", Section 2.1 and 2.5). Two other internal commands are directed to the 2N and 2N + 1 addresses. These two commands were discussed at length in STATUS [02C5] above.

In executing the CMD routine for device commands, the interface activates Relative Address and interactive control signals via the 8755 and 8156 I/O ports. The characteristics of these I/O ports were described in Section 2.2. The interface tests the response of the device to the interface signals; these stimulus and response signals were described in Sections 2.3 and 2.4. A 500 usec timer terminates the command in the event of device non-response; the non-response condition is recorded in BE-12. The command address and argument for all commands are also recorded in BE-9 and BE-8 respectively. The command arguments CDH,CDL are parity tested and if found erroneous, the command is aborted. At certain stages in the course of executing the command, control function codes are returned to the Controller. The function codes returned depend upon the response of the device and quality of signals received by the interface on the XMT bus. These function codes are described under "Function Codes" in Section 2.1.

CMD is composed of mostly simple in-line code; interacting with the device logic and I/O ports requires a lot of sequential states. CMD also uses a number of calls to lower level subroutines such as

SPIN [03E8], SUBT [0479], OUTIT [0408], INCRE [040F] and the function code drivers ACKN [0422], NACK [041A], etc. In spite of the bulk, CMD is relatively simple because there are only a few simple logic decisions to be made.

CMD is entered from LOOP [006E] (described above) after LOOP has determined that the control message is within the interface address block and the address (ADH,ADL), is not tainted by parity errors. The decision to branch to CMD is based upon the presence of a 1 in the most significant bit of ADH. If this bit is a 0, the MON [01A6] branch is taken.

CMD operations begin by an analysis of the Relative Address to determine whether the message is a device control command or an internal command. In Loop (at [005A]), ADDCK was called to determine whether RA is greater than IBASE1,2. Upon return, R0 contained the Relative Address LSB and R1 the MSB. C was set if IBASE1,2 was greater than RA - the condition for a device command. The JC CMD0 transfers control to CMD0 [00B1] where device command operations begin. Remember that IBASE1,2 is the first address of the last 16 addresses in the address block that are dedicated to the interface internal monitor and control functions. In the ADDCK comparison mentioned above, C would have been 0 for RA > IBASE1,2.

A side issue of the address comparison in ADDCK is the resultant 16-bit RA. The 16-bit value for RA from ADDCK may seem discordant when we remember that RA is just 8-bits in the digital logic. The reason for the 16-bit value is that SUBT [0479] in ADDCK is a general-purpose, 16-bit subtraction routine that returns difference in R0 and R1. SUBT or ADDCK is called from a number of locations in the code. The RA MSB is not restricted to a zero value. In the Standard Interface application for device control and monitoring, there can be 256 command and 256 data addresses. In addition, there are 16 internal command and data functions so there can be 16 command addresses (with three forbidden addresses) and 16 monitor addresses. Thus in the Standard Interface application, for valid address situations, ADDCK can return an RA of 271 (i.e., for the address ranges of 0 to 255 device and 0 to 15 internal addresses). In the context of the VLBA Standard Interface application and the CMD usage of ADDCK, the RA MSB is 0 and the LSB contains the useable device Relative Address.

CMD0 [00B1] starts the execution of the device command. The LOST bit is cleared and SPIN is called to acquire CDH and its associated parity error which is saved in BIT3. CDH is saved in R5 and ACKN is called to return the ACK function code. After clearing the LOST flag, SPIN is called again to acquire CDL and its associated parity error which is retained in BIT0. CDL is saved in R4 for subsequent use. The CDH, CDL parity errors are tested by the JB BIT3,CMD1 and JNB BIT0,CMD2 instructions. If there is a command argument parity error, CMD1 (described below) performs the error operations.

CMD2 [00D1] performs the operations of setting up the 8156 and 8755 I/O ports for output, outputting the command and RA arguments, and sequencing through the interface-device interactive control states. These I/O operations are very similar to those performed in NUMBER [052E] when the device ID value was read. At this point the reader should briefly review the 8755 and 8156 I/O port characteristics described in Section 2.2, the Program Memory Map in Section 3.2 and the interface board logic description in Section 2.3.

The Data Pointer (DPTR) is used as an address pointer to load command values into the 8156 control register by MOVX (indirect) commands. The command register controls the operating modes of 8156 Ports A, B and C. 1100H is loaded into DPTR; this is the address of the 8156 control (and status) register. The first (30H) value output sets Ports A and B to the output mode. The second value output (3BH) enables Ports A and B interrupts and sets Port C to the ALT 4 mode in which the Port C lines become interactive I/O lines for Ports A and B. In the Standard Interface application, only PC2 and PC5 are used to strobe the status register to set the Port A and B interrupt flags in the status register. The device DEV ACK response triggers the strobe one-shot.

ADDCK [045F] was called in LOOP ([005A] above). ADDCK returned with the Relative Address LSB in R2 and MSB in R3. R3 and R2 are loaded into registers B and A, respectiveley for output to the device. Remember that the value in R3 is zero so B is also zero. Next, the interface control discretes are set into B. Bit B.7 (IOR which makes DEV REQ) is set. Bit B.6 (R/W- which signals the device to accept a command and sets the 74LS245 buffer's direction) is cleared to the 0 (write) state. Bit B.5 (the command release) is cleared; it will be set below. Bit B.4 (buffer enable) is set; this enables the 74LS245 outputs to impress the Port A and B states on the CMD/MON bus. After the control states have been set up in B, it is pushed onto the STACK for temporary storage; the command release bit, B.5 will be merged into the control discretes below.

The address of the 8755 Port A (0) is loaded into the DPTR for the imminent output and the LOST bit is cleared. OUTIT [0408] is called to output the A and B registers to the 8755 ports A and B. (See OUTIT described above). These two outputs provide RA and R/W- to the device logic. DPTR is set to the 8755 Port B address and the STACK is popped onto A. (Note that this is the only place in the firmware where one register state is moved to another register via the STACK). Bit A.5, the command release bit, is set in A and is output to the 8755 Port B. The other 8755 Port B bits are not perturbed when bit 5 is set in the port latch. This last bit activates the DEV REQ line; the device logic should decode RA and strobe the command into the RA-designated register or logic. Note that the delay between the two sets of 8755 outputs provides address-settling delay for the device logic. This period is 336 oscillator periods - about 30 microseconds.

The device logic must accept the command argument and respond with a DEV ACK signal within 500 usec after DEV REQ went true. The CMDOU1 loop is a 63-count, 500 usec loop counter that recurrently tests the device DEV ACK line. Register B is loaded with the loop count value. The 8156 status register address 1100H is set into DPTR and the MOVX A,DPTR instruction loads A with the 8156 port status. ANL A,#9 masks out all 8156 status register bits except for the Port A and B interrupt flags. When DEV ACK goes true it triggers one-shot U12-7 which sets the 8156 ports A and B interrupt flags in the status register. The CJNE A,#9,CMDOU2 comparison, jump-not-equal instruction tests these two flags; if not present, the jump to CMDOU2 is taken. When these two flags become true (within the loop period), control falls through the CJNE instruction to the AJMP CMD3 instruction to exit the loop before the time-out. The last instruction in the loop, DJNZ B,CMDOU1, decrements the count in the B register and tests the count state. If B is not zero, the jump to the loop start (CMDOU1) is taken. When B = 0, control falls into the next instruction which is the no-device-response code described below.

The CMD3 [0129] code is reached when the DEV ACK signal goes true within the 500 usec allotted time. This code sends the DC1 function code to the Controller by calling DC1 [0426] which outputs the code via SPEOUT [0428] (described above). Upon return from SPEOUT, the 8755 Port B address (#1) is loaded into DPTR, A is cleared and is output to 8755 Port B. This zero state disables the bus driver outputs and clears the four Port B command discretes. DEV ACK goes to the zero state and R/W- remains in the Write (zero) state. RA remains at the command address state; there is no reason to change it.

The command receive counter (BE-2) address is loaded into R0 and INCRE [040F] is called to increment the counter.

Upon return from INCRE, the command address and arguments standing in registers R2 through R5 are loaded by MOV direct,Rn instructions into the the interface monitor and control functions in RAM for subsequent readout by the Controller. The command address is loaded into BE-9 and the argument is loaded into BE-8. After these four transfers, control is returned to LOOP to re-start the XMT bus message detection process. This completes the device command operations when there are no CDH, CDL parity errors and the device responds within the allotted 500 usec.

If CDH or CDL has a parity error, the parity test [00C2] in CMD0 transfers control to CMD1 [00C5]. In CMD1, the parity error LED memory bit PARX is set low; when clocked in SYNC, it will illuminate the front-panel PARX LED. The command data parity error counter BE-6 is incremented by setting the counter address 47H into R0. After clearing the LOST bit, INCRE is called to increment the counter and upon return, NACK [051A] (and SPEOUT [0428]) to output the NAK function code to the Controller via the RCV bus. Upon return from SPEOUT, control is returned to LOOP to re-start the XMT bus message detection process.

In CMDOU1 [0106], if the device does not respond to the DEV REQ signal within 500 usec, control falls through the DJNZ B,CMDOU1 loop control instruction to the device non-response code immediately below.

The DC2 [041E] function code is called to output this non-response code to the controller via SPEOUT [0428]. Upon return, the control discretes are cleared to zero which resets the interface stimulus signals to the quiescent state, just as described in CMD3 above. The command address and argument values in R2 through R5 are stored in BE-9 and BE-8 just as described three paragraphs above. After this is done, the RAM address (37H) counter is loaded into R0. The LOST bit is cleared and INCRE is called to increment this counter just as in the cases above. Upon return from INCRE, control is returned to LOOP to re-start the XMT bus message detection process.

The other major branch in CMD, CMD4 [0141] is taken when the address comparision at [00AF] determines that the message is an internal command. In this case, the message address is one of the top 16 addresses (BE-15 through BE-0) dedicated to the internal monitor and control functions. The action of these internal commands is to set the command argument value in CDH and CDL into the address designated by ADH and ADL. Typically the CDH, CDL value is a zero to reset the counters but it could be any value. Some of these addresses (BE-10, BE-3 and BE-0) cannot be written into because they contain interface parameters which should not be altered. The logic to inhibit this storage is described below.

At CMD4, the first action is to clear the LOST bit. After this, SPIN [03E8] is called to obtain CDH and the associated parity error bit which is stored in BIT3. CDH is saved in R5 and ACKN [0422] is called to return the ACK function code via SPEOUT [0428]. Upon return from SPEOUT, SPIN is called again to obtain CDL and its associated parity error bit which is in BIT0 and C. BIT3 and C are tested by a pair of JB, JNC instructions which transfer control to CMD1 to execute the command parity error response. CMD1 operations were described above.

If CDH and CDL are not tainted by parity errors, CDL is saved in R4 and the DPTR is loaded with the address of the CMD5 location 015FH. CMD5 is the start of a table of addresses and jumps to the LOADO subroutine (below). The Relative Address is used to form an index to the table which starts at address 015F. The index is added to the start address to form an entry address to the table. Each BE-N counter-jump entry point is 4 addresses past the previous entrry point so the RA is multiplied by 4 to form the index. Upon return from ADDCK [045F] in LOOP [005A] above, R0 contains the LSB of the Relative address. R0 is moved to A, the value 4 is loaded into B and the MUL AB instruction forms the table index. Next, the JMP @A+DPTR transfers control to the appropriate table entry point.

At each table entry point, R0 is loaded with the address of the BE-N MSB and an absolute jump to LOADO [019F] transfers control to the subroutine. In LOADO, the contents of R5 (containing CDH) is stored in the address pointed to by the contents of R0, an indirect storage operation. R0 is then incremented and the contents of R4 (CDL) are stored in the address pointed to by the contents of R0. After storing R4, control is transferred to CMD3 (described above) to complete the internal command operation by returning a DC1 function code to the Controller and storing the command address and argument. CMD3 returns control to LOOP after completing its operations.

Now how does the code avoid writing command arguments to forbidden locations ? Note the BE-10 entry point. At these forbidden locations entry points, the first instruction is an absolute jump to CMD3, the normal destination after updating a BE-N counter. This avoids any perturbation of the three forbidden locations. Two NOP's after the jump maintain the uniformity of the table entry points.

The algorithm for CMD is as follows:

```
        BEGIN  CMD
        IF  INTERNAL COMMAND  THEN  GOTO  CMD4
        ELSE  CLEAR LOST
        CALL  SPIN AND OBTAIN CDH AND ASSOCIATED PARITY BIT
        R4  EQ. CDH
        CALL  ACKN, RETURN
        CLEAR LOST
        CALL  SPIN AND OBTAIN CDL AND ASSOCIATED PARITY BIT
        R4  EQ. CDL
        IF  CDH OR CDL PARITY ERROR  THEN  SET PARX,  INCREMENT COMMAND PARITY ERROR COUNTER,  CLEAR LOST,
              CALL NACK,  GOTO  LOOP
        ELSE  SET  8156 TO STROBE OUTPUT MODE
        A  EQ. RA LOW
        B  EQ. RA HIGH
        SET DATA REQ BIT, WRITE BIT-, BFFR ENABLE BIT INTO B,  PUSH B
        CALL  OUTIT, SET RA AND CONTROL DISCRETES IN 8755 PORTS A AND B
        SET CMD RELEASE BIT
        B  EQ. 63
        DPTR  EQ.  1100H
CMDOU1  IF  DEV ACK  THEN  CALL DC1, CLEAR DEV REQ, BUFFER ENABLE, R/W-,
              INCREMENT COMMAND COUNTER, SAVE COMMAND ADDRESS AND ARGUMENT,
              GOTO  LOOP
        ELSE  DECREMENT B
        IF  B  EQ. 0  THEN  CALL DC2, CLEAR DEV REQ, BUFFER ENABLE, R/W-,
              SAVE COMMAND ADDRESS AND ARGUMENT,  CLEAR LOST,
              INCREMENT NO-RESPONSE COUNTER,  GOTO  LOOP
        ELSE  GOTO  CMDOU1
        END

CMD4    CLEAR  LOST
        CALL SPIN AND OBTAIN  CDH AND ASSOCIATED PARITY BIT
        R5  EQ.  CDH
        CALL ACKN  AND RETURN
        CALL SPIN AND OBTAIN  CDL AND ASSOCIATED PARITY BIT
        IF  CDH OF CDL PARITY ERROR  THEN GOTO  CMD1
        ELSE  A  EQ.  CDL
        DPTR  EQ. TABLE START ADDRESS
        TABLE INDEX  EQ.  RA*4
        TABLE ADDRESS  EQ. TABLE START ADDRESS + TABLE INDEX
        GOTO TABLE ADDRESS,  R0  EQ.  COUNTER ADDRESS
        GOTO LOADO,  IN LOADO  COUNTER EQ. R5,  INCREMENT COUNTER ADDRESS,  COUNTER  EQ. R4,  GOTO CMD3
```

The first two instructions in CMD (at addresses 00A1H and 00A3H) may seem strange because of an AD 2500 assembler quirk which identifies Bank registers by their address values rather than by a mnemonic label. In these cases, the MOV R2,00H instruction is actually MOV R2,R0 and the second instruction is MOV R3,R1. The op code of the first is AA 00 and the op code of the second is AB 01. These are MOV Rn,direct instructions (see page 6-54 in the 1991 edition of Intel's 8-Bit Embedded Controllers). The instruction format is: 1010 1rrr direct addr where rrr is a register address. The leading A is clear enough, what about the rest ? The address of Register 2 is 010 so this accounts for the second A. The address of Register 0 is 000; this accounts for the last two 0's. The reader is encouraged to interpret the second instruction opcode.

MON

MON [01A6] is the subroutine called at the end of LOOP [0071] to execute a data request message. On entry to MON, the Relative Address (RA) is contained in registers R0 (LSB) and R1 (MSB). The value of the MSB is 0. ADH is in R3 and ADL is in R2.

There are several tasks to be performed in MON. The first task is to distinguish between a request for device data and a request for internal interface data. If device data is requested, the 8755 and 8156 I/O ports must be activated to interact with the device logic to input the data. The device logic must respond to the DEV REQ within 500 usec. If it does not respond with a DEV ACK or ANENB within this time, the firmware assumes that the device is non-responsive. If the request is for internal interface data such as address parity errors (BE-7), the RA is used as an index to the table in RAM to access the data. (See the description of internal interface data in Section 2.1). The function codes and data are output on the RCV bus. In addition, MON acquires and parity tests CDH and CDL. Although this data is of no value in executing the data request message, CDH and CDL parity errors must be recorded and the CDH, CDL data saved in BE-8.

MON is composed of mostly simple in-line code; interacting with the device logic and I/O ports requires a lot of sequential states. MON also uses a number of calls to lower-level subroutines such as SPIN [03E8], SUBT [0479], OUTIT [0408], INCRE [040F] and the function code drivers ACKN [0422], etc. In spite of the bulk, MON is relatively simple because there are only a few simple logic decisions to be made.

The first operation in MON is to disable the serial port interrupt by clearing IEC.4. The Controller will not be transmitting any messages on the XMT bus until the interface has responded to the data request message. The next operation is to distinguish between device data requests and requests for internal monitor data. This is done by an address comparison. The relative address in R0 and R1 is loaded into R2 and R3, respectively. The start address of the internal interface command-monitor address block (IBASE1,2) is loaded into DPH (MSB) and DPL (LSB), respectivley, and SUBT [0479] to subtract IBASE1,2 (in DPTR) from the Relative address (in R2, R3). If RA < IBASE1,2, the C (carry) bit is set which indicates a device data request message and the JC MONC transfers control to the code to gather and output device data. If C = 0, the request was for internal monitor data and the AJMP MON2 transfers control to this set of code which will read out the requested data. MON2 is described below.

MONC [01B8] begins the operation of interacting with the device logic by setting up the 8156 for input. The DPTR is loaded with 1100H, the 8156 command register address. A is loaded with the command argument 38H that sets Ports A and B to input mode, sets the ALT 4 (strobed) mode and enables the Ports A and B inputs so that the DEV ACK or ANENB response will set the interrupt flags (AD3 and AD0) in the 8156 status register. (The reader is urged to briefly review the 8156 and 8755 I/O port descriptions in Section 2.3). The 8156 control argument is output to the 8156 by an indirect MOVX (external memory) instruction MOVX @DPTR,A. The 8156 address in DPTR is incremented to point to Port A and the (indeterminate value) Port A data is input (and discarded by being overwritten). DPTR is incremented again to point to PORT B and the indeterminate value in Port B is read and discarded. The reason that these two ports are each cycled through a read operation is that the 8156 could have just been used for an output operation in which the interrupt flag remains set after dropping low in the output cycle. Output mode to input mode (and vice versa) transitions do not clear these interrupt flags. Exercising the two ports through an input cycle clears them. The reader can see this effect in the two 8156 strobed mode timing diagrams in Section 2.2.

The 8755 ports are set up next. R2 contains the lower byte of RA and R3 contains the upper byte. R2 is loaded into A and R3 (which is zero) is loaded into B. The control discretes which condition the interface board output lines are next set up in B. B.6 (the R/W- bit) is set, which makes the R/W-

line to the device logic high and also sets the CMD/MON bus to the input direction. B.4 (BFR ENBL) is set which enables the 74LS245 CMD/MON buffers to drive the bus. B.5, the Cmd Release is set low (this is a data operation). B.7, IOR, the data request is initially set low but will be set high below. With this state set in B, the DPTR is set to 0, the address of 8755 Port A. The LOST bit (PSW.5) is cleared and OUTIT [0408] is called to output the A and B register to the two ports by another MOVX, (indirect to external memory) instruction. Upon return, B.7, the IOR bit is set in B (B was not altered by OUTIT) and the new state is output to Port B by the MOVX instruction. IOR activates DEV REQ. The reason that the IOR bit was set in Port B after than the other bits were set is that this provides the device logic a settling delay of about 204 oscillator periods - about 18 usec. The decode of RA and the set-up of RA-dependent device function thus has a 21 usec time margin before device action is initiated by DEV REQ. At this point, all conditions have been established for the device logic to initiate the data acquistion operation. The device logic should impress the data on the bus and raise the DEV ACK signal which will clock the data into the 8156 ports.

Next, the serial ports are enabled, the LOST bit is cleared and SPIN [03E8] is called to obtain CDH and its associated parity error bit which is saved in BIT3. ACKN [0422] is called next to output the ACK functon code to the Controller. The LOST bit is cleared again and SPIN is called to obtain CDL and the associated parity bit which remains in BIT0. The JB and JNB parity test instructions transfer control to MONO (discussed below) to record the detection of the error.

At MONIN0 [01F4], the 8156 status register address (1100H) is loaded into DPTR to test for the presence of the Ports A and B interrupt flag bits. A count of 24 is loaded into B for the impending data input-interrupt flag test loop MONIN1. MONIN1 reads into A the status of the 8156 status register by the MOVX A,DPTR instruction, again an indirect move from external memory. The ANL A,#9 instruction masks out all bits in A except for the two port interrupt flag bits. CJNE A,#9,MONIN2, the next instruction compares the contents of A (the interrupt flag bits, if present) with the bit pattern: 0000 1001. If the compared states are identical (i.e., both flag bits are present), control falls through to the AJMP MONIN3 which transfers control to MONIN3 (described below) to read the device data standing in the 8156 ports. If the bit patterns are not identical, the CJNE jump transfers control to MONIN2, a DJNZ B,MONIN1. This instruction decrements B, and if B is not zero after decrementing, it transfers control back to MONIN1 to repeat the interrupt flag test loop. The loop period is 24 counts or about 190 usec.

If the interrupt flag bits are not detected after 24 cycles through the test loop, control falls into the device no-response code. The DPTR is loaded with the 8755 Port B address and A is loaded with the count of 50 which is output to Port B. This state clears PB7 (IOR), holds PB4 (R/W-) set, holds PB5 (CMD REL) cleared and PB4 (BUFR EN) set. Thus DEV REQ is set low but the other lines to the device are not altered. When DEV REQ goes low, the device logic should disable the drive to the CMD/MON bus and disable the device RA decode logic. The next operation clears A and outputs the state to Port B; all device control discretes are now reset. This should disqualify all device operations until the next DEV REQ. The time interval between the two outputs is about 3 usec. Having cleared the device control discretes, the DC2 function code is called to report the non-response event. Upon return from outputting the DC2 function code, R0 is loaded with 39H, the address of the LSB (MONRP2) of the monitor no-response counter in RAM. INCRE [040F] is called to increment the BE-11 counter. Upon return from INCRE, control is returned to LOOP since there is no available device data to return to the Controller.

In MONC [01B8] if a CDH or CDL parity error was detected, control falls into MON0 which records the error. The error does not compromise the data to be acquired from the device but the event must be recorded. This is done at MON0 which clears the PARX (P1.4) line to the LED display memory. When clocked in SYNC at [03DC] the PARX LED will be illuminated. The address of the command data parity error counter LSB (MDPE2), 47H, is loaded into R0 and INCRE [040F] is called to increment the BE-6 counter. Upon return from INCRE, control falls into MONIN0 as described above. Counting the CDH, CDL parity errors is an incidental task.

At MONIN3 [0218], the destination of the interrupt flag test loop in MONIN1 above, the task of inputting the device data standing in the 8156 ports begins. From the test loop operations above, the DPTR was set to 1100H, the 8156 status register address. DPTR is next incremented twice to point to 8156 Port B, the MSB. The Port B data is read by the MOVX instruction and saved in R3. The DPTR is decremented to point to Port A and the data LSB is saved in R2.

Having read and stored the device data, the next step is to disconnect the device by resetting the control discretes set in 8755 Port A. The next five instructions are identical to those at the start in MONIN2, the no-response shut-down of the control discretes. Since these were described above, there is no need to repeat them here.

After clearing the device control discretes, the DPTR is loaded with 1100H, the address of the 8156 control register and the value 38H is loaded into A and output to the register with a MOVX instruction. This value commands the 8156 ports A and B to the output, Alt 4 mode which is the quiescent state for these ports unless commanded differently (done in NUMBER [052E] for example).

After resetting the 8156 ports, the data is output to the Controller at MON1 [022F]. First, DOUT (P1.3) is cleared, when clocked in SYNC [at 03DC], it will illuminate the DOUT LED. The MDH byte (in R3) is loaded into A and SPOOUT [043F] is called to output the data to the Controller with odd parity. Upon return from SPOOUT, MDL (in R2) is loaded into A and SPOOUT is called to output the data to the Controller.

Upon return from SPOOUT [043F], R0 is loaded with 4DH, the address of the count of correctly received monitor data request messages, BE-1. The LOST bit is cleared and INCRE [040F] is called to increment the counter. This is the last operation of acquiring and outputting device monitor data. Upon return from INCRE, control is returned to LOOP to resume the message detection process.

MON2 [023F] is the other main branch of MON which loads and outputs internal monitor data. The operations in MON2 and MON3 (below) are simpler than those involved in acquiring device data since there are no device interactions; all that is required is to use RA as an index to a jump-address table (MON3) to load data from the appropriate RAM addresses. After testing the CDH, CDL parity, a jump to MON1 causes the data to be transmitted to the Controller.

At MON2, the instruction MOV DPTR,#MON3 loads the address of the MON3 location into the DPTR; this is the base address of the jump-address table. Note that the jump table entry point intervals are 6 addresses; the table entry index is thus 6 addresses. R0 contains the Relative Address and B is loaded with 6. The next instruction MUL AB multiplies R0 by 6 and puts the results in A. The JMP @A+DPTR instruction transfers control to the address which is the sum of the base address (0248H) and the index in A. In each jump-table, a pair of consecutive RAM addresses of the BE-N data are loaded into R3 and R2. The first address is the MSB of the data value. After loading R3 and R2, an AJMP MON4 transfers control to MON4 where the parity states of CDH and CDL are evaluated.

The code in MON4 [02A6] from 02A6H to 02C1H is identical to the code in MONC (above) from 01D7H to 01F2H; this is the device data path described above. Since it is identical there is no need to repeat the description. After executing this MON4 code, the AJMP MON1 transfers control to MON1 [022F] which outputs it to the Controller. The MON1 operations were described above.

An important point about MON4 is that it is the entry point for a jump from STMON [030B] to read out the 2N and 2N + 1 address values requested from STATUS by a data request message with 2N and 2N + 1 addresses.

The MON algorithm is as follows.

```
        BEGIN  MON
        DISABLE SERIAL PORT INTERRUPT
        IF  INTERNAL MONITOR DATA REQUEST  THEN  FORM INDEX INTO JUMP-ADDRESS TABLE,
                ENABLE SERIAL PORT INTERRUPT,  CLEAR PSW.5,  CALL SPIN & OBTAIN CDH & ASSOCIATED PARITY ERROR,
                CALL ACKN AND OUTPUT ACK FUNCTION CODE,  CLEAR PSW.5,  CALL SPIN & OBTAIN CDL &
                ASSOCIATED PARITY ERROR
                IF  CDH OR CDL PARITY ERROR  THEN  SET PARX, INCREMENT CDH, CDL PARITY ERROR COUNTER, GOTO MON1
                ELSE  GOTO  MON1
        ELSE  SET 8156 PORTS A & B TO INPUT, ALT 4 MODE,  SET DEV ACK TEST DELAY TO 24,  SET UP RA AND CONTROL
                DISCRETES IN A & B,  CALL OUTIT & SET RA & CONTROL DISCRETES IN 8755 PORTS A & B,
                ENABLE SERIAL PORTS,  CLEAR PSW.5,  CALL SPIN & OBTAIN CDH & ASSOCIATED PARITY ERROR,  CALL
                ACKN & OUTPUT ACK FUNCTION CODE,  CLEAR PSW.5,  CALL SPIN & OBTAIN CDL & ASSOCIATED
                PARITY ERROR
        IF  CDH OR CDL PARITY ERROR  THEN  SET PARX, INCREMENT CDH-CDL PARITY ERROR COUNTER,  GOTO  MONINO
        ELSE  CONTINUE
MONINO  SET DEV ACK TEST DELAY COUNT  EQ. 24
        IF  DEV ACK  THEN  READ MDH & MDL FROM 8156 PORTS & STORE,  CLEAR PSW.5,  CLEAR DEV REQ
        ELSE  CONTINUE
        DECREMENT COUNT
        IF  COUNT  EQ. 0  THEN  CLEAR DEV REQ,  CALL DC2 & OUTPUT DC2 FUNCTION CODE,
                GOTO MON1
        ELSE  GOTO  MONINO
MON1    SET DOUT,  CALL SPOOUT & OUTPUT MDH,  CALL SPOOUT & OUTPUT MDL, CLEAR PSW.5,  INCREMENT MON DATA COUNTER
        GOTO  LOOP
        END
```

In MON at address 0247H is the instruction JMP @A,+DPTR with a 73 opcode. This is a jump indirect. The Intel data book shows this to be a JMP @A+DPTR with an instruction opcode of: 0111 0011. This instruction adds the eight-bit unsigned contents of the Accumulator with the sixteen-bit DATA POINTER, and loads the resulting sum into the PC. This will be the address of the subsequent instruction fetches. Sixteen bit addition is performed (modulo 2^{16}): a carry-out from the low-order eight bits propogates through the high order bits. Neither the Accumulator nor the DATA POINTER is altered. No flags are affected.

LOST

Distributed through the firmware are instructions which "CLR PSW.5". PSW.5 is a PSW (program status register) bit which is user-assignable. Other than the CLR PSW.5 and SET PSW.5 instructions, no other processor operations affect the state of this bit (another bit, PSW.1 also has the same property). In the firmware, this bit is cleared before (most) subroutines are called and is tested after exiting the subroutine by a jump to LOST [0587]. The last part of NUMBER (at 0587H) has the label "LOST" which tests PSW.5.

At LOST we see the following code:

```
LOST:   JNB     PSW.5,LOST1
        LJMP    START
LOST1:  SETB    PSW.5
        RET
```

On each pass through a subroutine with the CLR PSW.5 instruction before the call, under normal conditions, the LOST test should find the bit cleared. If it is found cleared, it is set again in preparation for the next subroutine pass. The RET instruction returns control to the location following the subroutine call.

If the bit is found set at LOST, it is clear that something has perturbed the state of the bit, probably a power glitch. When the 5-volt power is glitched by a short sag, the 8032 evidences a proclivity for spurious changes of some register bits to 1's. Curiously, there does not seem to be a tendency for spurious changes to 0's. The logic of the LOST code is the assumption that if this bit has been perturbed, other register states are probably questionable, which could cause firmware malfunctions. If the test finds the PSW.5 bit set, the LJMP to START re-initializes the interface. This is the only possible action.

The Controller must recognize the interface reset by testing the response to data request messages. An interface reset sets the address block to the default values; monitor data requests to the assigned interface block will not evoke a data or function code response. See the descriptions of STATUS [02C5], INTIAL [034C] and NUMBER [052E] routines for details on the implications of a processor reset.

The LOST code was added to the firmware subsequent to the firmware development to reduce vulnerability to power perturbation effects. Logic analyzer observations of malfunctioning software at the Pietown VLBA site showed that the processor could become "stuck" in a subroutine or execute it twice in succession although called just once. In a series of tests, the LOST code was installed, the 5-volt power was glitched, and CLR PSW.5 instructions were added one at a time before each type of subroutine call. As a result of these tests, the firmware has a much greater tolerance for power glitches but severe power sags or dropouts will perturb the firmware; this is to be expected of devices that are only characterized for 5-volt, +/- 10% power.

In LOST at location 058DH is the instruction SETB PSW.5 with an opcode of D2 D5. This is a SETB bit instruction. The Intel data book shows the opcode of: 1101 00010 bit address. The PSW register is an SFR with an address of D0H. The PSW is bit addressible so bit PSW.5 has an address of D5H.

The following are examples of subroutine flow and show the PSW.5 - LOST sequence. There are three cases as shown below. The bracketed hex address shows the first occurance of the case.

```
1) Subroutine sequences which start with CLR PSW.5:

[03CC]  CLR PSW.5,  ACALL NUMBER,  AJMP LOST, RET

[00EC]  CLR PSW.5,  ACALL  OUTIT,  AJMP LOST, RET

[0046]  CLR PSW.5,  ACALL SPIN,  ACALL READ,  AJMP LOST,  RET

[0077]  CLR PSW.5,  ACALL INCRE,  AJMP LOST,  RET

[03B5]  CLR PSW.5,  ACALL INCRE,  AJMP SYNC4, LJMP LOST,  RET

[031B]  CLR PSW.5,  ACALL NUM1,  LOST,  RET

[03D0]  CLR PSW.5,  AJMP LOST,  RET

2) Subroutine sequences which are not prefaced by  CLR PSW.5 but have CLR PSW.5 embedded in them:

[00B8]   ACALL ACKN,  AJMP SPEOUT, JZ SPEXIT, CLR PSW.5, ACALL COUL2, ACALL NUMBER, LOST, RET

[0111]   ACALL DC2,  AJMP SPEOUT,  JZ SPEXIT, CLR PSW.5, ACALL COUL2, ACALL NUMBER, LOST, RET

[0129]   ACALL DC1,  AJMP SPEOUT,  JZ SPEXIT, CLR PSW.5, ACALL COUL2, ACALL NUMBER, LOST, RET

[00CD]   ACALL NACK,  AJMP SPEOUT,  JZ SPEXIT, CLR PSW.5, ACALL COUL2, ACALL NUMBER, LOST, RET

[03B0]   DSYNC,  CLR PSW.5,  ACALL INCRE,  AJMP SYNC4,  LJMP START

[003E]   ACALL SYNC, CLR PSW.5,  ACALL INCRE,  CLR PSW.5,  ACALL READ,  RET
```

[04B5] TIMER1, CLR PSW.5, ACALL COUL2, ACALL NUMBER, LOST, RET

3) Subroutine sequences which do not CLR PSW.5 but terminate at LOST:

[0036] ACALL INTIAL, ACALL NUMBER, LOST, RET

4) Subroutine sequences which are not associated with CLR PSW.5 or LOST

[005A] ACALL ADDCK, ACALL SUBT, RET

[00AB] ACALL SUBT, RET

3.4 Firmware Program Listing

```
              2500 A.D. 8051 CROSS ASSEMBLER   -   VERSION 2.00a
              ------------------------------------------------

                   INPUT  FILENAME :  PDSET9.ASM
                   OUTPUT FILENAME :  PDSET9.OBJ


                            TITLE   PROCESSOR DATA SET

                            ;PROCESSOR DATA SET
                            ;WRITTEN BY WAYNE M. KOSKI
                            ;LAST REVISION: AUGUST 2, 1988


          ;MACRO SECTION


SETPCON:MACRO

          ;THE SETPCON MACRO IS USED TO DOUBLE THE BAUD RATE OF THE
          ;SERIAL PORT.

          ;ENTRY           NONE
          ;AFFECTS         REGISTARS: PCON

          DB       75H,87H,80H      ;SET THE DOUBLE BAUD RATE BIT
          ENDM


 SET2CON: MACRO

          ;THE SET2CON MACRO IS USED TO TURN ON TIMER 2.

          ;ENTRY           NONE
          ;AFFECTS         REGISTARS: T2CON, RCAP2H, RCAP2L

          DB       75H,0CAH,0       ;CLEAR COUNTERS
          DB       75H,0CBH,0
          DB       75H,0C8H,4       ;TURN ON TIMER 2
          ENDM


 XORB:    MACRO    ARG1,ARG2,ARG3

          ;THE XORB MACRO IS USED TO EXCLUSIVLY OR ARG1 AND ARG2 TOGETHER
          ;AND IT PLACES THE RESULT IN ARG3, WHERE ARG1,ARG2, AND ARG3
          ;ARE SINGLE BITS.

          ;ENTRY           BITS: ARG1,ARG2,ARG3
          ;AFFECTS BITS:   CARRY FLAG, ARG3

         MBIT1:   REG      ARG1
         MBIT2:   REG      ARG2
                  MOV      C,ARG1
                  ANL      C,/MBIT2
                  MOV      ARG3,C
                  MOV      C,ARG2
                  ANL      C,/MBIT1
                  ORL      C,ARG3
                  MOV      ARG3,C
                  ENDM
```

```
                        ;DATA DEFINITION AREA
                        ;THE FIRST TWO LOCATIONS ARE INPUTTED BY THE USER

00 44           MODEL:  EQU     'D'
00 42           REV:    EQU     'B'

7F F0           BASE:   EQU     7FF0H           ;DEFAULT ADDRESS SPACE
00 10           POINTS: EQU     16              ;DEFAULT NUMBER OF MON/CMD POINTS
30              RES1:   REG     30H             ;RESERVED
31              RES2:   REG     31H
32              RES3:   REG     32H
33              RES4:   REG     33H
34              RES5:   REG     34H
35              RES6:   REG     35H
36              DATRP1: REG     36H             ;COMMAND DEVICE NO RESPONSE
37              DATRP2: REG     37H
38              MONRP1: REG     38H             ;MONITOR NO RESPONSE
39              MONRP2: REG     39H
3A              MOD1:   REG     3AH             ;MODEL/REVISION
3B              MOD2:   REG     3BH
3C              CMDAD1: REG     3CH             ;COMMAND ADDRESS OF LAST MESSAGE
3D              CMDAD2: REG     3DH
3E              CMDDT1: REG     3EH             ;COMMAND DATA OF LAST MESSAGE
3F              CMDDT2: REG     3FH
40              ADDPE1: REG     40H             ;ALL ADDRESS PARITY ERRORS
41              ADDPE2: REG     41H
42              DATPE1: REG     42H             ;ALL DATA PARITY ERRORS
43              DATPE2: REG     43H
44              BADSY1: REG     44H             ;BAD SYNC
45              BADSY2: REG     45H
46              MDTPE1: REG     46H             ;MY DATA PARITY ERRORS
47              MDTPE2: REG     47H
48              NUMB1:  REG     48H             ;"N"
49              NUMB2:  REG     49H
4A              DATCO1: REG     4AH             ;COMMAND COUNT
4B              DATCO2: REG     4BH
4C              MONCO1: REG     4CH             ;MONITOR COUNTS
4D              MONCO2: REG     4DH
4E              BLOCK1: REG     4EH             ;START OF BLOCK
4F              BLOCK2: REG     4FH
50              COUNT1: REG     50H             ;ADDRESS SPACE COUNT
51              COUNT2: REG     51H             ;ADDRESS SPACE COUNT
52              EBASE1: REG     52H             ;END OF ADDRESS SPACE
53              EBASE2: REG     53H             ;END OF ADDRESS SPACE
54              IBASE1: REG     54H             ;INTERNAL ADDR
55              IBASE2: REG     55H             ;END OF ADDRESS SPACE
56              TIME1:  REG     56H             ;ONE MINUTE SYNC TIMER LSB
57              TIME2:  REG     57H             ;ONE MINUTE SYNC TIMER MSB
58              TIME3:  REG     58H             ;5 SECOND READ ID BYTE TIMER
59              BUFFC:  REG     59H             ;CURRENT BUFFER INDEX VALUE
5A              BUFFN:  REG     5AH             ;NEXT BUFFER INDEX VALUE
90              RCVEN:  REG     P1.0            ;XMIT TO COMPUTER ENABLE
91              MSG:    REG     P1.1            ;VALID MSG RCV STATUS
92              BUSY:   REG     P1.2            ;INTERFACE ACTIVE STATUS
93              DOUT:   REG     P1.3            ;INTERFACE TRANSMITTING STATUS
94              PARX:   REG     P1.4            ;PARITY ERROR DETECTED STATUS
95              ADDEN:  REG     P1.5            ;EXTERNAL ADDRESS REQUEST LINE
97              RCVEX:  REG     P1.7            ;EXTERNAL XMIT TO COMPUTER ENABLE
D0              P:      REG     PSW.0           ;ACC PARITY FLAG LOCATION
99              TI:     REG     SCON.1          ;XMIT INT FLAG LOCATION
98              RI:     REG     SCON.0          ;RCV INT FLAG LOCATION
8E              TR1:    REG     TCON.6          ;TIMER 1 RUN CNTRL BIT LOCATION
9A              RB8:    REG     SCON.2          ;RCV DATA BIT (PARITY BIT) LOCATION
9B              TB8:    REG     SCON.3          ;XMIT DATA BIT (PARITY BIT) LOCATION
00              BIT0:   REG     20H.0           ;RESULT OF THE XORB MACRO FUNCTION
```

```
      01              BIT1:   REG     20H.1           ;TEMP STORAGE LOCATIONS FOR THE
      02              BIT2:   REG     20H.2           ;XORB MACRO FUNCTION
      03              BIT3:   REG     20H.3           ;STORED RESULT OF THE XORB MACRO FUNCTION
      04              BIT4:   REG     20H.4           ;RESULT OF THE LOWER ADDR BOUND CHECK 0=PASS
      05              BIT5:   REG     20H.5           ;RESULT OF THE UPPER ADDR BOUND CHECK 1=PASS
      06              BIT6:   REG     20H.6           ;ID BYTE REQUEST FLAG
      07              BIT7:   REG     20H.7           ;XMIT FLAG  1=CAN XMIT
      08              BIT8:   REG     21H.0           ;SYNC CARRY FLAG
      09              BIT9:   REG     21H.1           ;SIZE ERROR FLAG


0000                          ORG     0

0000                  START:
0000  01 30                 AJMP    START1          ;JMP OVER INT VECTORS


                              ;SERIAL PORT INTERRUPT VECTOR

0023                          ORG     23H

0023  81 83                 AJMP    SERP            ;SERIAL PORT INT ROUTINE


                              ;TIMER 2 INTERRUPT VECTOR

002B                          ORG     2BH

002B  81 B5                 AJMP    TIMER           ;TIMER 2 INT ROUTINE


                              ;START OF MAIN PROGRAM

0030                          ORG     30H             ;START IT HERE

0030                  START1:
0030  75 81 07              MOV     SP,#7           ;RESET STACK POINTER
0033  75 D0 00              MOV     PSW,#0          ;RESET BANK = 0
0036  71 4C                 ACALL   INTIAL          ;INITIALIZE HARDWARE
0038                  LOOP:
0038  75 81 07              MOV     SP,#7           ;RESET STACK POINTER
003B  75 D0 00              MOV     PSW,#0          ;RESET BANK = 0
003E  71 B7                 ACALL   SYNC            ;WAIT TILL WE SYNC UP
0040  D2 92                 SETB    BUSY            ;CLEAR LIGHTS
0042  D2 93                 SETB    DOUT
0044  D2 94                 SETB    PARX
0046  C2 D5                 CLR     PSW.5           ;LOST FLAG
0048  71 E8                 ACALL   SPIN            ;GET UPPER ADDRESS
004A  92 03                 MOV     BIT3,C          ;SAVE PARITY ERROR
004C  F5 F0                 MOV     B,A             ;SAVE IN REG B
004E  C2 E7                 CLR     A.7             ;CLR CMD/MON BIT
0050  70 02                 JNZ     START2          ;CONTINUE IF >255
0052  41 C5                 AJMP    STATUS          ;IF <=255
0054  FB              START2: MOV     R3,A            ;SAVE ADH IN R3
0055  C2 D5                 CLR     PSW.5           ;LOST FLAG
0057  71 E8                 ACALL   SPIN            ;GET LOWER ADDRESS
0059  FA                    MOV     R2,A            ;SAVE IN REG R2
005A  91 5F                 ACALL   ADDCK           ;CHECK IF ADDR IS IN RANGE
005C  20 00 14              JB      BIT0,PARCI      ;IF ADDR PE
005F  20 03 11              JB      BIT3,PARCI      ;IF ADDR PE
0062  20 04 16              JB      BIT4,NOTUS      ;OUT OF RANGE
0065  30 05 13              JNB     BIT5,NOTUS      ;OUT OF RANGE
0068  D2 97                 SETB    RCVEX           ;ENABLE EXTERNAL RCV BUS
006A  D2 90                 SETB    RCVEN           ;ENABLE RCV BUS
006C  C2 92                 CLR     BUSY            ;SET INTERFACE BUSY STATUS LIGHT
```

```
006E  20 F7 30              JB      B.7,CMD         ;GOTO COMMAND ROUTINE IF SET
0071  21 A6                 AJMP    MON             ;GOTO MONITOR ROUTINE

0073                  PARCI:
0073  C2 94                 CLR     PARX            ;SET PARITY ERROR STATUS LIGHT
0075  78 41                 MOV     R0,#41H         ;ALL ADDR PE COUNTER
0077  C2 D5                 CLR     PSW.5
0079  91 0F                 ACALL   INCRE           ;ADD ONE
007B  75 F0 28        NOTUS: MOV     B,#40
007E  D5 F0 FD               DJNZ    B,$             ;DELAY ABOUT 87US
0081  C2 90                 CLR     RCVEN           ;TURN OFF RCV BUS DRIVER
0083  C2 97                 CLR     RCVEX           ;TURN OFF EXTERNAL BUS DRIVER
0085  C2 D5                 CLR     PSW.5           ;LOST FLAG
0087  71 E8                 ACALL   SPIN            ;GET DATA HIGH
0089  92 03                 MOV     BIT3,C          ;SAVE PARITY
008B  C2 D5                 CLR     PSW.5           ;LOST FLAG
008D  71 E8                 ACALL   SPIN            ;GET DATA LOW
008F  20 03 05              JB      BIT3,PARCI1     ;IF DATA PE
0092  20 00 02              JB      BIT0,PARCI1     ;IF DATA PE
0095  01 38                 AJMP    LOOP            ;GET NEXT MESS
0097  C2 94           PARCI1: CLR     PARX            ;SET PARITY ERROR STATUS LIGHT
0099  78 43                 MOV     R0,#43H         ;ALL DATA PE COUNTER
009B  C2 D5                 CLR     PSW.5
009D  91 0F                 ACALL   INCRE           ;ADD ONE
009F  01 38                 AJMP    LOOP

00A1                  CMD:
00A1  AA 00                 MOV     R2,00H          ;MOVE REL ADDRESS
00A3  AB 01                 MOV     R3,01H          ;
00A5  85 54 83              MOV     DPH,IBASE1      ;IS IT AN INTERNAL COMMAND?
00A8  85 55 82              MOV     DPL,IBASE2
00AB  91 79                 ACALL   SUBT
00AD  40 02                 JC      CMD0
00AF  21 41                 AJMP    CMD4
00B1  C2 D5           CMD0: CLR     PSW.5           ;CLEAR LOST FLAG
00B3  71 E8                 ACALL   SPIN            ;GET UPPER DATA BYTE
00B5  92 03                 MOV     BIT3,C          ;SAVE PE
00B7  FD                    MOV     R5,A            ;SAVE IT
00B8  91 22                 ACALL   ACKN            ;XMIT ACKNOWLEDGE
00BA  C2 D5                 CLR     PSW.5           ;LOST FLAG
00BC  71 E8                 ACALL   SPIN            ;GET LOWER DATA BYTE
00BE  FC                    MOV     R4,A            ;SAVE IT
00BF  20 03 03              JB      BIT3,CMD1       ;IF PARITY ERROR
00C2  30 00 0C              JNB     BIT0,CMD2       ;NO PARITY ERROR
00C5                  CMD1:
00C5  C2 94                 CLR     PARX            ;SET PARITY ERROR STATUS LIGHT
00C7  78 47                 MOV     R0,#47H         ;POINT TO DATA PE COUNTER
00C9  C2 D5                 CLR     PSW.5
00CB  91 0F                 ACALL   INCRE           ;INCREMENT IT
00CD  91 1A                 ACALL   NACK            ;SEND NEGATIVE ACK
00CF  01 38                 AJMP    LOOP
00D1                  CMD2:
00D1  90 11 00              MOV     DPTR,#1100H     ;8156 CMD REG
00D4  74 30                 MOV     A,#30H          ;CLEAR PORT C
00D6  F0                    MOVX    @DPTR,A
00D7  74 3B                 MOV     A,#3BH          ;OUTPUT MODE
00D9  F0                    MOVX    @DPTR,A         ;SET IT
00DA  EA                    MOV     A,R2            ;LOAD ADDR LOW
00DB  8B F0                 MOV     B,R3            ;LOAD ADDR HIGH
00DD  D2 F7                 SETB    B.7             ;SET DATA REQ
00DF  C2 F6                 CLR     B.6             ;SET TO WRITE
00E1  C2 F5                 CLR     B.5             ;NO CMD RELEASE YET
00E3  D2 F4                 SETB    B.4             ;ENABLE BUFFERS
00E5  C0 F0                 PUSH    B
00E7  75 83 00              MOV     DPH,#0          ;8755 PORT ADDR
```

```
00EA  C2 D5              CLR     PSW.5
00EC  91 08              ACALL   OUTIT           ;OUTPUT ADDRESS AND CNTRL
00EE  EC                 MOV     A,R4            ;GET DATA
00EF  8D F0              MOV     B,R5            ;GET DATA
00F1  75 83 11           MOV     DPH,#11H        ;POINT TO DATA STORAGE
00F4  C2 D5              CLR     PSW.5
00F6  91 08              ACALL   OUTIT
00F8  90 00 01           MOV     DPTR,#1H        ;SEND RELEASE BIT
00FB  D0 E0              POP     A
00FD  D2 E5              SETB    A.5
00FF  F0                 MOVX    @DPTR,A
0100  75 F0 3F           MOV     B,#63           ;WAIT COUNT
0103  90 11 00           MOV     DPTR,#1100H     ;PORT C STATUS LOCATION
0106  E0           CMDOU1: MOVX  A,@DPTR
0107  54 09              ANL     A,#9            ;DEVICE ACK?
0109  B4 09 02           CJNE    A,#9,CMDOU2     ;IF ACK
010C  21 29              AJMP    CMD3            ;IF DEVICE ACK
010E  D5 F0 F5     CMDOU2: DJNZ  B,CMDOU1
0111  91 1E              ACALL   DC2             ;NO DEVICE NEG ACK
0113  90 00 01           MOV     DPTR,#1         ;POINT TO ADDR HIGH
0116  74 00              MOV     A,#0            ;TURN OFF REQ
0118  F0                 MOVX    @DPTR,A
0119  8B 3C              MOV     3CH,R3
011B  8A 3D              MOV     3DH,R2
011D  8D 3E              MOV     3EH,R5
011F  8C 3F              MOV     3FH,R4
0121  78 37              MOV     R0,#37H         ;INC COMMAND NO RESPONSE
0123  C2 D5              CLR     PSW.5
0125  91 0F              ACALL   INCRE
0127  01 38              AJMP    LOOP
0129               CMD3:
0129  91 26              ACALL   DC1             ;SEND ACKNOWLEDGE
012B  90 00 01           MOV     DPTR,#1         ;POINT TO ADDR HIGH
012E  74 00              MOV     A,#0            ;TURN OFF REQ
0130  F0                 MOVX    @DPTR,A
0131  78 4B              MOV     R0,#4BH
0133  C2 D5              CLR     PSW.5
0135  91 0F              ACALL   INCRE           ;INC CMD RCV COUNTER
0137  8B 3C              MOV     3CH,R3
0139  8A 3D              MOV     3DH,R2
013B  8D 3E              MOV     3EH,R5
013D  8C 3F              MOV     3FH,R4
013F  01 38              AJMP    LOOP
0141               CMD4:
0141  C2 D5              CLR     PSW.5           ;LOST FLAG
0143  71 E8              ACALL   SPIN            ;GET UPPER CMD BYTE
0145  92 03              MOV     BIT3,C          ;SAVE PARITY BIT
0147  FD                 MOV     R5,A            ;STORE BYTE
0148  91 22              ACALL   ACKN            ;XMIT ACKNOWLEDGE
014A  C2 D5              CLR     PSW.5           ;LOST FLAG
014C  71 E8              ACALL   SPIN            ;GET LOWER CMD BYTE
014E  20 03 02           JB      BIT3,CMD4A      ;IF PARITY ERROR
0151  50 02              JNC     CMD4B           ;IF NO PARITY ERROR
0153  01 C5        CMD4A: AJMP   CMD1            ;IF PARITY ERROR
0155  FC           CMD4B: MOV    R4,A            ;STORE BYTE
0156  90 01 5F           MOV     DPTR,#CMD5      ;INDEX INTO INTERNAL COMMAND POINT
0159  E8                 MOV     A,R0
015A  75 F0 04           MOV     B,#4
015D  A4                 MUL     AB
015E  73                 JMP     @A+DPTR
015F  78 30        CMD5: MOV     R0,#30H         ;BE-15
0161  21 9F              AJMP    LOAD0
0163  78 32              MOV     R0,#32H         ;BE-14
0165  21 9F              AJMP    LOAD0
0167  78 34              MOV     R0,#34H         ;BE-13
```

```
0169   21 9F                    AJMP    LOAD0
016B   78 36                    MOV     R0,#36H         ;BE-12
016D   21 9F                    AJMP    LOAD0
016F   78 38                    MOV     R0,#38H         ;BE-11
0171   21 9F                    AJMP    LOAD0
0173   21 29                    AJMP    CMD3            ;BE-10
0175   00                       NOP
0176   00                       NOP
0177   78 3C                    MOV     R0,#3CH         ;BE-09
0179   21 9F                    AJMP    LOAD0
017B   78 3E                    MOV     R0,#3EH         ;BE-08
017D   21 9F                    AJMP    LOAD0
017F   78 40                    MOV     R0,#40H         ;BE-07
0181   21 9F                    AJMP    LOAD0
0183   78 42                    MOV     R0,#42H         ;BE-06
0185   21 9F                    AJMP    LOAD0
0187   78 44                    MOV     R0,#44H         ;BE-05
0189   21 9F                    AJMP    LOAD0
018B   78 46                    MOV     R0,#46H         ;BE-04
018D   21 9F                    AJMP    LOAD0
018F   21 29                    AJMP    CMD3            ;BE-03
0191   00                       NOP
0192   00                       NOP
0193   78 4A                    MOV     R0,#4AH         ;BE-02
0195   21 9F                    AJMP    LOAD0
0197   78 4C                    MOV     R0,#4CH         ;BE-01
0199   21 9F                    AJMP    LOAD0
019B   21 29                    AJMP    CMD3            ;BE-00
019D   00                       NOP
019E   00                       NOP
019F                    LOAD0:
019F   A6 05                    MOV     @R0,05H
01A1   08                       INC     R0
01A2   A6 04                    MOV     @R0,04H
01A4   21 29                    AJMP    CMD3


01A6                    MON:
01A6   C2 AC                    CLR     IEC.4           ;DISABLE SERIAL PORT
01A8   AA 00                    MOV     R2,00H          ;SAVE REL ADDRESS
01AA   AB 01                    MOV     R3,01H          ;
01AC   85 54 83                 MOV     DPH,IBASE1      ;IS IT AN INTERNAL MONITOR
01AF   85 55 82                 MOV     DPL,IBASE2
01B2   91 79                    ACALL   SUBT
01B4   40 02                    JC      MONC
01B6   41 3F                    AJMP    MON2
01B8   90 11 00         MONC:   MOV     DPTR,#1100H     ;POINT TO 8156 CMD REG
01BB   74 38                    MOV     A,#38H          ;INPUT MODE
01BD   F0                       MOVX    @DPTR,A         ;SET IT
01BE   A3                       INC     DPTR            ;READ PORT DATA
01BF   E0                       MOVX    A,@DPTR         ;AND DISCARD
01C0   A3                       INC     DPTR            ;
01C1   E0                       MOVX    A,@DPTR         ;
01C2   EA                       MOV     A,R2            ;GET ADDR LOW
01C3   8B F0                    MOV     B,R3            ;GET ADDR HIGH
01C5   C2 F7                    CLR     B.7             ;NO DATA REQ
01C7   D2 F6                    SETB    B.6             ;SET FOR READ
01C9   C2 F5                    CLR     B.5             ;NO CMD RELEASE
01CB   D2 F4                    SETB    B.4             ;ENABLE BUFFERS
01CD   90 00 00                 MOV     DPTR,#0         ;POINT TO 8755 PORTS
01D0   C2 D5                    CLR     PSW.5
01D2   91 08                    ACALL   OUTIT
01D4   D2 E7                    SETB    A.7             ;SET DATA REQ
01D6   F0                       MOVX    @DPTR,A
01D7   43 A8 90                 ORL     IEC,#90H        ;ENABLE SERIAL PORTS
```

```
01DA  C2 D5              CLR     PSW.5           ;LOST FLAG
01DC  71 E8              ACALL   SPIN            ;GET DATA HIGH
01DE  92 03              MOV     BIT3,C          ;SAVE PARITY
01E0  91 22              ACALL   ACKN            ;XMIT ACKNOWLEDGE
01E2  C2 D5              CLR     PSW.5           ;LOST FLAG
01E4  71 E8              ACALL   SPIN            ;GET DATA LOW
01E6  20 03 03           JB      BIT3,MON0       ;IF PARITY ERROR
01E9  30 00 08           JNB     BIT0,MONIN0     ;IF NO PARITY ERROR
01EC  C2 94        MON0: CLR     PARX            ;LIGHT PARITY LIGHT
01EE  78 47              MOV     R0,#47H         ;POINT TO DATA PE COUNTER
01F0  C2 D5              CLR     PSW.5
01F2  91 0F              ACALL   INCRE           ;INCREMENT IT
01F4  90 11 00     MONIN0: MOV   DPTR,#1100H     ;POINT TO STATUS
01F7  75 F0 18           MOV     B,#24           ;COUNT
01FA  E0           MONIN1: MOVX  A,@DPTR         ;GET STATUS
01FB  54 09              ANL     A,#9            ;STRIP
01FD  84 09 02           CJNE    A,#9,MONIN2
0200  41 18              AJMP    MONIN3          ;IF DATA IS THERE
0202  D5 F0 F5     MONIN2: DJNZ  B,MONIN1
0205  90 00 01           MOV     DPTR,#1         ;POINT TO ADDR HIGH
0208  74 50              MOV     A,#50H          ;TURN OFF DEV REQ
020A  F0                 MOVX    @DPTR,A
020B  74 00              MOV     A,#0            ;QUIESCENT STATE
020D  F0                 MOVX    @DPTR,A
020E  91 1E              ACALL   DC2             ;SEND NO DEVICE NEG ACK
0210  78 39              MOV     R0,#39H         ;INC MONITOR NO RESPONSE
0212  C2 D5              CLR     PSW.5
0214  91 0F              ACALL   INCRE
0216  01 38              AJMP    LOOP
0218  A3           MONIN3: INC   DPTR            ;POINT TO DATA HIGH
0219  A3                   INC   DPTR
021A  E0                 MOVX    A,@DPTR         ;GET IT
021B  FB                 MOV     R3,A            ;STORE
021C  15 82              DEC     DPL             ;POINT TO DATA LOW
021E  E0                 MOVX    A,@DPTR         ;GET IT
021F  FA                 MOV     R2,A            ;STORE
0220  90 00 01           MOV     DPTR,#1         ;POINT TO ADDR HIGH
0223  74 50              MOV     A,#50H          ;TURN OFF DEV REQ
0225  F0                 MOVX    @DPTR,A
0226  74 00              MOV     A,#0            ;QUIESCENT STATE
0228  F0                 MOVX    @DPTR,A
0229  90 11 00           MOV     DPTR,#1100H
022C  74 3B              MOV     A,#3BH          ;SET PORT TO OUTPUT
022E  F0                 MOVX    @DPTR,A
022F                MON1:
022F  C2 93              CLR     DOUT            ;SET MONITOR MSG LIGHT
0231  EB                 MOV     A,R3            ;GET UPPER ADDRESS
0232  91 3F              ACALL   SPOOUT          ;SEND UPPER DATA
0234  EA                 MOV     A,R2            ;GET LOWER ADDRESS
0235  91 3F              ACALL   SPOOUT          ;SEND LOWER DATA
0237  78 4D              MOV     R0,#4DH         ;POINT TO DATA COUNTER
0239  C2 D5              CLR     PSW.5
023B  91 0F              ACALL   INCRE
023D  01 38              AJMP    LOOP
023F                MON2:
023F  90 02 48           MOV     DPTR,#MON3      ;INDEX INTO INTERNAL MONITOR POINT
0242  E8                 MOV     A,R0
0243  75 F0 06           MOV     B,#6
0246  A4                 MUL     AB
0247  73                JMP     @A+DPTR
0248                MON3:
0248  AB 30              MOV     R3,30H          ;BE-15
024A  AA 31              MOV     R2,31H
024C  41 A6              AJMP    MON4
024E  AB 32              MOV     R3,32H          ;BE-14
```

```
0250  AA 33                 MOV     R2,33H
0252  41 A6                 AJMP    MON4
0254  AB 34                 MOV     R3,34H          ;BE-13
0256  AA 35                 MOV     R2,35H
0258  41 A6                 AJMP    MON4
025A  AB 36                 MOV     R3,36H          ;BE-12
025C  AA 37                 MOV     R2,37H
025E  41 A6                 AJMP    MON4
0260  AB 38                 MOV     R3,38H          ;BE-11
0262  AA 39                 MOV     R2,39H
0264  41 A6                 AJMP    MON4
0266  AB 3A                 MOV     R3,3AH          ;BE-10
0268  AA 3B                 MOV     R2,3BH
026A  41 A6                 AJMP    MON4
026C  AB 3C                 MOV     R3,3CH          ;BE-09
026E  AA 3D                 MOV     R2,3DH
0270  41 A6                 AJMP    MON4
0272  AB 3E                 MOV     R3,3EH          ;BE-08
0274  AA 3F                 MOV     R2,3FH
0276  41 A6                 AJMP    MON4
0278  AB 40                 MOV     R3,40H          ;BE-07
027A  AA 41                 MOV     R2,41H
027C  41 A6                 AJMP    MON4
027E  AB 42                 MOV     R3,42H          ;BE-06
0280  AA 43                 MOV     R2,43H
0282  41 A6                 AJMP    MON4
0284  AB 44                 MOV     R3,44H          ;BE-05
0286  AA 45                 MOV     R2,45H
0288  41 A6                 AJMP    MON4
028A  AB 46                 MOV     R3,46H          ;BE-04
028C  AA 47                 MOV     R2,47H
028E  41 A6                 AJMP    MON4
0290  C2 D5                 CLR     PSW.5
0292  B1 2E                 ACALL   NUMBER          ;BE-03
0294  41 A6                 AJMP    MON4
0296  AB 4A                 MOV     R3,4AH          ;BE-02
0298  AA 4B                 MOV     R2,4BH
029A  41 A6                 AJMP    MON4
029C  AB 4C                 MOV     R3,4CH          ;BE-01
029E  AA 4D                 MOV     R2,4DH
02A0  41 A6                 AJMP    MON4
02A2  AB 4E                 MOV     R3,4EH          ;BE-00
02A4  AA 4F                 MOV     R2,4FH
02A6                MON4:
02A6  43 A8 90              ORL     IEC,#90H        ;ENABLE SERIAL PORT INTERRUPT
02A9  C2 D5                 CLR     PSW.5           ;LOST FLAG
02AB  71 E8                 ACALL   SPIN            ;GET DATA HIGH
02AD  92 03                 MOV     BIT3,C          ;SAVE PARITY
02AF  91 22                 ACALL   ACKN            ;XMIT ACKNOWLEDGE
02B1  C2 D5                 CLR     PSW.5           ;LOST FLAG
02B3  71 E8                 ACALL   SPIN            ;GET DATA LOW
02B5  20 03 03              JB      BIT3,MON5       ;IF PARITY ERROR
02B8  30 00 08              JNB     BIT0,MON6       ;IF NO PARITY ERROR
02BB  C2 94         MON5:   CLR     PARX            ;LIGHT PARITY LIGHT
02BD  78 47                 MOV     R0,#47H         ;POINT TO DATA PE COUNTER
02BF  C2 D5                 CLR     PSW.5
02C1  91 0F                 ACALL   INCRE           ;INCREMENT IT
02C3  41 2F         MON6:   AJMP    MON1            ;NOW SEND IT

02C5  FB            STATUS: MOV     R3,A            ;STORE ADH IN R3
02C6  30 03 02              JNB     BIT3,STAT1      ;PE?
02C9  01 54                 AJMP    START2          ;IF PARITY ERROR
02CB  E5 49         STAT1:  MOV     A,NUMB2         ;GET "N"
02CD  20 D0 02              JB      P,STAT2         ;"N" PE?
02D0  01 54                 AJMP    START2          ;IF "N" PE
```

```
02D2  C2 E7           STAT2:  CLR     A.7             ;CLEAR PE BIT
02D4  23                      RL      A               ;*2
02D5  C0 F0                   PUSH    B               ;SAVE B
02D7  F5 F0                   MOV     B,A             ;PUT "N"*2 INTO B
02D9  C2 D5                   CLR     PSW.5           ;LOST FLAG
02DB  71 E8                   ACALL   SPIN            ;GET ADL
02DD  FA                      MOV     R2,A            ;SAVE ADL IN R2
02DE  30 00 04                JNB     BIT0,STAT3      ;PE?
02E1  D0 F0                   POP     B               ;RESTORE STACK
02E3  01 73                   AJMP    PARCI           ;IF PE
02E5  B5 F0 02        STAT3:  CJNE    A,B,STAT4       ;US?
02E8  41 F5                   AJMP    STAT5           ;IF US
02EA  05 F0           STAT4:  INC     B               ;B="N"*2+1
02EC  B5 F0 02                CJNE    A,B,STAT4A      ;US? SECOND CHECK
02EF  41 F5                   AJMP    STAT5           ;IF US
02F1  D0 F0           STAT4A: POP     B               ;RESTORE STACK
02F3  01 7B                   AJMP    NOTUS           ;NOT US
02F5  D2 97           STAT5:  SETB    RCVEX           ;TURN ON EXTERNAL BUS
02F7  D2 90                   SETB    RCVEN           ;TURN ON RCV BUS
02F9  D0 F0                   POP     B               ;GET ADH
02FB  20 F7 0F                JB      B.7,STCMD       ;IF SET DO COMMAND

02FE  20 E0 06        STMON:  JB      A.0,STMON1      ;IF UPPER "N"
0301  AB 50                   MOV     R3,COUNT1
0303  AA 51                   MOV     R2,COUNT2
0305  41 A6                   AJMP    MON4
0307  AB 4E           STMON1: MOV     R3,4EH
0309  AA 4F                   MOV     R2,4FH
030B  41 A6                   AJMP    MON4

030D  20 E0 0F        STCMD:  JB      A.0,STCMD4      ;IF UPPER
0310  78 50                   MOV     R0,#50H
0312  71 2E                   ACALL   STCMD5
0314  A6 05           STCMD2: MOV     @R0,05H         ;LOAD DATA
0316  08                      INC     R0
0317  A6 04                   MOV     @R0,04H
0319  C2 D5                   CLR     PSW.5
031B  B1 59                   ACALL   NUM1            ;NOW RESIZE SYSTEM
031D  21 29           STCMD3: AJMP    CMD3

031F  78 4E           STCMD4: MOV     R0,#4EH
0321  71 2E                   ACALL   STCMD5          ;GET DATA
0323  30 09 F7                JNB     BIT9,STCMD3     ;START ADDRESS TOO SMALL
0326  ED                      MOV     A,R5            ;GET HIGH ORDER
0327  C3                      CLR     C
0328  94 80                   SUBB    A,#80H          ;TOO HIGH?
032A  50 F1                   JNC     STCMD3          ;START IS TOO HIGH
032C  61 14                   AJMP    STCMD2

032E                  STCMD5:
032E  C2 D5                   CLR     PSW.5           ;LOST FLAG
0330  71 E8                   ACALL   SPIN
0332  FD                      MOV     R5,A            ;SAVE CDH IN R5
0333  92 03                   MOV     BIT3,C          ;SAVE PARITY ERROR
0335  C2 09                   CLR     BIT9
0337  60 02                   JZ      STCMD6
0339  D2 09                   SETB    BIT9

033B  91 22           STCMD6: ACALL   ACKN            ;XMIT ACKNOWLEDGE
033D  C2 D5                   CLR     PSW.5           ;LOST FLAG
033F  71 E8                   ACALL   SPIN            ;GET CDL
0341  20 03 02                JB      BIT3,STCMD7     ;IF PARITY ERROR
0344  50 04                   JNC     STCMD8          ;IF OK
0346  D0 F0           STCMD7: POP     B               ;RESTORE STACK
0348  01 C5                   AJMP    CMD1            ;IF PARITY ERROR
```

```
034A   FC              STCMD8: MOV     R4,A            ;SAVE CDL IN R4
034B   22                      RET
034C

034C                   INTIAL:

                               ;THIS ROUTINE INITIALIZES THE SERIAL AND PARALLEL PORTS
                               ;AND INTERNAL MEMORY LOCATIONS.

                               ;ENTRY          NONE
                               ;AFFECTS        MEMORY LOCATIONS: 30H-50H
                               ;               REGISTARS: R0,DPTR,A,SCON,TH1,TMOD,TCON,IEC
                               ;               BITS: TR1,BIT7,BUSY,MSG,DOUT,PARX

034C   78 30                   MOV     R0,#30H         ;LOAD 30H-50H WITH INITIAL INFO
034E   79 0A                   MOV     R1,#10
0350                   COUL:
0350   76 00                   MOV     @R0,#0
0352   08                      INC     R0
0353   D9 FB                   DJNZ    R1,COUL
0355   76 44                   MOV     @R0,#MODEL
0357   08                      INC     R0
0358   76 42                   MOV     @R0,#REV
035A   08                      INC     R0
035B   79 12                   MOV     R1,#18
035D                   COUL1:
035D   76 00                   MOV     @R0,#0
035F   08                      INC     R0
0360   D9 FB                   DJNZ    R1,COUL1
0362   76 7F                   MOV     @R0,#BASE/256
0364   08                      INC     R0
0365   76 F0                   MOV     @R0,#BASE.AND.0FFH
0367   75 50 00                MOV     COUNT1,#0
036A   75 51 10                MOV     COUNT2,#POINTS
036D   75 A8 00        COUL2:  MOV     IEC,#0          ;DISABLE ALL INTERRUPTS
0370   C2 90                   CLR     RCVEN           ;TURN OFF XMIT TO MAIN COMPUTER
0372   C2 97                   CLR     RCVEX           ;TURN OFF EXTERNAL BUS DRIVER
0374   90 00 02                MOV     DPTR,#2         ;8755 PORTS ARE OUTPUTS
0377   74 FF                   MOV     A,#0FFH
0379   F0                      MOVX    @DPTR,A
037A   A3                      INC     DPTR
037B   F0                      MOVX    @DPTR,A
037C   B1 2E                   ACALL   NUMBER          ;GET "N"

037E                           SETPCON                 ;INITIALIZE TIMER1 AND SERIAL PORT

                               ;THE SETPCON MACRO IS USED TO DOUBLE THE BAUD RATE OF THE
                               ;SERIAL PORT.

                               ;ENTRY          NONE
                               ;AFFECTS        REGISTARS: PCON

037E   75 87 80                DB      75H,87H,80H     ;SET THE DOUBLE BAUD RATE BIT
                               ENDM
0381   75 B0 FF                MOV     P3,#0FFH        ;SET P3 ALTERNATE FUNCTION
0384   75 98 D0                MOV     SCON,#11010000B
0387   75 89 20                MOV     TMOD,#00100000B
038A   75 8D FF                MOV     TH1,#255        ;57600 BAUD
038D   D2 8E                   SETB    TR1             ;START TIMER1
038F   D2 07                   SETB    BIT7            ;SET XMIT FLAG
0391   D2 92                   SETB    BUSY            ;CLEAR LIGHTS
0393   D2 91                   SETB    MSG
0395   D2 93                   SETB    DOUT
0397   D2 94                   SETB    PARX
0399   75 59 60                MOV     BUFFC,#60H      ;CURRENT CHAR STORAGE
```

```
039C   75 5A 60                MOV     BUFFN,#60H      ;NEXT CHAR STORAGE

039F                           SET2CON

                               ;THE SET2CON MACRO IS USED TO TURN ON TIMER 2.

                               ;ENTRY          NONE
                               ;AFFECTS        REGISTARS: T2CON, RCAP2H, RCAP2L

039F   75 CA 00                DB      75H,0CAH,0      ;CLEAR COUNTERS
03A2   75 CB 00                DB      75H,0CBH,0
03A5   75 C8 04                DB      75H,0C8H,4      ;TURN ON TIMER 2
                               ENDM
03A8   D2 BC                   SETB    IP.4            ;SERIAL PORT = HIGHEST INT
03AA   75 A8 90                MOV     IEC,#90H        ;ENABLE ALL INTS, SERIAL PORT
03AD   32                      RETI

03AE                   DSYNC:
03AE   78 45                   MOV     R0,#45H
03B0   C2 D5                   CLR     PSW.5
03B2   91 0F                   ACALL   INCRE
03B4   09                      INC     R1              ;INC CHAR COUNTER
03B5   61 E2                   AJMP    SYNC4

03B7                   SYNC:

                               ;THIS ROUTINE WAITS FOR THE SYNC CHAR (16H EVEN PARITY)
                               ;IF IT SEES SYNC AND INCORRECT PARITY
                               ;IT INCREMENTS THE DISCARD SYNC COUNTER.

                               ;ENTRY          NONE
                               ;AFFECTS        REGISTAR: A, R1
                               ;               BITS: CARRY FLAG,RI
                               ;               MEMORY LOCATIONS: 3AH,3BH

03B7   75 56 4C                MOV     TIME1,#4CH      ;RESET ONE MINUTE TIMER
03BA   75 57 04                MOV     TIME2,#4
03BD   79 00                   MOV     R1,#0           ;RESET CHAR COUNTER
03BF   E5 59           SYNC1:  MOV     A,BUFFC         ;GOT A CHAR?
03C1   B5 5A 0C                CJNE    A,BUFFN,SYNC2   ;IF CHAR
03C4   43 A8 A0                ORL     IEC,#0A0H       ;ALLOW TIMER 2 TO INTERRUPT
03C7   30 06 06                JNB     BIT6,SYNC2      ;ID REQ?
03CA   C2 D5                   CLR     PSW.5
03CC   B1 2E                   ACALL   NUMBER          ;TIME TO DO NUMBER
03CE   61 BF                   AJMP    SYNC1
03D0   C2 D5           SYNC2:  CLR     PSW.5           ;LOST FLAG
03D2   91 E6                   ACALL   READ            ;GET CHAR
03D4   92 08                   MOV     BIT8,C          ;SAVE PARITY FLAG
03D6   B4 16 08                CJNE    A,#16H,SYNC3    ;WAIT UNTIL IT IS 16H
03D9   30 08 D2                JNB     BIT8,DSYNC      ;IF ODD PARITY,NOT SYNC BYTE
03DC   D2 91                   SETB    MSG             ;CLOCK IN CURRENT LIGHTS
03DE   C2 91                   CLR     MSG             ;SET ACTIVE STATUS LIGHTS
03E0   22                      RET
03E1   09              SYNC3:  INC     R1              ;INC CHAR COUNTER
03E2   B9 08 DA        SYNC4:  CJNE    R1,#8,SYNC1     ;IF CHAR < 8
03E5   02 00 00                LJMP    START           ;WE ARE LOST

03E8                   SPIN:

                               ;THIS ROUTINE GET A CHAR AND TESTS FOR ODD PARITY.

                               ;ENTRY          NONE
                               ;AFFECTS        REGISTAR: A
                               ;               BITS: CARRY FLAG,BIT0,BIT1,BIT2
                               ;                     RI
```

```
03E8   E5 59                    MOV     A,BUFFC         ;GOT A CHAR?
03EA   B5 5A 03                 CJNE    A,BUFFN,SPIN1   ;IF CHAR
03ED   43 A8 A0                 ORL     IEC,#0A0H       ;ENABLE TIMER 2 TO INTERRUPT
03F0   91 E6            SPIN1:  ACALL   READ            ;GET CHAR
03F2   B3                       CPL     C
03F3   92 01                    MOV     BIT1,C
03F5   A2 D0                    MOV     C,P
03F7   92 02                    MOV     BIT2,C

03F9                            XORB    BIT1,BIT2,BIT0

                                ;THE XORB MACRO IS USED TO EXCLUSIVLY OR ARG1 AND ARG2 TOGETHER
                                ;AND IT PLACES THE RESULT IN ARG3, WHERE ARG1,ARG2, AND ARG3
                                ;ARE SINGLE BITS.

                                ;ENTRY          BITS: ARG1,ARG2,ARG3
                                ;AFFECTS        BITS: CARRY FLAG, ARG3

       01               MBIT1:  REG     BIT1
       02               MBIT2:  REG     BIT2
03F9   A2 01                    MOV     C,BIT1
03FB   B0 02                    ANL     C,/MBIT2
03FD   92 00                    MOV     BIT0,C
03FF   A2 02                    MOV     C,BIT2
0401   B0 01                    ANL     C,/MBIT1
0403   72 00                    ORL     C,BIT0
0405   92 00                    MOV     BIT0,C
                                ENDM
0407   22                       RET


0408                    OUTIT:
                                ;THIS ROUTINE OUTPUT TWO BYTES WHOSE ADDRESS IS CONTAINED
                                ;IN DPTR.

                                ;ENTRY          REGISTARS: A,B=DATA TO BE OUTPUTTED. DPTR=ADDRESS
                                ;AFFECTS        REGISTARS: A,DPTR

0408   F0                       MOVX    @DPTR,A         ;LOWER BYTE
0409   A3                       INC     DPTR
040A   E5 F0                    MOV     A,B
040C   F0                       MOVX    @DPTR,A         ;UPPER BYTE
040D   A1 87                    AJMP    LOST

040F                    INCRE:

                                ;THIS ROUTINE INCREMENTS A 16 BIT COUNTER BY 1 POINTED
                                ;TO BY THE INDEX REGISTAR R0.

                                ;ENTRY          REGISTARS: R0=ADDRESS OF COUNTER LSB
                                ;AFFECTS        REGISTARS: A,R0
                                ;               MEMORY LOCATIONS: @R0,@R0-1

040F   E6                       MOV     A,@R0
0410   24 01                    ADD     A,#1
0412   F6                       MOV     @R0,A
0413   18                       DEC     R0
0414   E6                       MOV     A,@R0
0415   34 00                    ADDC    A,#0
0417   F6                       MOV     @R0,A
0418   A1 87                    AJMP    LOST    ;CHECK FOR LOST

041A                    NACK:
```

```
                                ;THIS ROUTINE SENDS THE NEGATIVE ACKNOWLEDGE TO THE
                                ;MAIN COMPUTER.

                                ;ENTRY          NONE
                                ;AFFECTS        REGISTARS: A
041A   74 15                    MOV     A,#15H
041C   81 28                    AJMP    SPEOUT

041E                    DC2:

                                ;THIS ROUTINE SENDS THE SECOND NEGATIVE ACKNOWLEDGE TO THE
                                ;MAIN COMPUTER.

                                ;ENTRY          NONE
                                ;AFFECTS        REGISTARS: A

041E   74 12                    MOV     A,#12H
0420   81 28                    AJMP    SPEOUT

0422                    ACKN:

                                ;THIS ROUTINE SENDS THE  ACKNOWLEDGE TO THE
                                ;MAIN COMPUTER.

                                ;ENTRY          NONE
                                ;AFFECTS        REGISTARS: A

0422   74 06                    MOV     A,#6H
0424   81 28                    AJMP    SPEOUT

0426                    DC1:

                                ;THIS ROUTINE SENDS THE SECOND ACKNOWLEDGE TO THE
                                ;MAIN COMPUTER.

                                ;ENTRY          NONE
                                ;AFFECTS        REGISTARS: A

0426   74 11                    MOV     A,#11H


0428                    SPEOUT:
                                ;THIS ROUTINE OUTPUTS THE CHARACTOR WITH EVEN PARITY
                                ;CONTAINED IN (A) TO THE MAIN COMPUTER.

                                ;ENTRY          REGISTARS: A=CHAR
                                ;AFFECTS        REGISTARS: SBUF
                                ;               BITS: BIT7,CARRY FLAG,TB8

0428   A2 D0                    MOV     C,P
042A   92 9B                    MOV     TB8,C
042C   C0 E0                    PUSH    A               ;SAVE CHARACTER
042E   74 32                    MOV     A,#50           ;50 LOOP COUNTS
0430   20 07 05        SPE01:  JB      BIT7,SPE02      ;IF READY
0433   14                       DEC     A               ;A=A-1
0434   60 21                    JZ      SPEXIT          ;IF COUNT = 0
0436   81 30                    AJMP    SPE01           ;LOOP TIME = APPROX 380US
0438   D0 E0            SPE02:  POP     A               ;RESTORE CHARACTER
043A   C2 07                    CLR     BIT7
043C   F5 99                    MOV     SBUF,A
043E   22                       RET

043F                    SPOOUT:

                                ;THIS ROUTINE OUTPUTS THE CHARACTOR WITH ODD PARITY
```

```
                                ;CONTAINED IN (A) TO THE MAIN COMPUTER.

                                ;ENTRY          REGISTARS: A=CHAR
                                ;AFFECTS        REGISTARS: SBUF
                                ;               BITS: BIT7,CARRY FLAG,T88

043F   A2 D0                    MOV     C,P
0441   B3                       CPL     C
0442   92 9B                    MOV     T88,C
0444   C0 E0                    PUSH    A               ;SAVE CHARACTER
0446   74 32                    MOV     A,#50           ;50 LOOP COUNTS
0448   20 07 05          SP001:  JB      BIT7,SP002      ;IF READY
044B   14                       DEC     A               ;A=A-1
044C   60 09                    JZ      SPEXIT          ;IF COUNT = 0
044E   81 48                    AJMP    SP001           ;LOOP TIME = APPROX 380US
0450   D0 E0            SP002:  POP     A               ;RESTORE CHARACTER
0452   C2 07                    CLR     BIT7
0454   F5 99                    MOV     SBUF,A
0456   22                       RET

0457   D0 E0            SPEXIT: POP     A               ;RESTORE STACK
0459   C2 D5                    CLR     PSW.5
045B   71 6D                    ACALL   COUL2           ;MINI INITIALIZE
045D   01 38                    AJMP    LOOP            ;RESTART


045F                    ADDCK:

                                ;THIS ROUTINE CHECKS THE ADDRESS BOUNDRY AND PUT THE
                                ;RESULTING FLAGS IN BIT4, AND BIT5.

                                ;ENTRY          REGISTARS: R2,R3=ADDRESS TO CHECK
                                ;AFFECTS        REGISTARS: DPTR
                                ;               BITS: BIT4,BIT5

045F   85 4E 83                 MOV     DPH,BLOCK1              ;CHECK ADDRESS RANGE
0462   85 4F 82                 MOV     DPL,BLOCK2
0465   91 79                    ACALL   SUBT
0467   92 04                    MOV     BIT4,C                  ;STORE CARRY
0469   85 52 83                 MOV     DPH,EBASE1
046C   85 53 82                 MOV     DPL,EBASE2
046F   91 79                    ACALL   SUBT
0471   92 05                    MOV     BIT5,C
0473   85 4E 83                 MOV     DPH,BLOCK1              ;CREATE REL ADD
0476   85 4F 82                 MOV     DPL,BLOCK2


0479                    SUBT:

                                ;THIS ROUTINE SUBTRACTS WHAT CONTAINED IN R2,R3 BY DPTR
                                ;AND PLACES THE RESULT IN R0,R1.

                                ;ENTRY          REGISTARS: R2,R3=MINUEND DPTR=SUBTRAHEND
                                ;AFFECTS        REGISTARS: A,R0,R1
                                ;BITS:          CARRY FLAG

0479   EA                       MOV     A,R2            ;SUBTRACT R3,R2-DPH,DPL
047A   C3                       CLR     C               ;CLEAR BORROW
047B   95 82                    SUBB    A,DPL
047D   F8                       MOV     R0,A            ;STORE ANS IN R1,R0
047E   EB                       MOV     A,R3
047F   95 83                    SUBB    A,DPH
0481   F9                       MOV     R1,A
0482   22                       RET
```

```
0483                    SERP:

                                ;THIS IS THE SERIAL PORT INTERRUPT ROUTINE

                                ;ENTRY          VIA SERIAL INTERRUPT
                                ;AFFECTS        MEMORY LOCATIONS: 59H,5AH,60H-6FH
                                ;               BITS: TI,RI,BIT7

0483    30 99 04                        JNB     TI,SERP1        ;IF NOT TI
0486    D2 07                           SETB    BIT7            ;SAVE TX INT FLAG
0488    C2 99                           CLR     TI              ;CLR INT FLAG
048A    30 98 27        SERP1:  JNB     RI,SERP4        ;ESCAPE IF NO RCV FLAG
048D    C0 D0                           PUSH    PSW             ;SAVE THESE REGS
048F    C0 E0                           PUSH    A
0491    C0 00                           PUSH    00H
0493    A8 5A                           MOV     R0,BUFFN        ;INDEX NEXT CHAR
0495    A6 99                           MOV     @R0,SBUF        ;GET CHAR AND SAVE
0497    C8                              XCH     A,R0            ;EXCHANGE R0,A
0498    24 08                           ADD     A,#8            ;ADD 8
049A    C8                              XCH     A,R0            ;PUT BACK
049B    74 00                           MOV     A,#0            ;PARITY=0
049D    30 9A 01                        JNB     RB8,SERP2       ;IF PARITY BIT = 0
04A0    F4                              CPL     A               ;PARITY BIT = 1
04A1    F6              SERP2:  MOV     @R0,A           ;STORE PARITY BIT
04A2    05 5A                           INC     BUFFN           ;POINT TO NEXT ENTRY
04A4    74 68                           MOV     A,#68H          ;END OF BUFFER?
04A6    B5 5A 03                        CJNE    A,BUFFN,SERP3
04A9    75 5A 60                        MOV     BUFFN,#60H      ;IF END RESTART
04AC    C2 98           SERP3:  CLR     RI              ;CLEAR RCV INT FLAG
04AE    D0 00                           POP     00H             ;RESTORE REGS
04B0    D0 E0                           POP     A
04B2    D0 D0                           POP     PSW
04B4    32              SERP4:  RETI                    ;LEAVE AND RESTORE INT LEVELS


04B5                    TIMER:

                                ;TIMER ROUTINE DETERMINES WHEN WE TIME OUT
                                ;THE RCV ENABLE LINE AFTER 1 MINUTE OF DEAD
                                ;TIME AND REQUESTS NUMBER AFTER 5 SECONDS OF DEAD
                                ;TIME.

                                ;ENTRY                  VIA TIMER 2 INTERRUPT ROUTINE
                                ;AFFECTS                MEMORY LOCATIONS: 56H-58H

04B5    C0 D0                           PUSH    PSW
04B7    C0 E0                           PUSH    A
04B9    15 56                           DEC     TIME1
04BB    E5 56                           MOV     A,TIME1
04BD    70 11                           JNZ     TIMER3
04BF    15 57                           DEC     TIME2
04C1    E5 57                           MOV     A,TIME2
04C3    70 06                           JNZ     TIMER2
04C5    C2 D5           TIMER1: CLR     PSW.5
04C7    71 6D                           ACALL   COUL2
04C9    01 38                           AJMP    LOOP
04CB    C3              TIMER2: CLR     C
04CC    94 05                           SUBB    A,#5
04CE    50 F5                           JNC     TIMER1
04D0    15 58           TIMER3: DEC     TIME3
04D2    E5 58                           MOV     A,TIME3         ;TIME YET
04D4    70 02                           JNZ     TIMER4          ;IF NOT
04D6    D2 06                           SETB    BIT6
04D8                    TIMER4: SET2CON
```

```
                                ;THE SET2CON MACRO IS USED TO TURN ON TIMER 2.

                                ;ENTRY          NONE
                                ;AFFECTS                    REGISTARS: T2CON, RCAP2H, RCAP2L

04D8  75 CA 00                  DB      75H,0CAH,0          ;CLEAR COUNTERS
04DB  75 CB 00                  DB      75H,0CBH,0
04DE  75 C8 04                  DB      75H,0C8H,4          ;TURN ON TIMER 2
                                ENDM
04E1  D0 E0                     POP     A
04E3  D0 D0                     POP     PSW
04E5  32                        RETI

04E6                    READ:

                                ;THIS ROUTINE READS A CHAR STORED IN THE BUFFER AND
                                ;ITS CORRESPONDING PARITY BIT. IF NO CHAR IN BUFFER
                                ;IT WAITS FOR A CHAR AND INPUTS IT DIRECT.

                                ;ENTRY                      NONE
                                ;AFFECTS                    REGISTARS: A
                                ;                           MEMORY LOCATIONS: 56H-5FH
                                ;                           BITS: CARRY FLAG,BIT6

04E6  C2 AC                     CLR     IEC.4               ;DISABLE SERIAL PORT TEMP
04E8  E5 59                     MOV     A,BUFFC             ;GET CURRENT CHAR LOCATION
04EA  B5 5A 1B                  CJNE    A,BUFFN,READ2       ;IF CHAR IN BUFFER
04ED  43 A8 A0          READ1:  ORL     IEC,#0A0H           ;ENABLE TIMER2
04F0  53 C8 84                  DB      53H,0C8H,84H        ;ANL T2CON,84H
04F3  43 C8 04                  DB      43H,0C8H,4          ;ORL T2CON,4
04F6  30 CF 02                  DB      30H,0CFH,02         ;JNB T2CON.7,2
04F9  91 B5                     ACALL   TIMER
04FB  30 98 EF                  JNB     RI,READ1            ;RI?
04FE  C2 AD                     CLR     IEC.5               ;DISABLE TIMER 2 INTERRUPT
0500  E5 99                     MOV     A,SBUF              ;GET CHAR
0502  C2 98                     CLR     RI                  ;CLEAR INT FLAG
0504  A2 9A                     MOV     C,RB8               ;GET PARITY BIT
0506  A1 29                     AJMP    READ5               ;ESCAPE
0508  C2 AD              READ2:  CLR     IEC.5               ;DISABLE TIMER 2 INT
050A  C0 F0                     PUSH    B                   ;SAVE THESE REGS
050C  C0 00                     PUSH    00H
050E  F8                        MOV     R0,A                ;GET R0 = BUFFC
050F  86 F0                     MOV     B,@R0               ;B = CHAR
0511  24 08                     ADD     A,#8                ;POINT TO PARITY
0513  F8                        MOV     R0,A                ;INDEX IT
0514  05 59                     INC     BUFFC               ;POINT TO NEXT CHAR TO UNLOAD
0516  74 68                     MOV     A,#68H              ;END OF BUFFER
0518  B5 59 03                  CJNE    A,BUFFC,READ3
051B  75 59 60                  MOV     BUFFC,#60H          ;RESET TO START OF BUFFER
051E  E6                READ3:  MOV     A,@R0               ;GET PARITY
051F  C3                        CLR     C                   ;C = 0
0520  60 01                     JZ      READ4               ;IF A = 0
0522  D3                        SETB    C
0523  E5 F0              READ4:  MOV     A,B                 ;A = CHAR
0525  D0 00                     POP     00H
0527  D0 F0                     POP     B                   ;RESTORE REG
0529  43 A8 90          READ5:  ORL     IEC,#90H            ;ENABLE SERIAL PORT
052C  A1 87                     AJMP    LOST                ;ARE WE LOST?

052E  90 11 00          NUMBER: MOV     DPTR,#1100H         ;SET UP 8156 FOR DIRECT INPUT
0531  74 00                     MOV     A,#00H
0533  F0                        MOVX    @DPTR,A
0534  74 50                     MOV     A,#50H              ;TURN ON BUFFERS AND GOTO READ MODE
0536  90 00 01                  MOV     DPTR,#1
0539  F0                        MOVX    @DPTR,A
```

```
053A   C2 95                    CLR     ADDEN           ;REQUEST ADDRESS
053C   90 11 01                 MOV     DPTR,#1101H     ;POINT TO DATA PORT
053F   E0                       MOVX    A,@DPTR         ;GET DATA
0540   E0                       MOVX    A,@DPTR         ;AGAIN FOR GOOD MEASURE
0541   F5 49                    MOV     NUMB2,A         ;SAVE HERE
0543   A3                       INC     DPTR            ;POINT TO NEXT DATA
0544   E0                       MOVX    A,@DPTR         ;GET DATA
0545   E0                       MOVX    A,@DPTR         ;AGAIN FOR GOOD MEASURE
0546   F5 48                    MOV     NUMB1,A         ;SAVE HERE
0548   D2 95                    SETB    ADDEN           ;TURN OFF ADDRESS REQUEST
054A   15 82                    DEC     DPTR            ;SET 8156 TO OUTPUT MODE
054C   74 3B                    MOV     A,#3BH
054E   F0                       MOVX    @DPTR,A
054F   90 00 01                 MOV     DPTR,#1         ;TURN OFF BUFFER AND GOTO WRITE MODE
0552   74 00                    MOV     A,#0
0554   F0                       MOVX    @DPTR,A
0555   AB 48                    MOV     R3,NUMB1        ;SET UP FOR MONITOR
0557   AA 49                    MOV     R2,NUMB2
0559   E5 4F            NUM1:   MOV     A,BLOCK2        ;DETERMINE SIZE
055B   25 51                    ADD     A,COUNT2
055D   F5 82                    MOV     DPL,A
055F   E5 4E                    MOV     A,BLOCK1
0561   35 50                    ADDC    A,COUNT1
0563   F5 83                    MOV     DPH,A
0565   30 E7 07                 JNB     A.7,NUM2
0568   B4 80 1C                 CJNE    A,#80H,NUM3
056B   E5 82                    MOV     A,DPL
056D   70 18                    JNZ     NUM3
056F   85 82 53          NUM2:   MOV     EBASE2,DPL
0572   85 83 52                 MOV     EBASE1,DPH
0575   C3                       CLR     C
0576   E5 51                    MOV     A,COUNT2
0578   94 10                    SUBB    A,#16
057A   F5 55                    MOV     IBASE2,A
057C   E5 50                    MOV     A,COUNT1
057E   94 00                    SUBB    A,#0
0580   F5 54                    MOV     IBASE1,A
0582   C2 06                    CLR     BIT6
0584   75 58 4C                 MOV     TIME3,#4CH
0587                    NUM3:
0587   30 D5 03          LOST:   JNB     PSW.5,LOST1
058A   02 00 00                 LJMP    START
058D   D2 D5             LOST1:  SETB    PSW.5
058F   22                       RET

                                ;AND AT THE END OF THE EPROM LETS REENTER AT 0000H

07F0                            ORG     07F0H

07F0   00                       NOP
07F1   00                       NOP
07F2   00                       NOP
07F3   00                       NOP                     ;JUST A FEW NOPS TO PREPARE TO ENTER
07F4   02 00 00                 LJMP    START           ;NOW RE-ENTER

07F7                            END
```

********** S Y M B O L I C R E F E R E N C E T A B L E **********

ACKN	0422	ADDCK	045F	ADDEN	0000	ADDPE1	= 0040
ADDPE2	= 0041	BADSY1	= 0044	BADSY2	= 0045	BASE	= 7FF0
BIT0	0000	BIT1	0000	BIT2	0000	BIT3	0000
BIT4	0000	BIT5	0000	BIT6	0000	BIT7	0000
BIT8	0000	BIT9	0000	BLOCK1	= 004E	BLOCK2	= 004F
BUFFC	= 0059	BUFFN	= 005A	BUSY	0000	CMD	00A1
CMD0	00B1	CMD1	00C5	CMD2	00D1	CMD3	0129
CMD4	0141	CMD4A	0153	CMD4B	0155	CMD5	015F
CMDAD1	= 003C	CMDAD2	= 003D	CMDDT1	= 003E	CMDDT2	= 003F
CMDOU1	0106	CMDOU2	010E	COUL	0350	COUL1	035D
COUL2	036D	COUNT1	= 0050	COUNT2	= 0051	DATCO1	= 004A
DATCO2	= 004B	DATPE1	= 0042	DATPE2	= 0043	DATRP1	= 0036
DATRP2	= 0037	DC1	0426	DC2	041E	DOUT	0000
DSYNC	03AE	EBASE1	= 0052	EBASE2	= 0053	IBASE1	= 0054
IBASE2	= 0055	INCRE	040F	INTIAL	034C	LOAD0	019F
LOOP	0038	LOST	0587	LOST1	058D	MBIT1	03F9
MBIT2	03F9	MDTPE1	= 0046	MDTPE2	= 0047	MOD1	= 003A
MOD2	= 003B	MODEL	= 0044	MON	01A6	MON0	01EC
MON1	022F	MON2	023F	MON3	0248	MON4	02A6
MON5	02BB	MON6	02C3	MONC	0188	MONCO1	= 004C
MONCO2	= 004D	MONIN0	01F4	MONIN1	01FA	MONIN2	0202
MONIN3	0218	MONRP1	= 0038	MONRP2	= 0039	MSG	0000
NACK	041A	NOTUS	007B	NUM1	0559	NUM2	056F
NUM3	0587	NUMB1	= 0048	NUMB2	= 0049	NUMBER	052E
OUTIT	0408	P	0000	PARCI	0073	PARCI1	0097
PARX	0000	POINTS	= 0010	RB8	0000	RCVEN	0000
RCVEX	0000	READ	04E6	READ1	04ED	READ2	0508
READ3	051E	READ4	0523	READ5	0529	RES1	= 0030
RES2	= 0031	RES3	= 0032	RES4	= 0033	RES5	= 0034
RES6	= 0035	REV	= 0042	RI	0000	SERP	0483
SERP1	048A	SERP2	04A1	SERP3	04AC	SERP4	04B4
SPE01	0430	SPE02	0438	SPEOUT	0428	SPEXIT	0457
SPIN	03E8	SPIN1	03F0	SPO01	0448	SPO02	0450
SPOOUT	043F	START	0000	START1	0030	START2	0054
STAT1	02CB	STAT2	02D2	STAT3	02E5	STAT4	02EA
STAT4A	02F1	STAT5	02F5	STATUS	02C5	STCMD	030D
STCMD2	0314	STCMD3	031D	STCMD4	031F	STCMD5	032E
STCMD6	033B	STCMD7	0346	STCMD8	034A	STMON	02FE
STMON1	0307	SUBT	0479	SYNC	03B7	SYNC1	03BF
SYNC2	03D0	SYNC3	03E1	SYNC4	03E2	TB8	0000
TI	0000	TIME1	= 0056	TIME2	= 0057	TIME3	= 0058
TIMER	04B5	TIMER1	04C5	TIMER2	04CB	TIMER3	04D0
TIMER4	04D8	TR1	0000				

0000 ASSEMBLY ERRORS

4.0 STANDARD INTERFACE BOARD PHYSICAL DESCRIPTION

Figure 24 (below) is a perspective sketch of the VLBA Standard Interface Board. The board uses two-sided printed circuit artwork with two "D" series, pin contact connectors. A two-contact jumper plug is used to select the external Reset bus for board reset. Power requirements are described in Section 2.9.

The board envelope is 6.25 long x 5.50 wide x .70 thick (including I/O connectors). The board is shown in greater detail in the assembly drawings in Section 5.0.

The board I/O connectors are Cinch DD-50PC (P1, parallel I/O) and Cinch DB-25PC (P2, serial I/O). These mate with DD-50S and DB-25S connectors, respectively. All chips are installed in IC sockets or strip IC connectors.

The sketch shows the board mounted on two module rails - a typical technique. Three #4 clearance holes on the edges of the board permit screw-attachment to the rails. Another mounting technique is to spacer-mount the board on four #4 spacers attached to a module side panel. Four #4 mounting holes, (inside the rail space) are provided for spacer-mounting the board. A third technique, used in the M103 module, is to mount the board in plastic PC card guides attached to the module top and bottom screens. This mounting scheme is restricted to modules having widths of 2-wide or more.

When the board is not in use it should be placed in a conductive storage bag to protect the sensitive IC's and expensive A/D converter from static damage.

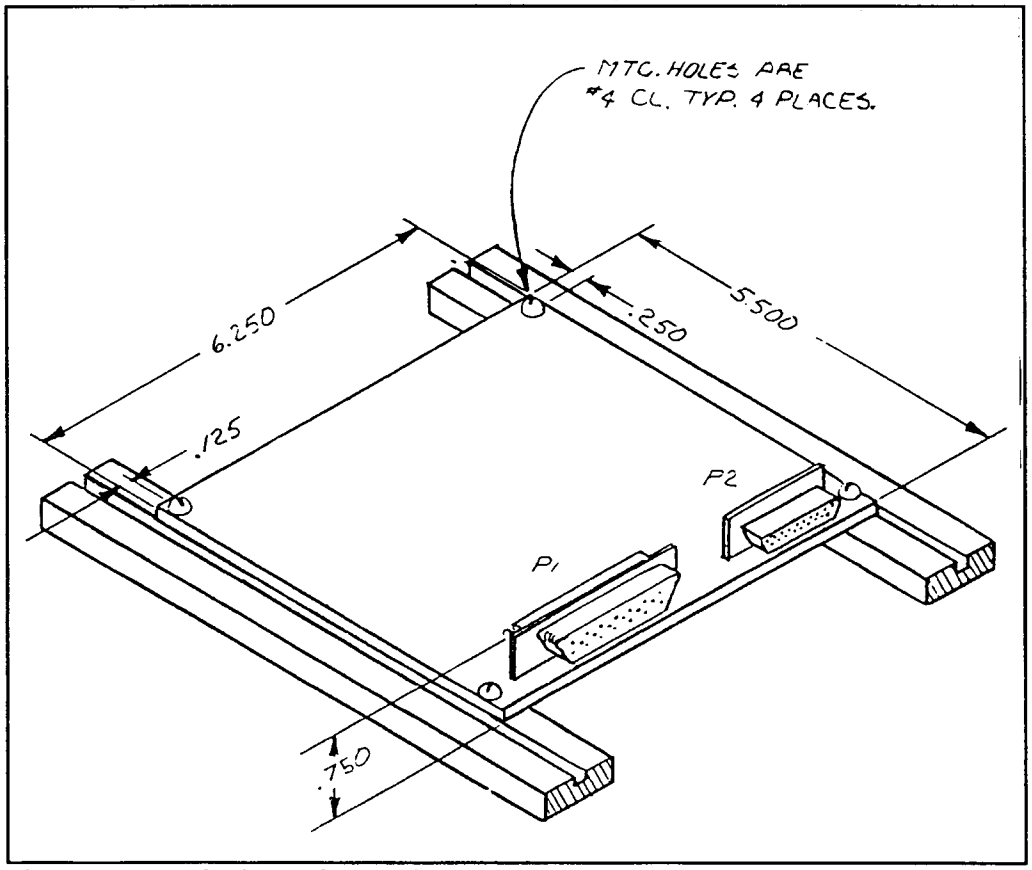


Figure 24, Standard Interface Package

5.0 LIST OF DRAWINGS AND DRAWINGS

Most of the VLBA Standard Interface drawings are included in this manual; they follow this list in the order shown.

Version "D"

D55002S002	Standard Interface Board Model "D" Schematic Diagram
D55002A002	Assembly Drawing, Model "D"
A55002B002	Assembly Bill of Materials, Model "D" (not included in this manual)
D55002Q002	Printed Circuit Artwork Master, Model "D"
D55002P002	PCB Drill Drawing, Model "D" (not included in this manual)

Version "S"

D55002S001	Standard Interface Board Model "S" Schematic Diagram
D55002A001	Assembly Drawing, Model "S"
A55002B001	Assembly Bill of Materials, Model "S" (not included in this manual)
D55002Q001	Printed Circuit Artwork Master, Model "S"
D55002P001	PCB Drill Drawing, Model "S" (not included in this manual)

Sample Device Interface Schematic

C55002S003	Standard Interface Board Sample Device Interface Schematic

Controller

C55001A015	VME - 705A Driver/Receiver Card Modifications (not included in this manual)

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
C	1-91	ANDREATTA		REDRAWN WITH ACAD ;

CPU ADDRESS/DATA BUS

CONTROL PROCESSOR
U1
INTEL P8032AH uCONT

EPROM PROGRAM MEMORY
U2
INTEL D8755A-2

RAM-I/O
U5
INTEL P8156H-2

U10 74LS245
U11 74LS245
CONTROL/MONITOR LINE BUFFERS

PARALLEL I/O
P1

SERIAL I/O
P2

DIFF DRIVER U8 75174

DIFF RECEIVER U9 75175

U12B 96LS02

U7A 74LS221

U7B 74LS221

U3 74LS173

U18 LS04

LS00 U15

LS10 U17

RELATIVE ADDR TO ADDR BUS DRIVER U4 (SHT 2)

STB/ACK FROM U18-7 (SHT 2)

12 BIT ANALOG CONV. FROM U14 (SHT 2)

R/W U13-1 (SHT 2)

IDR U17-4 (SHT 2)

INSTALL HAND WIRE LINK TO P1.7 OR DESIRED SIGNAL

EXTERNAL INDICATOR DRIVER

ACTIVITY INDICATORS (EXTERNAL)

(EXTERNAL) P/O USERS DEVICE

RESET LINK

R to D - FOR SYSTEM RESET INPUT (P2-5,6)
R to G - FOR EXTERNAL RESET INPUT (P2-23)

WARNING :
CPU U1 WILL NOT RUN IF LINK 'R' IS LEFT FLOATING. IT MUST BE TERMINATED TO LINK 'G' OR 'D'.

NOTES :

1. ALL IC's ARE 'LS' LOGIC UNLESS OTHERWISE INDICATED.
2. USE EXACT PART NUMBERS ONLY ON U1, U2, AND U5. DO NOT SUBSTITUTE.
3. BYPASS CAPACITORS (.01 'Cx' ON ASSEMBLY DWG.) NOT SHOWN.
4. * = NOT USED.
5. ALL RESISTOR VALUES ARE IN OHMS.

A55002B002	BOM
D55002A002	ASSEMBLY
D55002P002	DRILL DWG.
D55002Q002	ARTWORK
NEXT ASSEMBLY	DWG. TYPE

CPU, MEMORY, AND I/O DRIVERS

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ± —
3 PLACE DECIMALS (.000) ± —
2 PLACE DECIMALS (.00) ± —
1 PLACE DECIMALS (.0) ± —
MATERIAL : —
FINISH : —

VLBA PROJECT: STANDARD INTERFACE BOARD

TITLE: STANDARD INTERFACE BOARD MODEL 'D' SCHEMATIC DIAGRAM

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

DRAWN BY	HARDEN	DATE	12-85
DESIGNED BY	WEBER	DATE	5-84
APPROVED BY		DATE	
SHEET NUMBER	1 of 2	DRAWING NUMBER	D55002S002
REV.	C	SCALE	—

ACAD : SIBDSK-1

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
C	1-91	ANDREATTA		REDRAWN WITH ACAD ;

PARALLEL I/O P1

+5V 50 → +5V DISTR
5V COMM 34 → DIGITAL GND
+15V 40 → +15V DISTR
HIQ GND 38 → ANALOG GND
-15V 39 → -15V DISTR
DEV ACK 36 → DEVICE ACKNOWLEDGE
ANENB 37 → ANALOG ENABLE

R/W U16-12 (SHT 1)
IDR U2-39 (SHT 1)

+5V R7 1K
PU1
74LS74 U13 (PR, D, Q, CLR)
LS10 U17
LS00 U15
U16 LS04
U12A 96LS02 (R, C, Q, CLR)
R6 15K, C6 220pF
STRB/ACR → TO U1-12 CPU INTR-0 (SHT 1)

H = D0-D11 IS IN HI-Z STATE
L = D0-D11 DUMPED ON BUS

(25 uS) (500 uS)

S/H CONT (66) | BUSY (24) | STRB (48) | EN1 EN2 EN3 (54 56 53) OUTPUT D0-D11 | D11 EN (41) | SHORT CYCLE (26) | +5V (49) | -15V (64) | ANALOG COMM | +15V (65)

(L = CONVERSION IN PROGRESS)

U14
SDM854BG (BURR-BROWN)
ANALOG DATA ACQUISITION SYSTEM

CONFIGURED FOR
8-CHAN DIFFERENTIAL INPUTS
BIPOLAR OPERATION (±10V RANGE)
12-BIT RESOLUTION (4.8828 mV/BIT)
2's COMPLIMENT OUTPUT CODING
OUTPUT IS ONE 12-BIT WORD, HI-Z DRIVERS

DIFFERENTIAL ANALOG INPUTS
ANLG-0 H 1 / L 18 → CH1+ (11) / CH1- (10)
ANLG-1 H 2 / L 19 → CH2+ (18) / CH2- (71)
ANLG-2 H 3 / L 20 → CH3+ (9) / CH3- (70)
ANLG-3 H 4 / L 21 → CH4+ (8) / CH4- (72)
ANLG-4 H 5 / L 22 → CH5+ (7) / CH5- (74)
ANLG-5 H 6 / L 23 → CH6+ (6) / CH6- (73)
ANLG-6 H 7 / L 24 → CH7+ (5) / CH7- (76)
ANLG-7 H 8 / L 25 → CH8+ (4) / CH8- (77)

D0 (29 LSB) → CON/MON-4
D1 (30) → CON/MON-5
D2 (31) → CON/MON-6
D3 (32) → CON/MON-7
D4 (33) → CON/MON-8
D5 (34) → CON/MON-9
D6 (35) → CON/MON-10
D7 (36) → CON/MON-11
D8 (37) → CON/MON-12
D9 (38) → CON/MON-13
D10 (39) → CON/MON-14
D11 (40) / D11 (44 MSB) → CON/MON-15

CON/MON BUS (12-BIT ANALOG CONVERSION) TO P1 (SHT 1)

MUX EN1 (12)
MUX EN2 (13)
CLR (18) +5V
S/H OFFSET (67) ← +15V, RV3 1K, R9 24K → FINE OFFSET ADJ

A0 (19) A1 (20) A2 (21) A3 (22) DIG GND (25) | LOAD (23) | DLY OUT (45) TRIG (46) | DLY ADJ (47) | BPO (59) | 2.5V REF OUT (55) IN (57) | MUX OUT (78) (2) | 20V RANGE (58) | S/H OUT (62) | S/H IN (60)

250 uS → LOAD REG, TRIG A/D
(250 uS) C11 .01 uF
RV2 50 ZERO, RV1 100 GAIN
R10 240

+5V, 470 → HI/LO SEL (17)

U6 74LS157: 1Y 2Y 3Y 4Y (9 4 12 7), SEL (1), G (15), 1A 1B 2A 2B 3A 3B 4A 4B (11 10 2 3 14 13 5 6)
ADDR-0 ADDR-3 ADDR-1 ADDR-4 ADDR-2 ADDR-5

RG1 (1), RG2 (4), (5), (10), (8)
NC (2) (3) OFFSET (6) (7) (9)
-15V +15V

FOR 4.8828 mV/bit INSTALL JUMPER IN LIEU OF R10
FOR 5.0 mV/bit INSTALL 240 OHM VV RESISTOR PER BURR-BROWN SPECS

U19 BURR-BROWN INA101CM INSTR. AMPL.
U19 PIN OUT (TOP VIEW)

+5V C10 1 uF, C9 .01 uF
-15V C15 1 uF, C14 .01 uF
ANALOG GND +15V C13 1 uF, C12 .01 uF

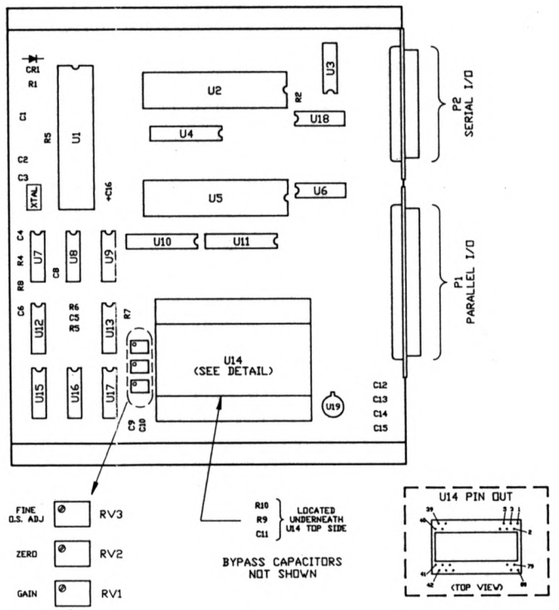
+5V 74LS245 ADDR BUS DRIVER
U4: A0-A7 (9 8 7 6 5 4 3 2), B0-B7 (11 12 13 14 15 16 17 18), DIR (1), G (19), (20), (10)
FROM U2 (SHT 1) → RA0-RA7
PU2
ADDRESS BUS

PARELLEL I/O P1
ADDR-0 (48) → RA-0
ADDR-1 (47) → RA-1
ADDR-2 (46) → RA-2
ADDR-3 (45) → RA-3
ADDR-4 (44) → RA-4
ADDR-5 (43) → RA-5
ADDR-6 (42) → RA-6
ADDR-7 (41) → RA-7

SERIAL I/O P2

+5V DISTR → 1, 14 → +5V
DIGITAL GND → 13, 25 → LOGIC COMM
+15V DISTR → 15 → +15V
ANALOG GND → 17 → ±15V COMM
-15V DISTR → 16 → -15V

EXTERNAL ACCESSORY DC POWER AVAILABLE FOR TEST AND ALIGNMENT PURPOSES



FINE O.S. ADJ RV3
ZERO RV2
GAIN RV1

R10, R9, C11 LOCATED UNDERNEATH U14 TOP SIDE
BYPASS CAPACITORS NOT SHOWN

U14 PIN OUT (TOP VIEW)

ANALOG CIRCUITRY

A55002B002	BOM
D55002A002	ASSEMBLY
D55002P002	DRILL DWG.
D55002Q002	ARTWORK
NEXT ASSEMBLY	DWG. TYPE

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ± —
3 PLACE DECIMALS (.000) ± —
2 PLACE DECIMALS (.00) ± —
1 PLACE DECIMALS (.X) ± —
MATERIAL : —
FINISH : —

VLBA STANDARD INTERFACE BOARD

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

TITLE: STANDARD INTERFACE BOARD MODEL 'D' SCHEMATIC DIAGRAM

DRAWN BY	DATE
HARDEN	12-85
DESIGNED BY	DATE
WEBER	5-84
APPROVED BY	DATE

SHEET NUMBER 2 of 2 | DRAWING NUMBER D55002S002 | REV. C | SCALE —

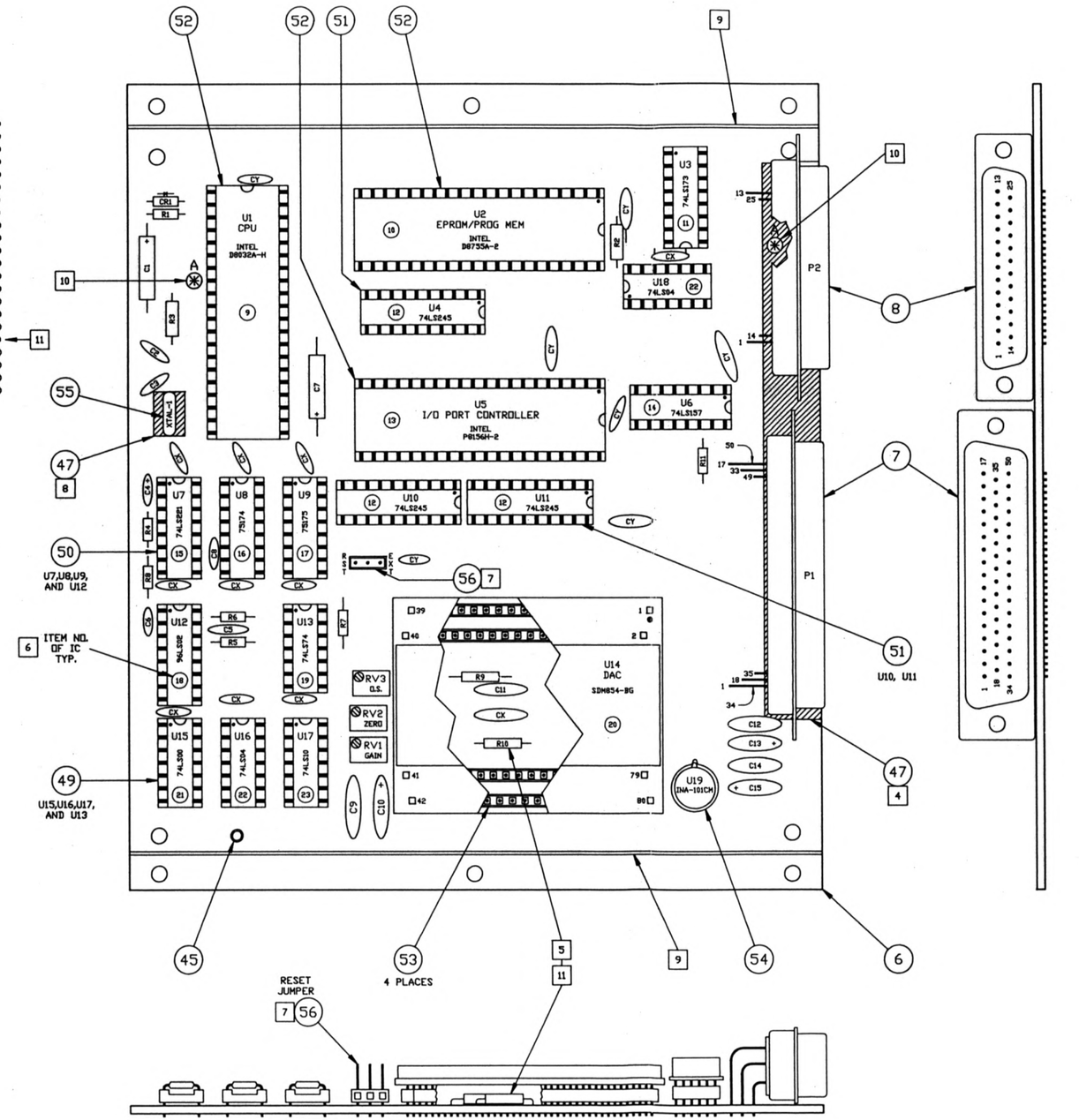
ACAD : SIBDSK-2

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
C	1-91	ANDREATTA		REDRAWN WITH ACAD ;

DISCRETE COMPONENT TABLE

REFERENCE BOM
DWG. #A55002B002

___C1	___C7		22uF	(ITEM #25)
___C2	___C3		30pF	(ITEM #26)
___C4	___C5		2.2uF	(ITEM #27)
___C6			220pF	(ITEM #28)
___C8	___C9		.1uF	(ITEM #29)
___C10	___C13	___C15	1uF	(ITEM #30)
___C11	___C12	___C14	.01uF	(ITEM #31)
___*CX*			.01uF	(ITEM #31)
___*CY*			.1uF	(ITEM #29)
___R1			4.7K OHM	(ITEM #32)
___R2	___R7		1K OHM	(ITEM #33)
___R3	___R11		470 OHM	(ITEM #34)
___R4			51K OHM	(ITEM #35)
___R5			39K OHM	(ITEM #36)
___R6			15K OHM	(ITEM #37)
___R8			12K OHM	(ITEM #38)
___R9			24K OHM	(ITEM #39)
___R10			240 OHM	(ITEM #40)
___RV1			100 OHM POT	(ITEM #42)
___RV2			50 OHM POT	(ITEM #43)
___RV3			1K OHM POT	(ITEM #44)
___CR1			1N914	(ITEM #46)



NOTES :

1. THIS BOARD, MODEL 'D' (REV. 'C'), SUPERCEDES VLBA STANDARD INTERFACE BOARD D550029002 (REV. 'B').
2. MODEL 'D' BOARDS ARE FOR DIFFERENTIAL ANALOG HANDLING. MODEL 'S' BOARDS ARE SINGLE ENDED VERSIONS.
3. PCB IS ETCHED FROM 2-OUNCE COPPER BOARDS.

4 PLACE KAPTON 1KV DIELECTRIC TAPE (ITEM #47) ALONG COMPONENT SIDE AS SHOWN BEFORE CONNECTOR INSTALLATION FOR ISOLATION BETWEEN POWER TRACES AND CONNECTOR SHELL.

5 ITEMS #31, #40, AND #41 (R9, R10, AND C11) ARE MOUNTED ON COMPONENT SIDE COVERED (HIDDEN) BY U14 WHEN INSTALLED.

6 ALL IC'S ARE SOCKETED. NOTE ORIENTATIONS.

7 USER MUST PROVIDE DESIRED CPU/BUS RESET CODING BY USING MIDGIE FEMALE JUMPER HEADER (ITEM #57).
LINK RST-TO-GND FOR EXTERNAL SWITCH RESET VIA P2-23
LINK RST-TO-EXT DRVR FOR EXTERNAL BUS RESET (SYSTEM RESET) VIA P2-5 (RST+) AND P2-6 (RST-)

WARNING → POWER-ON RESET ENABLED IN BOTH CODING MODES. NO LINK INHIBITS ALL RESETS INCLUDING POWER-ON RESET.

8 PLACE KAPTON 1KV DIAELECTRIC TAPE (ITEM #47) UNDER CRYSTAL TO ISOLATE TRACES FROM METAL CRYSTAL ENCLOSURE.

9 BOARD IS DIMENSIONED FOR MOUNTING ON STANDARD NRAO MODULE RAILS. TOP AND BOTTOM EDGES MAY BE TRIMMED FOR OTHER APPLICATIONS. DO NOT EXCEED TRIM MARKS OR DAMAGE TO TRACES WILL RESULT.

10 TWO PLATED THRU VIA'S MATKED 'A' ARE USED TO :
a.) WHEN CONNECTED TOGETHER BY SOLDERING WIRE, PLACES CPU PORT TERM P1.7 INTO P2-7 FOR USER APPLICATION
b) 'A' BY P2 CAN BE USED TO ADD ANY DESIRED SIGNAL TO P2-7 FOR USER APPLICATION OR TESTING. (P2-7 IS A SPARE, UNCOMMITTED PIN FOR SUCH USE)

11 A PIECE OF YELLOW WIRE WRAP WIRE (ITEM #48) SHALL BE USED INSTEAD OF R10 (ITEM #40).

ASSEMBLY STEPS :

- ✓ 1. CLEAN AND INSPECT BOARD FOR ANY BAD TRACES, PLUGGED HOLES, ETC...
- ☐ 2. TRIM BOARD TO DESIRED SIZE AND ENSURE MOUNTING HOLES TO BE USED ARE DRILLED BEFORE ASSEMBLY.
- ☐ 3. INSTALL AND SOLDER IC SOCKETS. (NOTE ORIENTATION)

14 PIN	___U13 ___U15 ___U16 ___U17 ___U18	(ITEM #49)	5 REQ
16 PIN	___U3 ___U6 ___U7 ___U8 ___U9 ___U12	(ITEM #50)	6 REQ
20 PIN	___U4 ___U10 ___U11	(ITEM #51)	3 REQ
40 PIN	___U1 ___U2 ___U5	(ITEM #52)	3 REQ
20 PIN	___STRIP SOCKETS TO MAKE SOCKET FOR U14	(ITEM #53)	4 REQ
10 PIN	___TO-5 SOCKET FOR U19	(ITEM #54)	1 REQ

- ☐ 4. PLACE KAPTON TAPE (ITEM #47) WHERE REQ'D. (SEE NOTES #4 AND #8)
- ☐ 5. INSTALL AND SOLDER CONNECTORS P1 AND P2 (ITEMS #7 AND #8)
- ☐ 6. INSTALL AND SOLDER THE FOLLOWING :
  ___XTAL-1 ___RV1 ___RV2 ___RV3
- ☐ 7. INSTALL AND SOLDER DISCRETE COMPONENTS USING DISCRETE COMPONENT TABLE.
- ☐ 8. INSTALL RESET JUMPER FOR DESIRED MODE. (SEE NOTE #7)
- ☐ 9. CLEAN BOARD TO REMOVE SOLDER FLUX AND INSPECT FOR COLD SOLDER JOINTS, SPLASHES BETWEEN TRACES, ETC...
- ☐ 10. INSTALL IC'S INTO SOCKETS. (NOTE ORIENTATION)
- ☐ 11. WRITE SERIAL NUMBER ON BOARD.

A55001N001 = VLBA MONITOR AND CONTROL BUS SPEC.
A55001N002 = VLBA MONITOR AND CONTROL STANDARD INTERFACE SPEC.

BOM = DWG. #A55002B002

D55002S002	SCHEMATIC
D55002P002	DRILL DWG.
D55002Q002	ARTWORK
NEXT ASSEMBLY	DWG. TYPE

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ± —— 3 PLACE DECIMALS (.000) ± —— 2 PLACE DECIMALS (.00) ± —— 1 PLACE DECIMALS (.0) ± ——
MATERIAL : ——
FINISH : ——

VLBA PROJECT : STANDARD INTERFACE BOARD

TITLE : STANDARD INTERFACE BOARD MODEL 'D' ASSEMBLY

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

DRAWN BY	HARDEN	DATE 12-85
DESIGNED BY	WEBER	DATE 5-84
APPROVED BY		DATE

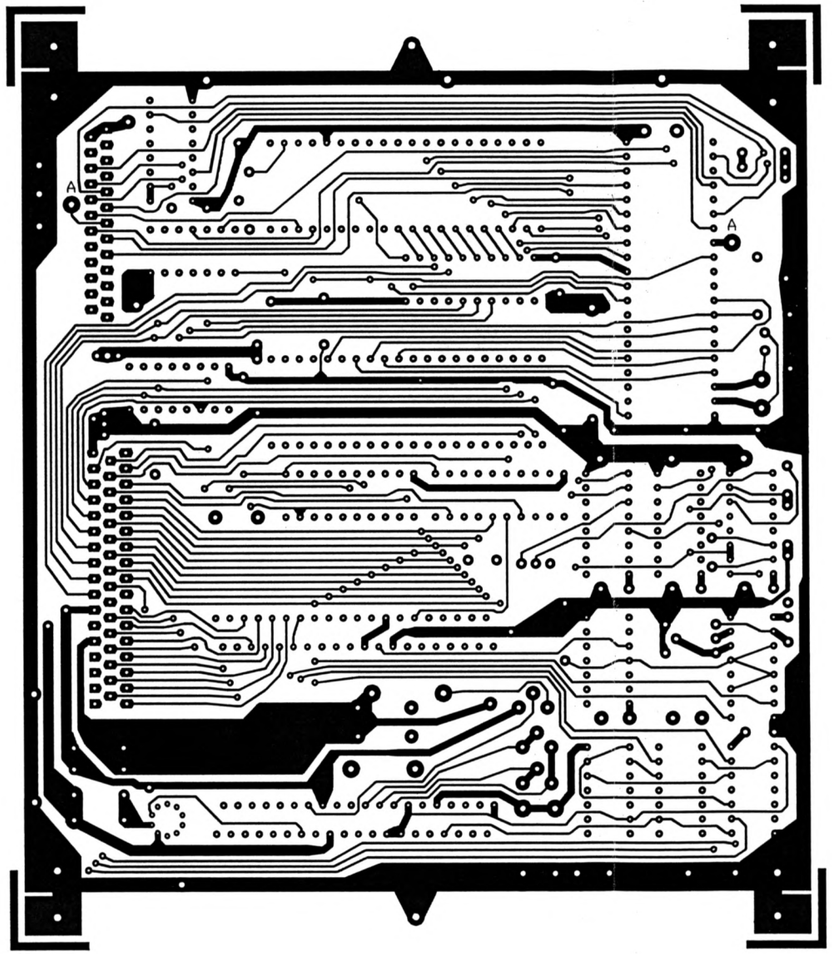
SHEET NUMBER 1 of 1 | DRAWING NUMBER D55002A002 | REV. C | SCALE 2/1

ACAD : SIBDASSY
REF : VLBA-14

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
C	1-91	ANDREATTA		REDRAWN WITH ACAD ;

REDUCE TO
6.500

REDUCE TO
7.350



CIRCUIT SIDE
SCALE 2/1

CIRCUIT SIDE

D55002S002	SCHEMATIC
A55002B002	BOM
D55002A002	ASSEMBLY
D55002P002	DRILL DWG.
NEXT ASSEMBLY	DWG. TYPE

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ± ——
3 PLACE DECIMALS (.XXX) ± .005
2 PLACE DECIMALS (.XX) ± ——
1 PLACE DECIMALS (.X) ± ——
MATERIAL : ——
FINISH : ——

VLBA PROJECT: STANDARD INTERFACE BOARD

TITLE: STANDARD INTERFACE BOARD MODEL 'D' ARTWORK

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

DRAWN BY HARDEN	DATE 4-85
DESIGNED BY KOSKI	DATE 4-85
APPROVED BY	DATE

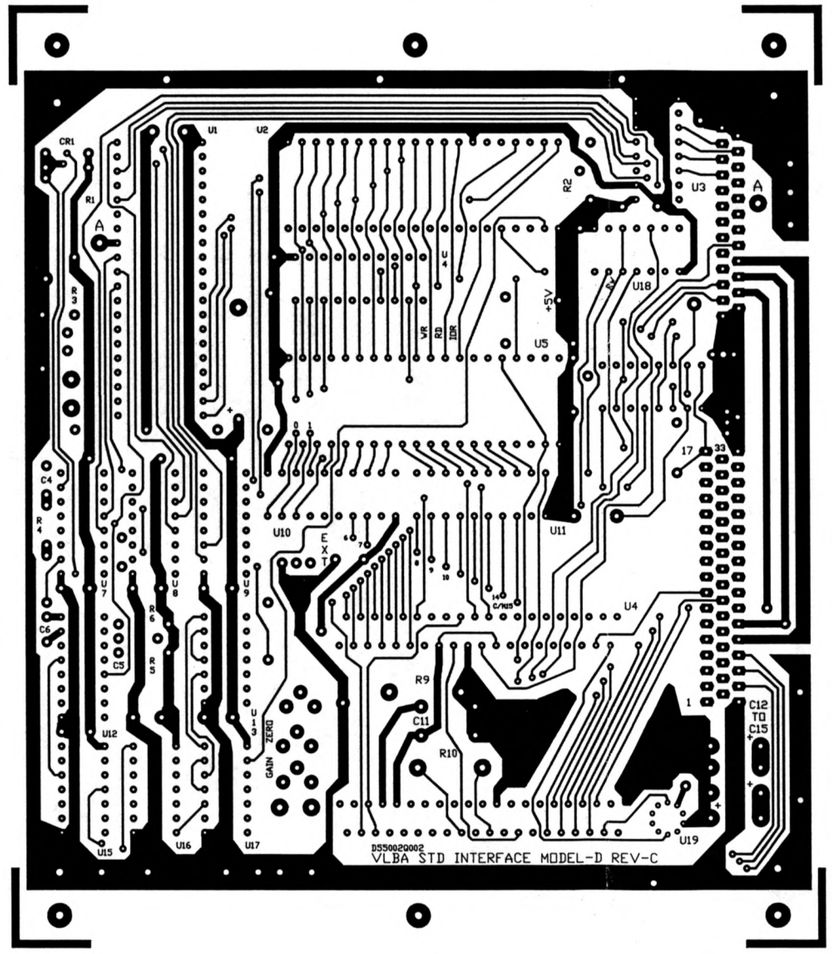
SHEET NUMBER 2 of 2 | DRAWING NUMBER D55002Q002 | REV. C | SCALE 2/1

ACAD : SIBDAW-2
REF : VLBA-14

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
C	1-91	ANDREATTA		REDRAWN WITH ACAD ;

REDUCE TO
6.500

REDUCE TO
7.350



COMPONENT SIDE
SCALE 2/1

COMPONENT SIDE

D55002S002	SCHEMATIC
A55002B002	BOM
D55002A002	ASSEMBLY
D55002P002	DRILL DWG.
NEXT ASSEMBLY	DWG. TYPE

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ± —
3 PLACE DECIMALS (.000) ± .005
2 PLACE DECIMALS (.00) ± —
1 PLACE DECIMALS (.X) ± —
MATERIAL : —
FINISH : —

VLBA CONTROL STANDARD INTERFACE BOARD

TITLE: STANDARD INTERFACE BOARD MODEL 'D' ARTWORK

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

DRAWN BY HARDEN	DATE 4-85
DESIGNED BY KOSKI	DATE 4-85
APPROVED BY	DATE

SHEET NUMBER 1 of 2 | DRAWING NUMBER D55002Q002 | REV. C | SCALE 2/1

ACAD : SIBDAW-1
REF : VLBA-14

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
D	2-91	ANDREATTA		REDRAWN WITH ACAD ;

CPU ADDRESS/DATA BUS

CONTROL/MONITOR LINE BUFFERS

U6 INTEL P8032AH uCONT (CONTROL PROCESSOR)

U11 INTEL D8755A-2 (EPROM PROGRAM MEMORY)

U12 INTEL P8156H-2 (RAM-I/O)

U15 74LS245

U14 74LS245

SERIAL I/O P2

PARALLEL I/O P1

DIFF DRIVER U3 75174

DIFF RECEIVER U2 75175

U1A 74LS221

U1B 74LS221

U19 74LS173

U9B 96LS02

U10 LS04

LS00 U13

RELATIVE ADDR TO ADDR BUS DRIVER U17 (SHT 2)

12 BIT ANALOG CONV. FROM U18, U20 (SHT 2)

IOR TO U7-2 (SHT 2)

STB/ACK FROM U9-7 (SHT 2)

PU1 TO U9-3 (SHT 2)

RESET LINK

R to D - FOR SYSTEM RESET INPUT (P2-5,6)
R to G - FOR EXTERNAL RESET INPUT (P2-23)

WARNING :
CPU U1 WILL NOT RUN IF LINK 'R' IS LEFT FLOATING. IT MUST BE TERMINATED TO LINK 'G' OR 'D'.

EXTERNAL INDICATOR DRIVER

NOTES :

1. ALL IC's ARE "LS" LOGIC UNLESS OTHERWISE INDICATED.
2. USE EXACT PART NUMBERS ONLY ON U6, U11, AND U12. DO NOT SUBSTITUTE.
3. BYPASS CAPACITORS NOT SHOWN.
4. * = NOT USED.
5. ALL RESISTOR VALUES ARE IN OHMS.

ACAD : SIBSSK-1

A55002B001	BOM
D55002A001	ASSEMBLY
D55002P001	DRILL DWG.
D55002Q001	ARTWORK
NEXT ASSEMBLY	DWG. TYPE

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ± —
3 PLACE DECIMALS (.XXX) ± —
2 PLACE DECIMALS (.XX) ± —
1 PLACE DECIMALS (.X) ± —
MATERIAL : —
FINISH : —

CPU, MEMORY, AND I/O DRIVERS

VLBA PROJECT : STANDARD INTERFACE BOARD

TITLE : STANDARD INTERFACE BOARD MODEL 'S' SCHEMATIC DIAGRAM

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

DRAWN BY	KOSKI
DATE	5-85
DESIGNED BY	WEBER
DATE	5-84
APPROVED BY	
DATE	
SHEET NUMBER	1 of 2
DRAWING NUMBER	D55002S001
REV.	D
SCALE	—

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
D	2-91	ANDREATTA		REDRAWN WITH ACAD ;

PARALLEL I/O P1

+5V — 50 — +5V DISTR
5V COMM — 34 — DIGITAL GND
+15V — 40 — +15V DISTR
HIQ GND — 38 — ANALOG GND
-15V — 39
ANENB — 37
DEV ACK — 36

U4A 96LS02 — R2 30K, C2 220pF — 2uS
U4B 96LS02 — R6 30K, C10 220pF — 2uS
U5A 96LS02 — R3 36K, C3 .022uF — 250uS
U5B 96LS02 — R7 15K, C11 220pF — 1uS
U8A, U8B 74LS74 (PRE D Q CLR)
74LS08 U7
R10 470 to +5V
U9A 96LS02 — R11 15K, C13 220pF — 1uS
STB/ACK TO U6-12 (SHT 1) 7/C
PU1 FROM U19-13 (SHT 1) 5/A
IDR FROM U11-39 (SHT 1) 5/B
U10 74LS04

GND, VLOGIC +5V, 2uS, R/C, STS, A0

DIFFERENTIAL ANALOG INPUTS

	H	L	
ANLG-0	1	18	CH-1
ANLG-1	2	19	CH-2
ANLG-2	3	20	CH-3
ANLG-3	4	21	CH-4
ANLG-4	5	22	CH-5
ANLG-5	6	23	CH-6
ANLG-6	7	24	CH-7
ANLG-7	8	25	CH-8

U18
HS-9412J
8 CHANNEL, 12-BIT DATA ACQUISITION SYSTEM

MA0 MA1 MA2, GAIN, OFFSET, VCC, VEE

HI/LO SEL — 17
U16 74LS157 (SEL, 4A 4B 3A 3B 2A 2B 1A 1B, 4Y 3Y 2Y 1Y, G)
R15 470 to +5V
R16 6M (GAIN)
R17 1.5M (OFFSET)
R18 20K POT, R19 20K POT
+15V (THIS SHEET) 1/C
-15V (THIS SHEET) 1/B

U10 74LS04
U20 74LS374 (CLK, OE, 7D 8D 6D 5D 1D 2D 3D 4D, 7Q 8Q 6Q 5Q 1Q 2Q 3Q 4Q)
DB11/DB3, DB10/DB2, DB9/DB1, DB8/DB0, DB7, DB6, DB5, DB4
CON/MON-15, CON/MON-14, CON/MON-13, CON/MON-12, CON/MON-11, CON/MON-10, CON/MON-9, CON/MON-8, CON/MON-7, CON/MON-6, CON/MON-5, CON/MON-4
12 BIT ANALOG CONV TO P1 (SHT 1) 5/C

PARALLEL I/O P1 — SERIAL I/O P2

+15V — 40 — 15 — +15V
HIQ GND — 38 — 17 — HIQ GND
-15V — 39 — 16 — -15V
U18 (VCC, GND, VEE, VLOGIC)
C36 10uF TANT, C37 1uF, C33 1uF, C35 10uF TANT
+15V (THIS SHT) 5/B

PARALLEL I/O P1

+5V — 50 — +5V
5V COMM — 34 — 5V COMM
U1 U2 U3 U4 U5 U6 U7 U8 U9 U10 U11 U12 U13
C4 .01uF, C5 .01uF, C6 .01uF, C7 .01uF, C8 .01uF, C14 1uF, C15 .01uF, C16 .01uF, C17 .01uF, C21 .01uF, C22 1uF, C23 1uF, C24 .01uF

SERIAL I/O P2

+5V DISTR
+5V — 1,14 — +5V
5V COMM — 13,25 — 5V COMM
U20 U19 U18 U17 U16 U15 U14
C34 22uF, C32 1uF, C31 .01uF, C30 10uF TANT, C29 1uF, C28 1uF, C27 .01uF, C26 1uF, C25 1uF

RA0 RA1 RA2 RA3 RA4 RA5 RA6 RA7
FROM U11 (SHT 1) 4/D
R14 470 to +5V
74LS245 ADDR BUS DRIVER
U17 (A0 A1 A2 A3 A4 A5 A6 A7 DIR, B0 B1 B2 B3 B4 B5 B6 B7, G)
ADDR-0 ADDR-1 ADDR-2 ADDR-3 ADDR-4 ADDR-5 ADDR-6 ADDR-7
PARELLEL I/O P1
RA-0 — 48
RA-1 — 47
RA-2 — 46
RA-3 — 45
RA-4 — 44
RA-5 — 43
RA-6 — 42
RA-7 — 41

ACAD : SIBSSK-2

ANALOG CIRCUITRY

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ±
3 PLACE DECIMALS (.000) ±
2 PLACE DECIMALS (.00) ±
1 PLACE DECIMALS (.X) ±
MATERIAL :
FINISH :

A55002B001	BOM
D55002A001	ASSEMBLY
D55002P001	DRILL DWG.
D55002Q001	ARTWORK
NEXT ASSEMBLY	DWG. TYPE

VLBA PROJECT: STANDARD INTERFACE BOARD

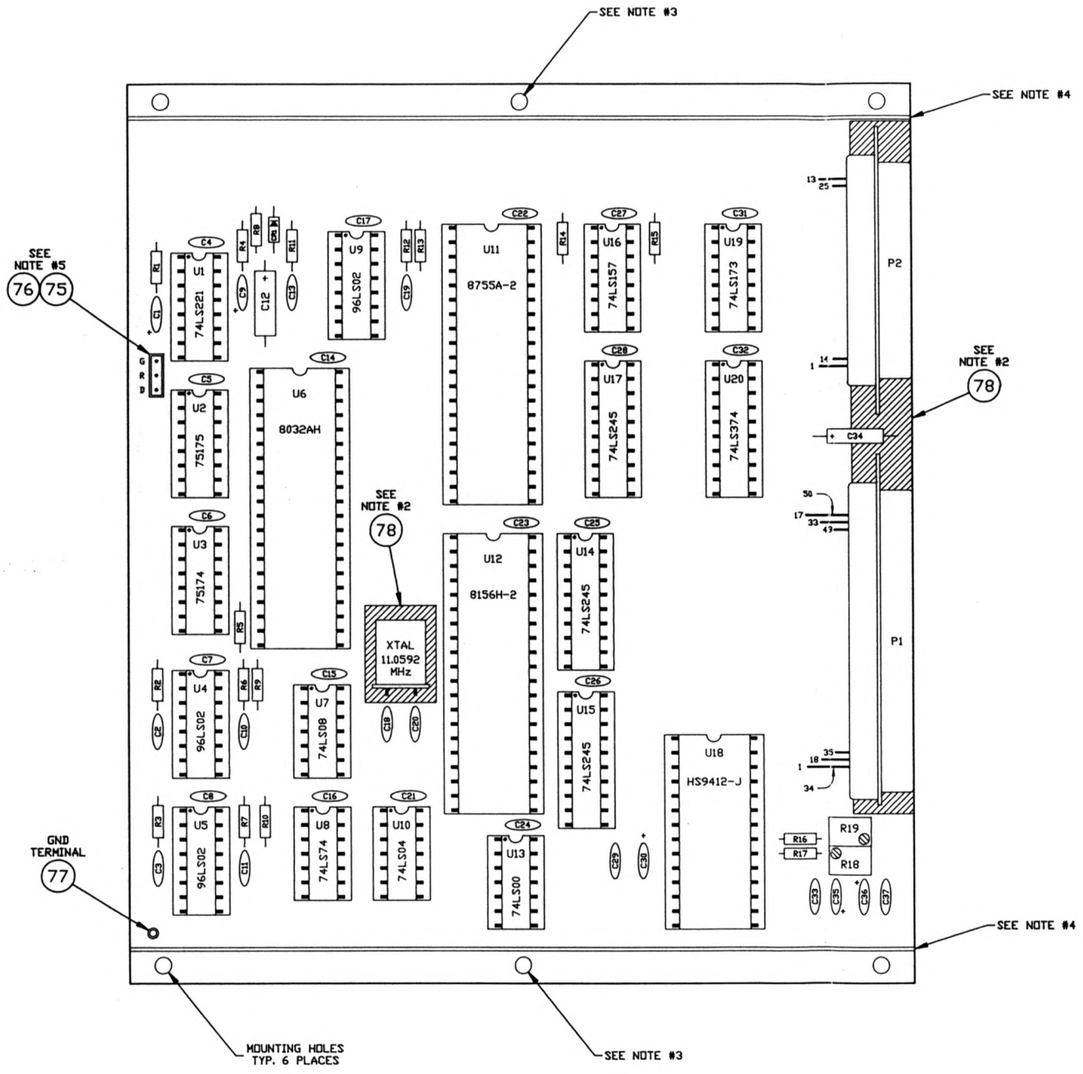
TITLE: STANDARD INTERFACE BOARD MODEL 'S' SCHEMATIC DIAGRAM

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

DRAWN BY KOSKI	DATE 5-85
DESIGNED BY WEBER	DATE 5-84
APPROVED BY	DATE

SHEET NUMBER 2 of 2 | DRAWING NUMBER D55002S001 | REV. D | SCALE —

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
A	4-91	ANDREATTA		REDRAWN WITH ACAD ;



NOTES :

1. ALL IC's ARE SOCKETED. SEE BOM ITEMS #8 THRU #12.
2. PLACE KAPTON TAPE ON COMPONENT SIDE OF BOARD UNDER CONNECTORS P1 AND P2 AND THE XTAL BEFORE INSTALLLATION.
3. CENTER MOUNTING HOLE MAY BE USED IF USER DESIRES ADDED STRENGTH.
4. BOARD IS DESIGNED FOR MOUNTING ON THE RAILS OF STANDARD NRAO MODULES. THE BOARD MAY BE TRIMMED FOR MOLUNTING INSIDE THE RAILS OR OTHER SCHEMES BY CUTTING AT THE ALTERNATE TRIM LINES ETCHED ON THE BOARD. DO NOT EXCEED THIS TRIM LINE AS GROUND DISTRIBUTION WILL BE AFFECTED.
5. JUMP BETWEEN G-R OR R-D USING MIDGIE FEMALE JUMPER (ITEM #76) ON MALE HEADER (ITEM #75). SEE SCHEMATIC FOR DETAILS.

BOM = DWG #A55002B001

D55002S001	SCHEMATIC
D55002P001	DRILL DWG.
D55002Q001	ARTWORK
NEXT ASSEMBLY	DWG. TYPE

UNLESS OTHERMISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ± ——
3 PLACE DECIMALS (.XXX) ± ——
2 PLACE DECIMALS (.XX) ± ——
1 PLACE DECIMALS (.X) ± ——
MATERIAL : ——
FINISH : ——

VLBA ELECTRONICS

STANDARD INTERFACE BOARD

TITLE: STANDARD INTERFACE BOARD MODEL 'S' ASSEMBLY

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

DRAWN BY KOSKI	DATE 8-85
DESIGNED BY KOSKI	DATE 8-85
APPROVED BY	DATE

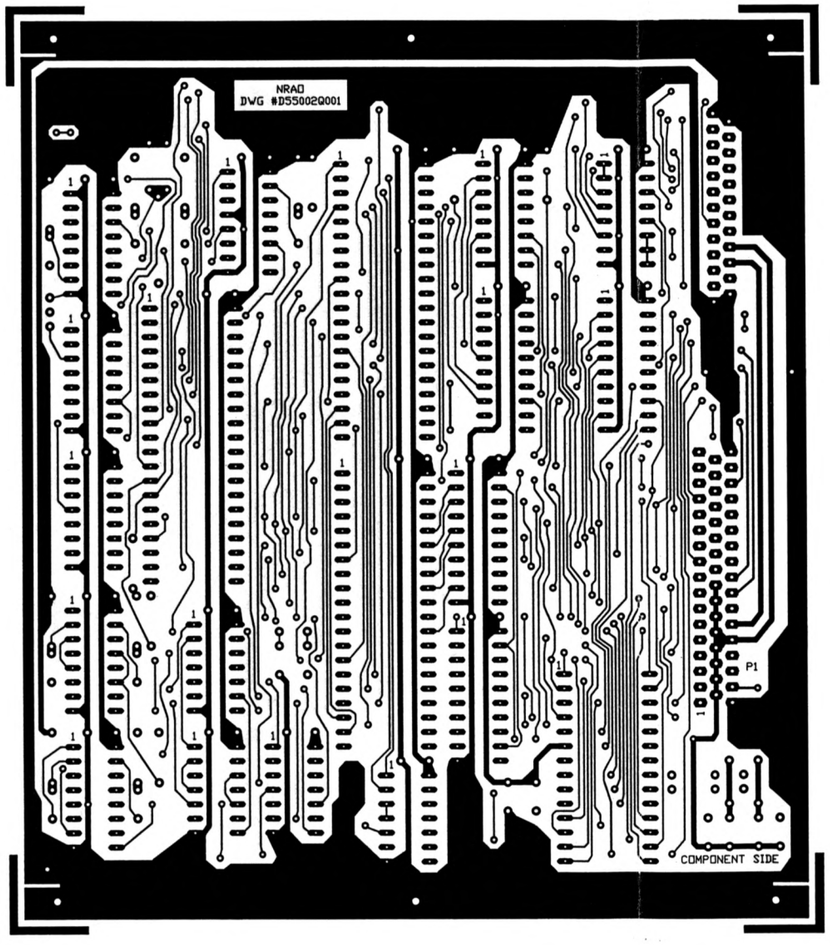
SHEET NUMBER 1 of 1 | DRAWING NUMBER D55002A001 | REV. A | SCALE 2/1

ACAD : SIBSASSY
REF : VLBA-15

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
C	4-91	ANDREATTA		REDRAWN WITH ACAD ;

REDUCE TO
6.500

REDUCE TO
7.250



COMPONENT SIDE
SCALE 2/1

COMPONENT SIDE

D55002S001	SCHEMATIC
A55002B001	BOM
D55002A001	ASSEMBLY
D55002P001	DRILL DWG.
NEXT ASSEMBLY	DWG. TYPE

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ±
3 PLACE DECIMALS (.000) ± .005
2 PLACE DECIMALS (.00) ±
1 PLACE DECIMALS (.0) ±
MATERIAL :
FINISH :

VLBA PROJECT: STANDARD INTERFACE BOARD

TITLE: STANDARD INTERFACE BOARD MODEL 'S' ARTWORK

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

DRAWN BY KOSKI	DATE 8-85
DESIGNED BY KOSKI	DATE 8-85
APPROVED BY	DATE

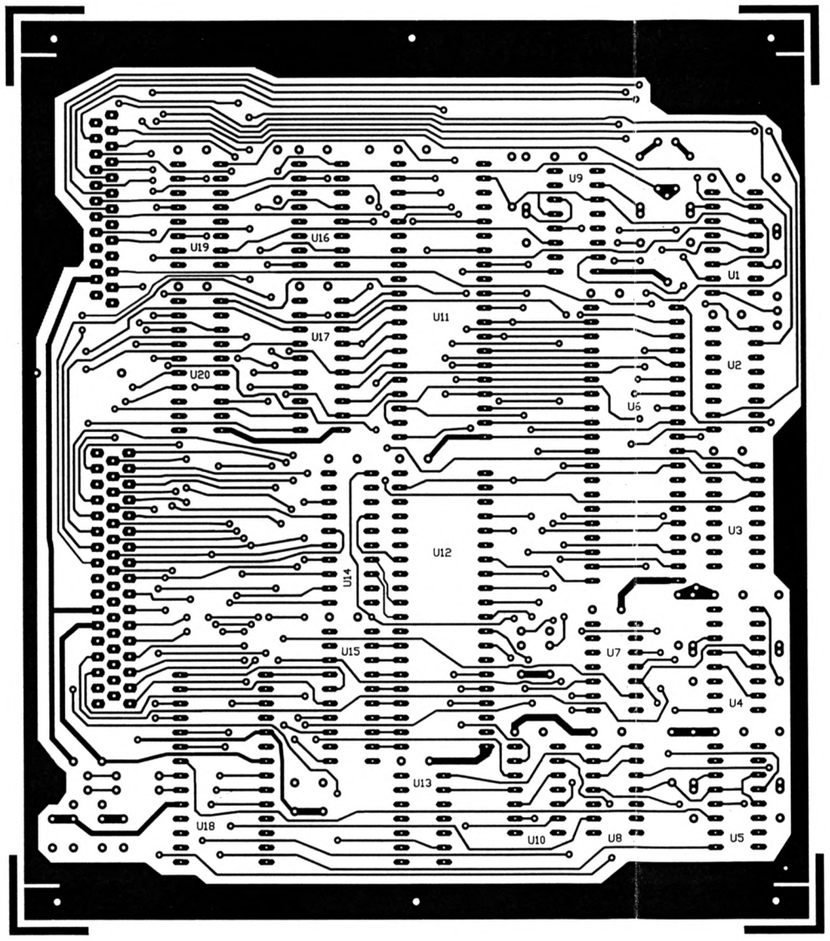
SHEET NUMBER 1 of 2 | DRAWING NUMBER D55002Q001 | REV. C | SCALE 2/1

ACAD : SIBSAW-1
REF : VLBA-15

REV	DATE	DRAWN BY	APPRV'D BY	DESCRIPTION
C	4-91	ANDREATTA		REDRAWN WITH ACAD ;

REDUCE TO
6.500

REDUCE TO
7.250



CIRCUIT SIDE
SCALE 2/1

CIRCUIT SIDE

D55002S001	SCHEMATIC
A55002B001	BOM
D55002A001	ASSEMBLY
D55002P001	DRILL DWG.
NEXT ASSEMBLY	DWG. TYPE

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ± —
3 PLACE DECIMALS (.XXX) ± .005
2 PLACE DECIMALS (.XX) ± —
1 PLACE DECIMALS (.X) ± —
MATERIAL : —
FINISH : —

VLBA SUBSYSTEM
STANDARD INTERFACE BOARD
STANDARD INTERFACE BOARD MODEL 'S' ARTWORK

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

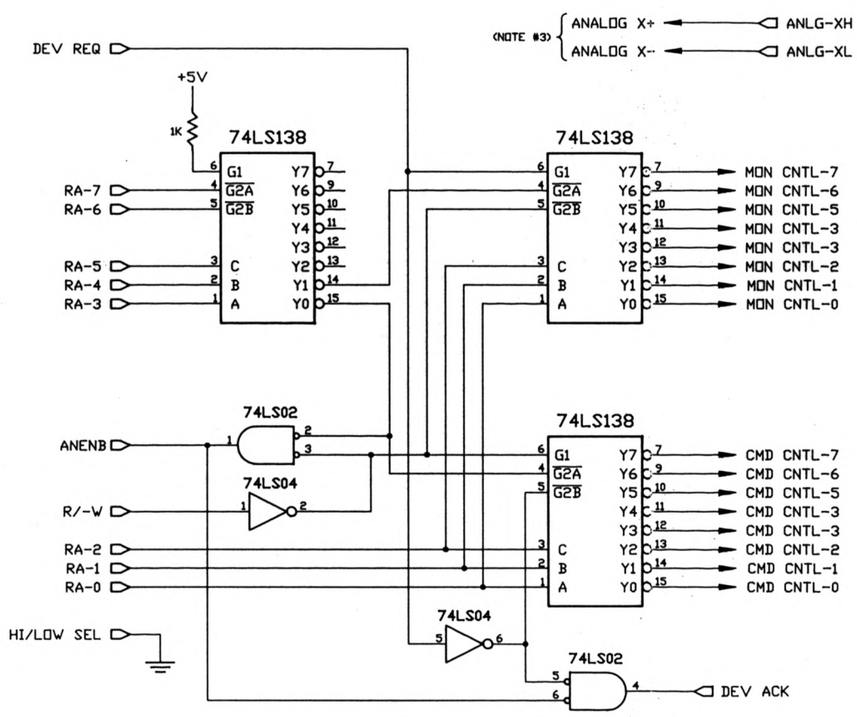
DRAWN BY KOSKI	DATE 8-85
DESIGNED BY KOSKI	DATE 8-85
APPROVED BY	DATE

SHEET NUMBER 2 of 2 | DRAWING NUMBER D55002Q001 | REV. C | SCALE 2/1

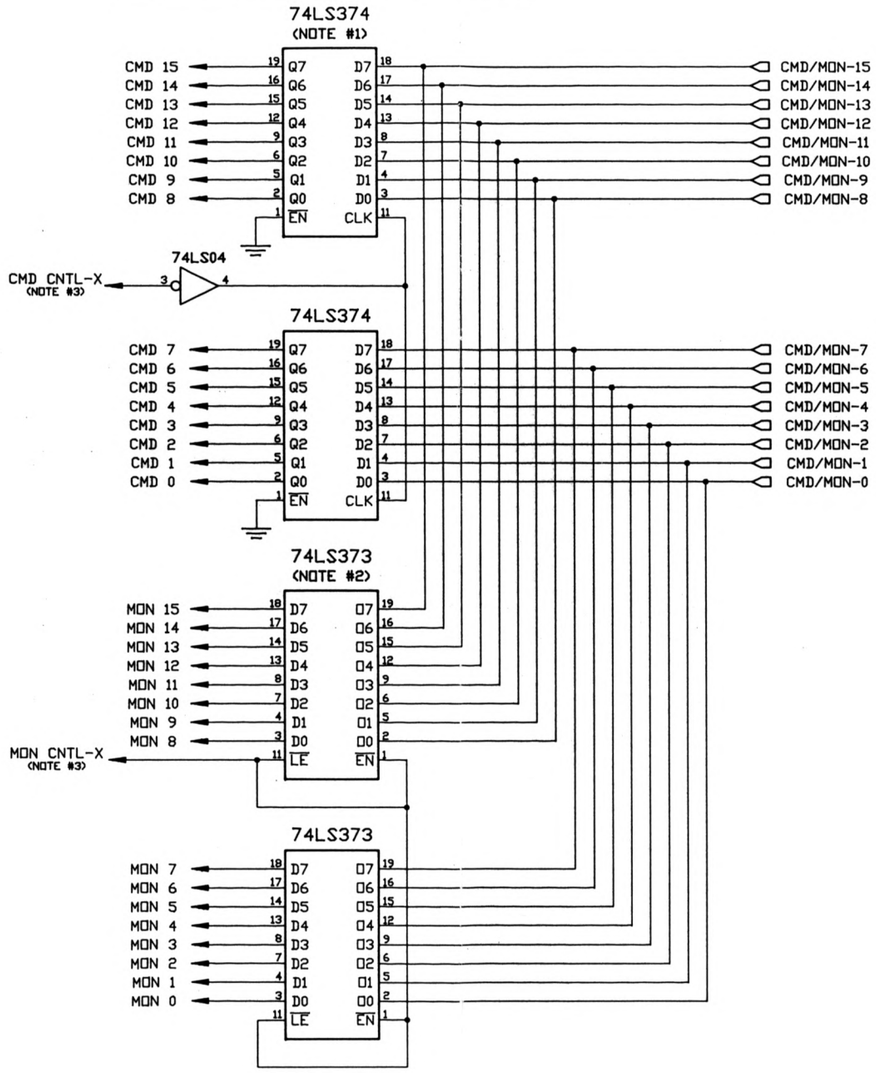
ACAD : SIBSAW-2
REF : VLBA-15

REV	DATE	DRAWN BY	APPR'D BY	DESCRIPTION
B	2-91	ANDREATTA		REDRAWN WITH ACAD ;

1 OF 8 POSSIBLE ANALOGS



1 OF 8 POSSIBLE CMD/MON STAGES



NOTES :

1. YOU CAN UTILIZE OTHER STORAGE GATES IF YOUR CIRCUIT REQUIRES POWER ON RESET, ETC... SUGGESTIONS ARE 74LS174, 74LS175, AND 74LS273.
2. IN PLACE OF THE 74LS373 YOU CAN USE A TRI-STATE BUFFER GATE SUCH AS 74LS240, 74LS241, 74LS367, OR 74LS368 ETC...
3. THE PARAMETER 'X' WILL TAKE ON THE VALUE 0-7.
4. ◄——— = DENOTES A CONNECTION WITHIN THE DEVICE

   ———◁ = DENOTES A CONNECTION BETWEEN THE DEVICE AND STANDARD INTERFACE

8 ANALOG/MONITOR STAGES/COMMAND STAGES

ACAD : SIB-SMPL

NEXT ASSEMBLY	DWG. TYPE

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES : ANGLES ±
3 PLACE DECIMALS (.XXX) ±
2 PLACE DECIMALS (.XX) ±
1 PLACE DECIMALS (.X) ±
MATERIAL :
FINISH :

VLBA PROJECT: STANDARD INTERFACE BOARD

TITLE: STANDARD INTERFACE BOARD SAMPLE DEVICE INTERFACE SCHEMATIC

NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO 87801

DRAWN BY	KOSKI	DATE	8-85
DESIGNED BY	KOSKI	DATE	8-85
APPROVED BY		DATE	

SHEET NUMBER 1 of 1 | DRAWING NUMBER C55002S003 | REV. B | SCALE —

6.0 APPENDIX

This Section contains the following:

Data sheets: 8032, 8755, 8156, 75174, 75175, SDM854, INA101 and HS9410

Specifications:

A55001N002-A, Monitor and Control Standard Interface
A55001N001, Monitor and Control Bus at VLBA Stations
EIA RS-485 Standard

MCS®-51 FAMILY OF MICROCONTROLLERS ARCHITECTURAL OVERVIEW

CONTENTS

INTRODUCTION

The 8051 is the original member of the MCS®-51 family, and is the core for all MCS-51 devices. The features of the 8051 core are:

- 8-bit CPU optimized for control applications
- Extensive Boolean processing (single-bit logic) capabilities
- 64K Program Memory address space
- 64K Data Memory address space
- 4K bytes of on-chip Program Memory
- 128 bytes of on-chip Data RAM
- 32 bidirectional and individually addressable I/O lines
- Two 16-bit timer/counters
- Full duplex UART
- 6-source/5-vector interrupt structure with two priority levels
- On-chip clock oscillator

The basic architectural structure of this 8051 core is shown in Figure 1.

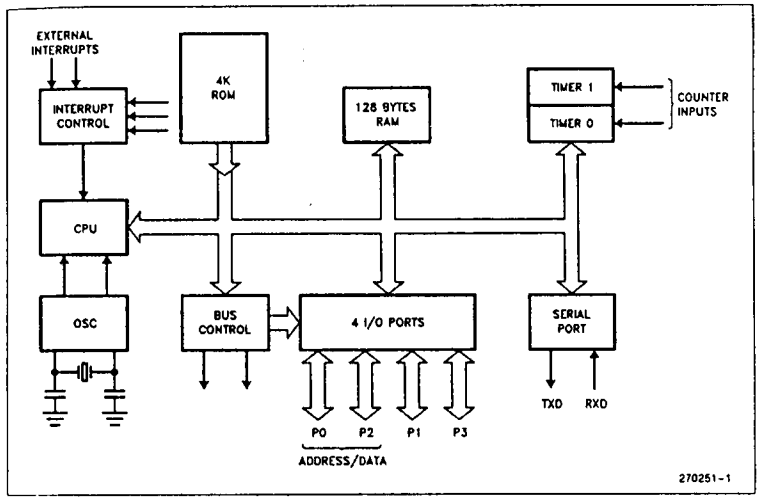


Figure 1. Block Diagram of the 8051 Core

Each device on the MCS-51 family consists of all the core features plus some additional features. A feature comparison of all the MCS-51 devices is shown in Table 1.

Table 1. The MCS®-51 Family of Microcontrollers

Device	ROMless Version	EPROM Version	ROM Bytes	RAM Bytes	8-Bit I/O Ports	16-Bit Timer/ Counters	Programmable Counter Array (PCA)	UART	Serial Expansion Port (SEP)	Global Serial Channel (GSC)	DMA Channels	A/D Channels	Interrupt Sources/ Vectors	Power Down and Idle Modes
8051	8031	—	4K	128	4	2		✓					6/5	
8051AH	8031AH	8751H 8751BH	4K	128	4	2		✓					6/5	
8052AH	8032AH	8752BH	8K	256	4	3		✓					8/6	
80C51BH	80C31BH	87C51	4K	128	4	2		✓					6/5	✓
80C52	80C32	—	8K	256	4	3		✓					8/6	✓
83C51FA	80C51FA	87C51FA	8K	256	4	3	✓	✓					14/7	✓
83C51FB	80C51FA	87C51FB	16K	256	4	3	✓	✓					14/7	✓
83C152JA	80C152JA	—	8K	256	5	2		✓		✓	2		19/11	✓
—	80C152JB	—	—	256	7	2		✓		✓	2		19/11	✓
83C152JC	80C152JC	—	8K	256	5	2		✓		✓	2		15/11	✓
—	80C152JD	—	—	256	7	2		✓		✓	2		19/11	✓
83C452	80C452	87C452P	8K	256	5	2		✓					9/8	✓

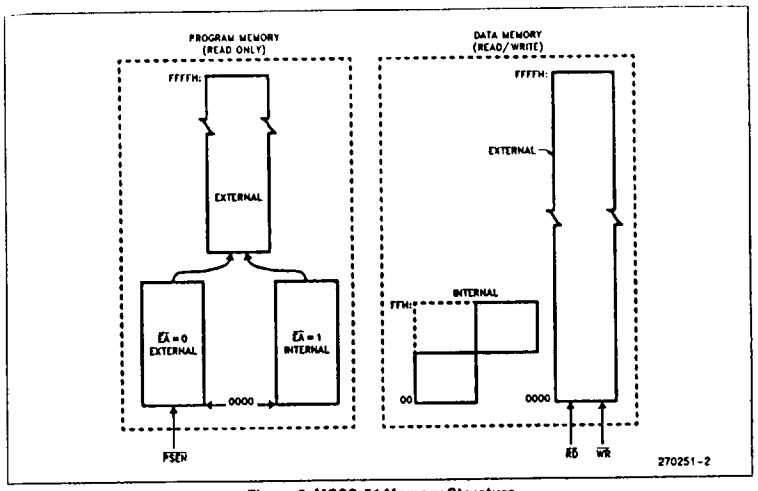


Figure 2. MCS®-51 Memory Structure

CHMOS Devices

Functionally, the CHMOS devices (designated with "C" in the middle of the device name) are all fully compatible with the 8051, but being CMOS, draw less current than an HMOS counterpart. To further exploit the power savings available in CMOS circuitry, two reduced power modes are added:

- Software-invoked Idle Mode, during which the CPU is turned off while the RAM and other on-chip peripherals continue operating. In this mode, current draw is reduced to about 15% of the current drawn when the device is fully active.
- Software-invoked Power Down Mode, during which all on-chip activities are suspended. The on-chip RAM continues to hold its data. In this mode the device typically draws less than 10 μA.

Although the 80C51BH is functionally compatible with its HMOS counterpart, specific differences between the two types of devices must be considered in the design of an application circuit if one wishes to ensure complete interchangeability between the HMOS and CHMOS devices. These considerations are discussed in the Application Note AP-252, "Designing with the 80C51BH".

For more information on the individual devices and features listed in Table 1, refer to the Hardware Descriptions and Data Sheets of the specific device.

MEMORY ORGANIZATION IN MCS®-51 DEVICES

5

Logical Separation of Program and Data Memory

All MCS-51 devices have separate address spaces for Program and Data Memory, as shown in Figure 2. The logical separation of Program and Data Memory allows the Data Memory to be accessed by 8-bit addresses, which can be more quickly stored and manipulated by an 8-bit CPU. Nevertheless, 16-bit Data Memory addresses can also be generated through the DPTR register.

Program Memory can only be read, not written to. There can be up to 64K bytes of Program Memory. In the ROM and EPROM versions of these devices the lowest 4K, 8K or 16K bytes of Program Memory are provided on-chip. Refer to Table 1 for the amount of on-chip ROM (or EPROM) on each device. In the ROMless versions all Program Memory is external. The read strobe for external Program Memory is the signal $\overline{PSEN}$ (Program Store Enable).

Data Memory occupies a separate address space from Program Memory. Up to 64K bytes of external RAM can be addressed in the external Data Memory space. The CPU generates read and write signals, $\overline{RD}$ and $\overline{WR}$, as needed during external Data Memory accesses.

External Program Memory and external Data Memory may be combined if desired by applying the $\overline{RD}$ and $\overline{PSEN}$ signals to the inputs of an AND gate and using the output of the gate as the read strobe to the external Program/Data memory.

Program Memory

Figure 3 shows a map of the lower part of the Program Memory. After reset, the CPU begins execution from location 0000H.

As shown in Figure 3, each interrupt is assigned a fixed location in Program Memory. The interrupt causes the CPU to jump to that location, where it commences execution of the service routine. External Interrupt 0, for example, is assigned to location 0003H. If External Interrupt 0 is going to be used, its service routine must begin at location 0003H. If the interrupt is not going to be used, its service location is available as general purpose Program Memory.

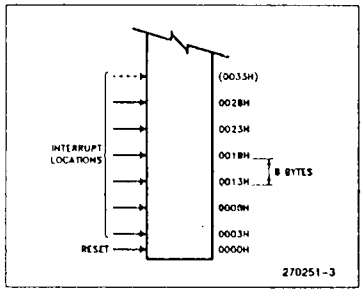


Figure 3. MCS®-51 Program Memory

The interrupt service locations are spaced at 8-byte intervals: 0003H for External Interrupt 0, 000BH for Timer 0, 0013H for External Interrupt 1, 001BH for Timer 1, etc. If an interrupt service routine is short enough (as is often the case in control applications), it can reside entirely within that 8-byte interval. Longer service routines can use a jump instruction to skip over subsequent interrupt locations, if other interrupts are in use.

The lowest 4K (or 8K or 16K) bytes of Program Memory can be either in the on-chip ROM or in an external ROM. This selection is made by strapping the $\overline{EA}$ (External Access) pin to either V_{CC} or V_{SS}.

In the 4K byte ROM devices, if the $\overline{EA}$ pin is strapped to V_{CC}, then program fetches to addresses 0000H through 0FFFH are directed to the internal ROM. Program fetches to addresses 1000H through FFFFH are directed to external ROM.

In the 8K byte ROM devices, $\overline{EA}$ = V_{CC} selects addresses 0000H through 1FFFH to be internal, and addresses 2000H through FFFFH to be external.

In the 16K byte ROM devices, $\overline{EA}$ = V_{CC} selects addresses 0000H through 3FFFH to be internal, and addresses 4000H through FFFFH to be external.

If the $\overline{EA}$ pin is strapped to V_{SS}, then all program fetches are directed to external ROM. The ROMless parts must have this pin externally strapped to V_{SS} to enable them to execute properly.

The read strobe to external ROM, $\overline{PSEN}$, is used for all external program fetches. $\overline{PSEN}$ is not activated for internal program fetches.

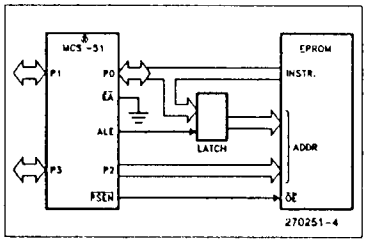


Figure 4. Executing from External Program Memory

The hardware configuration for external program execution is shown in Figure 4. Note that 16 I/O lines (Ports 0 and 2) are dedicated to bus functions during external Program Memory fetches. Port 0 (P0 in Figure 4) serves as a multiplexed address/data bus. It emits the low byte of the Program Counter (PCL) as an address, and then goes into a float state awaiting the arrival of the code byte from the Program Memory. During the time that the low byte of the Program Counter is valid on P0, the signal ALE (Address Latch Enable) clocks this byte into an address latch. Meanwhile, Port 2 (P2 in Figure 4) emits the high byte of the Program Counter (PCH). Then $\overline{PSEN}$ strobes the EPROM and the code byte is read into the microcontroller.

Program Memory addresses are always 16 bits wide, even though the actual amount of Program Memory used may be less than 64K bytes. External program execution sacrifices two of the 8-bit ports, P0 and P2, to the function of addressing the Program Memory.

Data Memory

The right half of Figure 2 shows the internal and external Data Memory spaces available to the MCS-51 user.

Figure 5 shows a hardware configuration for accessing up to 2K bytes of external RAM. The CPU in this case is executing from internal ROM. Port 0 serves as a multiplexed address/data bus to the RAM, and 3 lines of Port 2 are being used to page the RAM. The CPU generates $\overline{RD}$ and $\overline{WR}$ signals as needed during external RAM accesses.

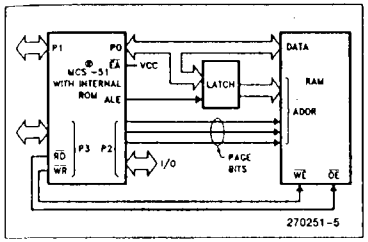


Figure 5. Accessing External Data Memory. If the Program Memory is Internal, the Other Bits of P2 are Available as I/O.

There can be up to 64K bytes of external Data Memory. External Data Memory addresses can be either 1 or 2 bytes wide. One-byte addresses are often used in conjunction with one or more other I/O lines to page the RAM, as shown in Figure 5. Two-byte addresses can also be used, in which case the high address byte is emitted at Port 2.

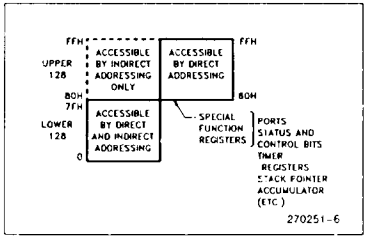


Figure 6. Internal Data Memory

Internal Data Memory is mapped in Figure 6. The memory space is shown divided into three blocks, which are generally referred to as the Lower 128, the Upper 128, and SFR space.

Internal Data Memory addresses are always one byte wide, which implies an address space of only 256 bytes. However, the addressing modes for internal RAM can in fact accommodate 384 bytes, using a simple trick. Direct addresses higher than 7FH access one memory space, and indirect addresses higher than 7FH access a different memory space. Thus Figure 6 shows the Upper 128 and SFR space occupying the same block of addresses, 80H through FFH, although they are physically separate entities.

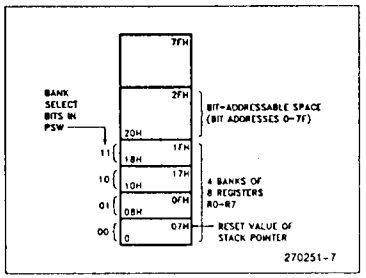


Figure 7. The Lower 128 Bytes of Internal RAM

The Lower 128 bytes of RAM are present in all MCS-51 devices as mapped in Figure 7. The lowest 32 bytes are grouped into 4 banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word (PSW) select which register bank is in use. This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing.

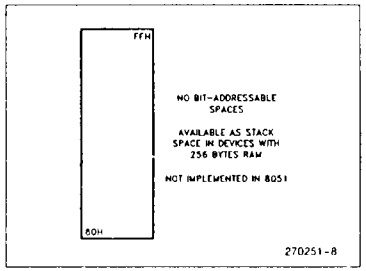


Figure 8. The Upper 128 Bytes of Internal RAM

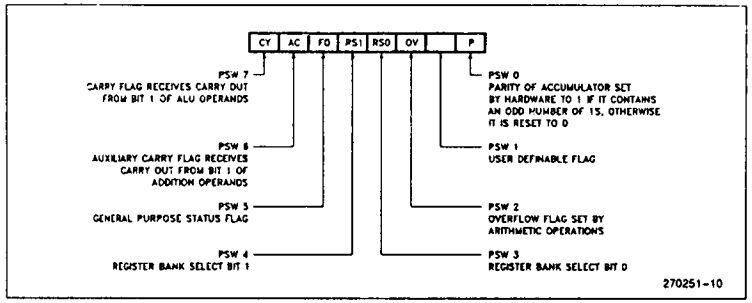


Figure 10. PSW (Program Status Word) Register in MCS®-51 Devices

The next 16 bytes above the register banks form a block of bit-addressable memory space. The MCS-51 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00H through 7FH.

All of the bytes in the Lower 128 can be accessed by either direct or indirect addressing. The Upper 128 (Figure 8) can only be accessed by indirect addressing. The Upper 128 bytes of RAM are not implemented in the 8051, but are in the devices with 256 bytes of RAM. (See Table 1).

Figure 9 gives a brief look at the Special Function Register (SFR) space. SFRs include the Port latches, timers, peripheral controls, etc. These registers can only be accessed by direct addressing. In general, all MCS-51 microcontrollers have the same SFRs as the 8051, and at the same addresses in SFR space. However, enhancements to the 8051 have additional SFRs that are not present in the 8051, nor perhaps in other proliferations of the family.

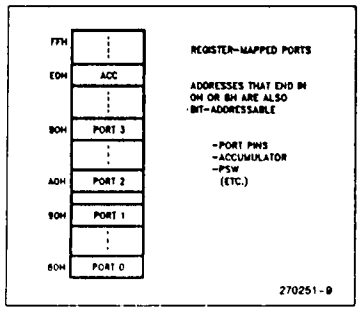


Figure 9. SFR Space

Sixteen addresses in SFR space are both byte- and bit-addressable. The bit-addressable SFRs are those whose address ends in 000B. The bit addresses in this area are 80H through FFH.

THE MCS®-51 INSTRUCTION SET

All members of the MCS-51 family execute the same instruction set. The MCS-51 instruction set is optimized for 8-bit control applications. It provides a variety of fast addressing modes for accessing the internal RAM to facilitate byte operations on small data structures. The instruction set provides extensive support for one-bit variables as a separate data type, allowing direct bit manipulation in control and logic systems that require Boolean processing.

An overview of the MCS-51 instruction set is presented below, with a brief description of how certain instructions might be used. References to "the assembler" in this discussion are to Intel's MCS-51 Macro Assembler, ASM51. More detailed information on the instruction set can be found in the MCS-51 Macro Assembler User's Guide (Order No. 9800937 for ISIS Systems, Order No. 122752 for DOS Systems).

Program Status Word

The Program Status Word (PSW) contains several status bits that reflect the current state of the CPU. The PSW, shown in Figure 10, resides in SFR space. It contains the Carry bit, the Auxiliary Carry (for BCD operations), the two register bank select bits, the Overflow flag, a Parity bit, and two user-definable status flags.

The Carry bit, other than serving the functions of a Carry bit in arithmetic operations, also serves as the "Accumulator" for a number of Boolean operations.

The bits RS0 and RS1 are used to select one of the four register banks shown in Figure 7. A number of instructions refer to these RAM locations as R0 through R7. The selection of which of the four banks is being referred to is made on the basis of the bits RS0 and RS1 at execution time.

The Parity bit reflects the number of 1s in the Accumulator: P = 1 if the Accumulator contains an odd number of 1s, and P = 0 if the Accumulator contains an even number of 1s. Thus the number of 1s in the Accumulator plus P is always even.

Two bits in the PSW are uncommitted and may be used as general purpose status flags.

Addressing Modes

The addressing modes in the MCS-51 instruction set are as follows:

DIRECT ADDRESSING

In direct addressing the operand is specified by an 8-bit address field in the instruction. Only internal Data RAM and SFRs can be directly addressed.

INDIRECT ADDRESSING

In indirect addressing the instruction specifies a register which contains the address of the operand. Both internal and external RAM can be indirectly addressed.

The address register for 8-bit addresses can be R0 or R1 of the selected register bank, or the Stack Pointer. The address register for 16-bit addresses can only be the 16-bit "data pointer" register, DPTR.

REGISTER INSTRUCTIONS

The register banks, containing registers R0 through R7, can be accessed by certain instructions which carry a 3-bit register specification within the opcode of the instruction. Instructions that access the registers this way are code efficient, since this mode eliminates an address byte. When the instruction is executed, one of the eight registers in the selected bank is accessed. One of four banks is selected at execution time by the two bank select bits in the PSW.

REGISTER-SPECIFIC INSTRUCTIONS

Some instructions are specific to a certain register. For example, some instructions always operate on the Accumulator, or Data Pointer, etc., so no address byte is needed to point to it. The opcode itself does that. Instructions that refer to the Accumulator as A assemble as accumulator-specific opcodes.

IMMEDIATE CONSTANTS

The value of a constant can follow the opcode in Program Memory. For example,

```
MOV A, #100
```

loads the Accumulator with the decimal number 100. The same number could be specified in hex digits as 64H.

INDEXED ADDRESSING

Only Program Memory can be accessed with indexed addressing, and it can only be read. This addressing mode is intended for reading look-up tables in Program Memory. A 16-bit base register (either DPTR or the Program Counter) points to the base of the table, and the Accumulator is set up with the table entry number. The address of the table entry in Program Memory is formed by adding the Accumulator data to the base pointer.

Another type of indexed addressing is used in the "case jump" instruction. In this case the destination address of a jump instruction is computed as the sum of the base pointer and the Accumulator data.

Arithmetic Instructions

The menu of arithmetic instructions is listed in Table 2. The table indicates the addressing modes that can be used with each instruction to access the <byte> operand. For example, the ADD A, <byte> instruction can be written as:

```
ADD     A,7FH       (direct addressing)
ADD     A,@R0       (indirect addressing)
ADD     A,R7        (register addressing)
ADD     A,#127      (immediate constant)
```

The execution times listed in Table 2 assume a 12 MHz clock frequency. All of the arithmetic instructions execute in 1 μs except the INC DPTR instruction, which takes 2 μs, and the Multiply and Divide instructions, which take 4 μs.

Note that any byte in the internal Data Memory space can be incremented or decremented without going through the Accumulator.

One of the INC instructions operates on the 16-bit Data Pointer. The Data Pointer is used to generate 16-bit addresses for external memory, so being able to increment it in one 16-bit operation is a useful feature.

The MUL AB instruction multiplies the Accumulator by the data in the B register and puts the 16-bit product into the concatenated B and Accumulator registers.

Table 2. A List of the MCS®-51 Arithmetic Instructions

Mnemonic	Operation	Addressing Modes				Execution Time (μs)
		Dir	Ind	Reg	Imm	
ADD A,<byte>	A = A + <byte>	X	X	X	X	1
ADDC A,<byte>	A = A + <byte> + C	X	X	X	X	1
SUBB A,<byte>	A = A − <byte> − C	X	X	X	X	1
INC A	A = A + 1	Accumulator only				1
INC <byte>	<byte> = <byte> + 1	X	X	X		1
INC DPTR	DPTR = DPTR + 1	Data Pointer only				2
DEC A	A = A − 1	Accumulator only				1
DEC <byte>	<byte> = <byte> − 1	X	X	X		1
MUL AB	B:A = B x A	ACC and B only				4
DIV AB	A = Int [A/B] B = Mod [A/B]	ACC and B only				4
DA A	Decimal Adjust	Accumulator only				1

The DIV AB instruction divides the Accumulator by the data in the B register and leaves the 8-bit quotient in the Accumulator, and the 8-bit remainder in the B register.

Oddly enough, DIV AB finds less use in arithmetic "divide" routines than in radix conversions and programmable shift operations. An example of the use of DIV AB in a radix conversion will be given later. In shift operations, dividing a number by 2^n shifts its n bits to the right. Using DIV AB to perform the division completes the shift in 4 μs and leaves the B register holding the bits that were shifted out.

The DA A instruction is for BCD arithmetic operations. In BCD arithmetic, ADD and ADDC instructions should always be followed by a DA A operation, to ensure that the result is also in BCD. Note that DA A will not convert a binary number to BCD. The DA A operation produces a meaningful result only as the second step in the addition of two BCD bytes.

Table 3. A List of the MCS®-51 Logical Instructions

Mnemonic	Operation	Addressing Modes				Execution Time (μs)
		Dir	Ind	Reg	Imm	
ANL A,<byte>	A = A .AND. <byte>	X	X	X	X	1
ANL <byte>,A	<byte> = <byte> .AND. A	X				1
ANL <byte>,#data	<byte> = <byte> .AND. #data	X				2
ORL A,<byte>	A = A .OR. <byte>	X	X	X	X	1
ORL <byte>,A	<byte> = <byte> .OR. A	X				1
ORL <byte>,#data	<byte> = <byte> .OR. #data	X				2
XRL A,<byte>	A = A .XOR. <byte>	X	X	X	X	1
XRL <byte>,A	<byte> = <byte> .XOR. A	X				1
XRL <byte>,#data	<byte> = <byte> .XOR. #data	X				2
CRL A	A = 00H	Accumulator only				1
CPL A	A = .NOT. A	Accumulator only				1
RL A	Rotate ACC Left 1 bit	Accumulator only				1
RLC A	Rotate Left through Carry	Accumulator only				1
RR A	Rotate ACC Right 1 bit	Accumulator only				1
RRC A	Rotate Right through Carry	Accumulator only				1
SWAP A	Swap Nibbles in A	Accumulator only				1

Logical Instructions

Table 3 shows the list of MCS-51 logical instructions. The instructions that perform Boolean operations (AND, OR, Exclusive OR, NOT) on bytes perform the operation on a bit-by-bit basis. That is, if the Accumulator contains 00110101B and <byte> contains 01010011B, then

```
ANL     A,<byte>
```

will leave the Accumulator holding 00010001B.

The addressing modes that can be used to access the <byte> operand are listed in Table 3. Thus, the ANL A,<byte> instruction may take any of the forms

```
ANL     A,7FH       (direct addressing)
ANL     A,@R1       (indirect addressing)
ANL     A,R6        (register addressing)
ANL     A,#53H      (immediate constant)
```

All of the logical instructions that are Accumulator-specific execute in 1μs (using a 12 MHz clock). The others take 2 μs.

Note that Boolean operations can be performed on any byte in the lower 128 internal Data Memory space or the SFR space using direct addressing, without having to use the Accumulator. The XRL <byte>, #data instruction, for example, offers a quick and easy way to invert port bits, as in

```
XRL     P1,#0FFH
```

If the operation is in response to an interrupt, not using the Accumulator saves the time and effort to stack it in the service routine.

The Rotate instructions (RL A, RLC A, etc.) shift the Accumulator 1 bit to the left or right. For a left rotation, the MSB rolls into the LSB position. For a right rotation, the LSR rolls into the MSB position.

The SWAP A instruction interchanges the high and low nibbles within the Accumulator. This is a useful operation in BCD manipulations. For example, if the Accumulator contains a binary number which is known to be less than 100, it can be quickly converted to BCD by the following code:

```
MOV     B,#10
DIV     AB
SWAP    A
ADD     A,B
```

Dividing the number by 10 leaves the tens digit in the low nibble of the Accumulator, and the ones digit in the B register. The SWAP and ADD instructions move the tens digit to the high nibble of the Accumulator, and the ones digit to the low nibble.

Data Transfers

INTERNAL RAM

Table 4 shows the menu of instructions that are available for moving data around within the internal memory spaces, and the addressing modes that can be used with each one. With a 12 MHz clock, all of these instructions execute in either 1 or 2 μs.

The MOV <dest>, <src> instruction allows data to be transferred between any two internal RAM or SFR locations without going through the Accumulator. Remember the Upper 128 byes of data RAM can be accessed only by indirect addressing, and SFR space only by direct addressing.

Note that in all MCS-51 devices, the stack resides in on-chip RAM, and grows upwards. The PUSH instruction first increments the Stack Pointer (SP), then copies the byte into the stack. PUSH and POP use only direct addressing to identify the byte being saved or restored,

Table 4. A List of the MCS®-51 Data Transfer Instructions that Access Internal Data Memory Space

Mnemonic	Operation	Addressing Modes				Execution Time (μs)
		Dir	Ind	Reg	Imm	
MOV A,<src>	A = <src>	X	X	X	X	1
MOV <dest>,A	<dest> = A	X	X	X		1
MOV <dest>,<src>	<dest> = <src>	X	X	X	X	2
MOV DPTR,#data16	DPTR = 16-bit immediate constant.				X	2
PUSH <src>	INC SP : MOV "@SP",<src>	X				2
POP <dest>	MOV <dest>, "@SP" : DEC SP	X				2
XCH A,<byte>	ACC and <byte> exchange data	X	X	X		1
XCHD A,@Ri	ACC and @Ri exchange low nibbles		X			1

but the stack itself is accessed by indirect addressing using the SP register. This means the stack can go into the Upper 128, if they are implemented, but not into SFR space.

In devices that do not implement the Upper 128, if the SP points to the Upper 128, PUSHed bytes are lost, and POPped bytes are indeterminate.

The Data Transfer instructions include a 16-bit MOV that can be used to initialize the Data Pointer (DPTR) for look-up tables in Program Memory, or for 16-bit external Data Memory accesses.

The XCH A, <byte> instruction causes the Accumulator and addressed byte to exchange data. The XCHD A,@Ri instruction is similar, but only the low nibbles are involved in the exchange.

To see how XCH and XCHD can be used to facilitate data manipulations, consider first the problem of shifting an 8-digit BCD number two digits to the right. Figure 11 shows how this can be done using direct MOVs, and for comparison how it can be done using XCH instructions. To aid in understanding how the code works, the contents of the registers that are holding the BCD number and the content of the Accumulator are shown alongside each instruction to indicate their status after the instruction has been executed.

		2A	2B	2C	2D	2E	ACC
MOV	A,2EH	00	12	34	56	78	78
MOV	2EH,2DH	00	12	34	56	56	78
MOV	2DH,2CH	00	12	34	34	56	78
MOV	2CH,2BH	00	12	12	34	56	78
MOV	2BH,#0	00	00	12	34	56	78

(a) Using direct MOVs: 14 bytes, 9 µs

		2A	2B	2C	2D	2E	ACC
CLR	A	00	12	34	56	78	00
XCH	A,2BH	00	00	34	56	78	12
XCH	A,2CH	00	00	12	56	78	34
XCH	A,2DH	00	00	12	34	78	56
XCH	A,2EH	00	00	12	34	56	78

(b) Using XCHs: 9 bytes, 5 µs

Figure 11. Shifting a BCD Number Two Digits to the Right

After the routine has been executed, the Accumulator contains the two digits that were shifted out on the right. Doing the routine with direct MOVs uses 14 code bytes and 9 µs of execution time (assuming a 12 MHz clock). The same operation with XCHs uses less code and executes almost twice as fast.

To right-shift by an odd number of digits, a one-digit shift must be executed. Figure 12 shows a sample of code that will right-shift a BCD number one digit, using the XCHD instruction. Again, the contents of the registers holding the number and of the Accumulator are shown alongside each instruction.

			2A	2B	2C	2D	2E	ACC
	MOV	R1,#2EH	00	12	34	56	78	XX
	MOV	R0,#2DH	00	12	34	56	78	XX
	loop for R1 = 2EH:							
LOOP:	MOV	A,@R1	00	12	34	56	78	78
	XCHD	A,@R0	00	12	34	58	78	76
	SWAP	A	00	12	34	58	78	67
	MOV	@R1,A	00	12	34	58	67	67
	DEC	R1	00	12	34	58	67	67
	DEC	R0	00	12	34	58	67	67
	CJNE	R1,#2AH,LOOP						
	loop for R1 = 2DH:		00	12	38	45	67	45
	loop for R1 = 2CH:		00	18	23	45	67	23
	loop for R1 = 2BH:		08	01	23	45	67	01
	CLR	A	08	01	23	45	67	00
	XCH	A,2AH	00	01	23	45	67	08

Figure 12. Shifting a BCD Number One Digit to the Right

First, pointers R1 and R0 are set up to point to the two bytes containing the last four BCD digits. Then a loop is executed which leaves the last byte, location 2EH, holding the last two digits of the shifted number. The pointers are decremented, and the loop is repeated for location 2DH. The CJNE instruction (Compare and Jump if Not Equal) is a loop control that will be described later.

The loop is executed from LOOP to CJNE for R1 = 2EH, 2DH, 2CH and 2BH. At that point the digit that was originally shifted out on the right has propagated to location 2AH. Since that location should be left with 0s, the lost digit is moved to the Accumulator.

EXTERNAL RAM

Table 5 shows a list of the Data Transfer instructions that access external Data Memory. Only indirect addressing can be used. The choice is whether to use a one-byte address, @Ri, where Ri can be either R0 or R1 of the selected register bank, or a two-byte address, @DPTR. The disadvantage to using 16-bit addresses if only a few K bytes of external RAM are involved is that 16-bit addresses use all 8 bits of Port 2 as address bus. On the other hand, 8-bit addresses allow one to address a few K bytes of RAM, as shown in Figure 5, without having to sacrifice all of Port 2.

All of these instructions execute in 2 µs, with a 12 MHz clock.

Table 5. A List of the MCS®-51 Data Transfer Instructions that Access External Data Memory Space

Address Width	Mnemonic	Operation	Execution Time (µs)
8 bits	MOVX A,@Ri	Read external RAM @Ri	2
8 bits	MOVX @Ri,A	Write external RAM @Ri	2
16 bits	MOVX A,@DPTR	Read external RAM @DPTR	2
16 bits	MOVX @DPTR,A	Write external RAM @DPTR	2

Note that in all external Data RAM accesses, the Accumulator is always either the destination or source of the data.

The read and write strobes to external RAM are activated only during the execution of a MOVX instruction. Normally these signals are inactive, and in fact if they're not going to be used at all, their pins are available as extra I/O lines. More about that later.

LOOKUP TABLES

Table 6 shows the two instructions that are available for reading lookup tables in Program Memory. Since these instructions access only Program Memory, the lookup tables can only be read, not updated. The mnemonic is MOVC for "move constant".

If the table access is to external Program Memory, then the read strobe is $\overline{PSEN}$.

Table 6. The MCS®-51 Lookup Table Read Instructions

Mnemonic	Operation	Execution Time (µs)
MOVC A,@A + DPTR	Read Pgm Memory at (A + DPTR)	2
MOVC A,@A + PC	Read Pgm Memory at (A + PC)	2

The first MOVC instruction in Table 6 can accommodate a table of up to 256 entries, numbered 0 through 255. The number of the desired entry is loaded into the Accumulator, and the Data Pointer is set up to point to beginning of the table. Then

```
MOVC    A,@A+DPTR
```

copies the desired table entry into the Accumulator.

The other MOVC instruction works the same way, except the Program Counter (PC) is used as the table base, and the table is accessed through a subroutine. First the number of the desired entry is loaded into the Accumulator, and the subroutine is called:

```
        MOV     A,ENTRY_NUMBER
        CALL    TABLE
```

The subroutine "TABLE" would look like this:

```
TABLE:  MOVC    A,@A+PC
        RET
```

The table itself immediately follows the RET (return) instruction in Program Memory. This type of table can have up to 255 entries, numbered 1 through 255. Number 0 can not be used, because at the time the MOVC instruction is executed, the PC contains the address of the RET instruction. An entry numbered 0 would be the RET opcode itself.

Boolean Instructions

MCS-51 devices contain a complete Boolean (single-bit) processor. The internal RAM contains 128 addressable bits, and the SFR space can support up to 128 other addressable bits. All of the port lines are bit-addressable, and each one can be treated as a separate single-bit port. The instructions that access these bits are not just conditional branches, but a complete menu of move, set, clear, complement, OR, and AND instructions. These kinds of bit operations are not easily obtained in other architectures with any amount of byte-oriented software.

Table 7. A List of the MCS®-51 Boolean Instructions

Mnemonic		Operation	Execution Time (μs)
ANL	C,bit	C = C .AND. bit	2
ANL	C,/bit	C = C .AND. .NOT. bit	2
ORL	C,bit	C = C .OR. bit	2
ORL	C,/bit	C = C .OR. .NOT. bit	2
MOV	C,bit	C = bit	1
MOV	bit,C	bit = C	2
CLR	C	C = 0	1
CLR	bit	bit = 0	1
SETB	C	C = 1	1
SETB	bit	bit = 1	1
CPL	C	C = .NOT. C	1
CPL	bit	bit = .NOT. bit	1
JC	rel	Jump if C = 1	2
JNC	rel	Jump if C = 0	2
JB	bit,rel	Jump if bit = 1	2
JNB	bit,rel	Jump if bit = 0	2
JBC	bit,rel	Jump if bit = 1; CLR bit	2

The instruction set for the Boolean processor is shown in Table 7. All bit accesses are by direct addressing. Bit addresses 00H through 7FH are in the Lower 128, and bit addresses 80H through FFH are in SFR space.

Note how easily an internal flag can be moved to a port pin:

```
MOV     C,FLAG
MOV     P1.0,C
```

In this example, FLAG is the name of any addressable bit in the Lower 128 or SFR space. An I/O line (the LSB of Port 1, in this case) is set or cleared depending on whether the flag bit is 1 or 0.

The Carry bit in the PSW is used as the single-bit Accumulator of the Boolean processor. Bit instructions that refer to the Carry bit as C assemble as Carry-specific instructions (CLR C, etc). The Carry bit also has a direct address, since it resides in the PSW register, which is bit-addressable.

Note that the Boolean instruction set includes ANL and ORL operations, but not the XRL (Exclusive OR) operation. An XRL operation is simple to implement in software. Suppose, for example, it is required to form the Exclusive OR of two bits:

```
C = bit1 .XRL. bit2
```

The software to do that could be as follows:

```
        MOV     C,bit1
        JNB     bit2,OVER
        CPL     C
OVER:   (continue)
```

First, bit1 is moved to the Carry. If bit2 = 0, then C now contains the correct result. That is, bit1 .XRL. bit2 = bit1 if bit2 = 0. On the other hand, if bit2 = 1 C now contains the complement of the correct result. It need only be inverted (CPL C) to complete the operation.

This code uses the JNB instruction, one of a series of bit-test instructions which execute a jump if the addressed bit is set (JC, JB, JBC) or if the addressed bit is not set (JNC, JNB). In the above case, bit2 is being tested, and if bit2 = 0 the CPL C instruction is jumped over.

JBC executes the jump if the addressed bit is set, and also clears the bit. Thus a flag can be tested and cleared in one operation.

All the PSW bits are directly addressable, so the Parity bit, or the general purpose flags, for example, are also available to the bit-test instructions.

RELATIVE OFFSET

The destination address for these jumps is specified to the assembler by a label or by an actual address in Program Memory. However, the destination address assembles to a **relative** offset byte. This is a signed (two's complement) offset byte which is added to the PC in two's complement arithmetic if the jump is executed.

The range of the jump is therefore −128 to +127 Program Memory bytes relative to the first byte following the instruction.

Jump Instructions

Table 8 shows the list of unconditional jumps.

Table 8. Unconditional Jumps in MCS®-51 Devices

Mnemonic	Operation	Execution Time (μs)
JMP addr	Jump to addr	2
JMP @A + DPTR	Jump to A + DPTR	2
CALL addr	Call subroutine at addr	2
RET	Return from subroutine	2
RETI	Return from interrupt	2
NOP	No operation	1

The Table lists a single "JMP addr" instruction, but in fact there are three—SJMP, LJMP and AJMP—which differ in the format of the destination address. JMP is a generic mnemonic which can be used if the programmer does not care which way the jump is encoded.

The SJMP instruction encodes the destination address as a relative offset, as described above. The instruction is 2 bytes long, consisting of the opcode and the relative offset byte. The jump distance is limited to a range of −128 to +127 bytes relative to the instruction following the SJMP.

The LJMP instruction encodes the destination address as a 16-bit constant. The instruction is 3 bytes long, consisting of the opcode and two address bytes. The destination address can be anywhere in the 64K Program Memory space.

The AJMP instruction encodes the destination address as an 11-bit constant. The instruction is 2 bytes long, consisting of the opcode, which itself contains 3 of the 11 address bits, followed by another byte containing the low 8 bits of the destination address. When the instruction is executed, these 11 bits are simply substituted for the low 11 bits in the PC. The high 5 bits stay the same. Hence the destination has to be within the same 2K block as the instruction following the AJMP.

In all cases the programmer specifies the destination address to the assembler in the same way: as a label or as a 16-bit constant. The assembler will put the destination address into the correct format for the given instruction. If the format required by the instruction will not support the distance to the specified destination address, a "Destination out of range" message is written into the List file.

The JMP @A + DPTR instruction supports case jumps. The destination address is computed at execution time as the sum of the 16-bit DPTR register and the Accumulator. Typically, DPTR is set up with the address of a jump table, and the Accumulator is given an index to the table. In a 5-way branch, for example, an integer 0 through 4 is loaded into the Accumulator. The code to be executed might be as follows:

```
MOV     DPTR,#JUMP_TABLE
MOV     A,INDEX_NUMBER
RL      A
JMP     @A + DPTR
```

The RL A instruction converts the index number (0 through 4) to an even number on the range 0 through 8, because each entry in the jump table is 2 bytes long:

```
JUMP_TABLE:
        AJMP    CASE_0
        AJMP    CASE_1
        AJMP    CASE_2
        AJMP    CASE_3
        AJMP    CASE_4
```

Table 8 shows a single "CALL addr" instruction, but there are two of them—LCALL and ACALL—which differ in the format in which the subroutine address is given to the CPU. CALL is a generic mnemonic which can be used if the programmer does not care which way the address is encoded.

The LCALL instruction uses the 16-bit address format, and the subroutine can be anywhere in the 64K Program Memory space. The ACALL instruction uses the 11-bit format, and the subroutine must be in the same 2K block as the instruction following the ACALL.

In any case the programmer specifies the subroutine address to the assembler in the same way: as a label or as a 16-bit constant. The assembler will put the address into the correct format for the given instructions.

Subroutines should end with a RET instruction, which returns execution to the instruction following the CALL.

RETI is used to return from an interrupt service routine. The only difference between RET and RETI is that RETI tells the interrupt control system that the interrupt in progress is done. If there is no interrupt in progress at the time RETI is executed, then the RETI is functionally identical to RET.

Table 9 shows the list of conditional jumps available to the MCS-51 user. All of these jumps specify the destination address by the relative offset method, and so are limited to a jump distance of −128 to +127 bytes from the instruction following the conditional jump instruction. Important to note, however, the user specifies to the assembler the actual destination address the same way as the other jumps: as a label or a 16-bit constant.

Table 9. Conditional Jumps in MCS®-51 Devices

Mnemonic	Operation	Addressing Modes				Execution Time (μs)
		Dir	Ind	Reg	Imm	
JZ rel	Jump if A = 0	Accumulator only				2
JNZ rel	Jump if A ≠ 0	Accumulator only				2
DJNZ <byte>,rel	Decrement and jump if not zero	X		X		2
CJNE A,<byte>,rel	Jump if A ≠ <byte>	X			X	2
CJNE <byte>,#data,rel	Jump if <byte> ≠ #data		X	X		2

There is no Zero bit in the PSW. The JZ and JNZ instructions test the Accumulator data for that condition.

The DJNZ instruction (Decrement and Jump if Not Zero) is for loop control. To execute a loop N times, load a counter byte with N and terminate the loop with a DJNZ to the beginning of the loop, as shown below for N = 10:

```
        MOV     COUNTER,#10
LOOP:   (begin loop)
          .
          .
          .
        (end loop)
        DJNZ    COUNTER,LOOP
        (continue)
```

The CJNE instruction (Compare and Jump if Not Equal) can also be used for loop control as in Figure 12. Two bytes are specified in the operand field of the instruction. The jump is executed only if the two bytes are not equal. In the example of Figure 12, the two bytes were the data in R1 and the constant 2AH. The initial data in R1 was 2EH. Every time the loop was executed, R1 was decremented, and the looping was to continue until the R1 data reached 2AH.

Another application of this instruction is in "greater than, less than" comparisons. The two bytes in the operand field are taken as unsigned integers. If the first is less than the second, then the Carry bit is set (1). If the first is greater than or equal to the second, then the Carry bit is cleared.

CPU TIMING

All MCS-51 microcontrollers have an on-chip oscillator which can be used if desired as the clock source for the CPU. To use the on-chip oscillator, connect a crystal or ceramic resonator between the XTAL1 and XTAL2 pins of the microcontroller, and capacitors to ground as shown in Figure 13.

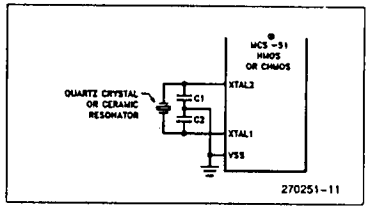


Figure 13. Using the On-Chip Oscillator

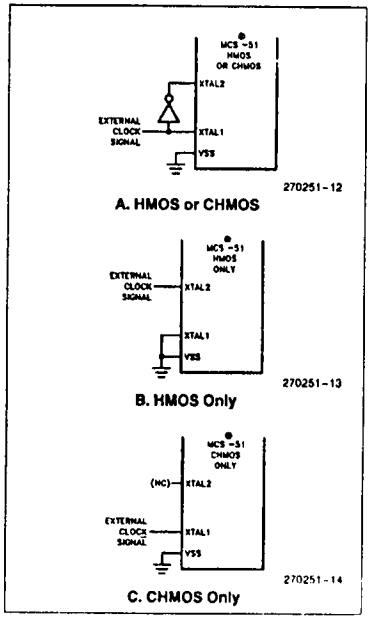


Figure 14. Using an External Clock

Examples of how to drive the clock with an external oscillator are shown in Figure 14. Note that in the HMOS devices (8051, etc.) the signal at the XTAL2 pin actually drives the internal clock generator. In the CHMOS devices (80C51BH, etc.) the signal at the XTAL1 pin drives the internal clock generator. If only one pin is going to be driven with the external oscillator signal, make sure it is the right pin.

The internal clock generator defines the sequence of states that make up the MCS-51 machine cycle.

Machine Cycles

A machine cycle consists of a sequence of 6 states, numbered S1 through S6. Each state time lasts for two oscillator periods. Thus a machine cycle takes 12 oscillator periods or 1 μs if the oscillator frequency is 12 MHz.

Each state is divided into a Phase 1 half and a Phase 2 half. Figure 15 shows the fetch/execute sequences in

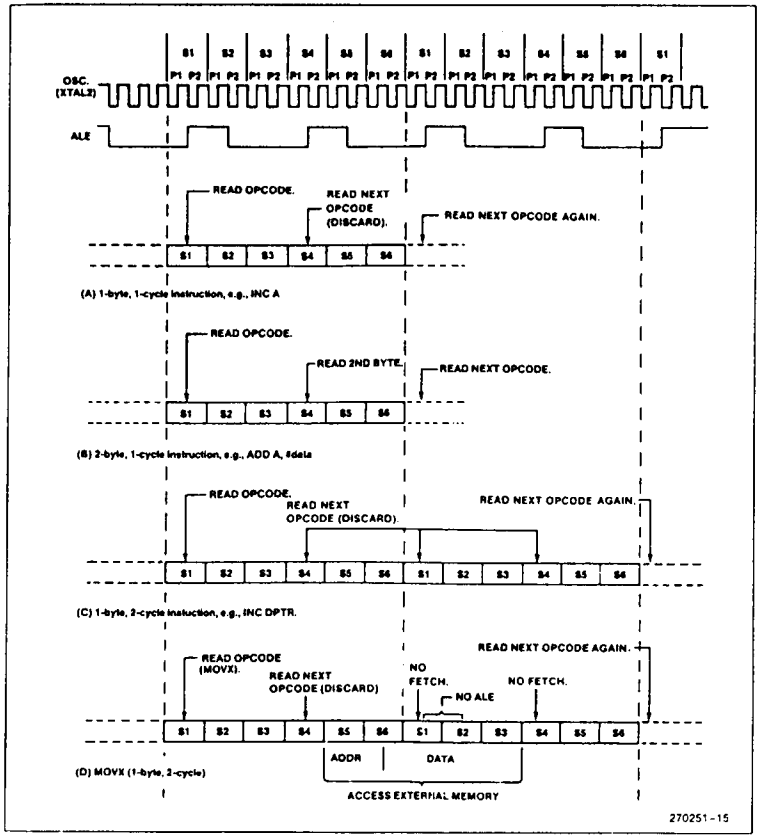


Figure 15. State Sequences in MCS®-51 Devices

states and phases for various kinds of instructions. Normally two program fetches are generated during each machine cycle, even if the instruction being executed doesn't require it. If the instruction being executed doesn't need more code bytes, the CPU simply ignores the extra fetch, and the Program Counter is not incremented.

Execution of a one-cycle instruction (Figure 15A and B) begins during State 1 of the machine cycle, when the opcode is latched into the Instruction Register. A second fetch occurs during S4 of the same machine cycle. Execution is complete at the end of State 6 of this machine cycle.

The MOVX instructions take two machine cycles to execute. No program fetch is generated during the second cycle of a MOVX instruction. This is the only time program fetches are skipped. The fetch/execute sequence for MOVX instructions is shown in Figure 15(D).

The fetch/execute sequences are the same whether the Program Memory is internal or external to the chip. Execution times do not depend on whether the Program Memory is internal or external.

Figure 16 shows the signals and timing involved in program fetches when the Program Memory is external. If Program Memory is external, then the Program Memory read strobe $\overline{PSEN}$ is normally activated twice per machine cycle, as shown in Figure 16(A).

If an access to external Data Memory occurs, as shown in Figure 16(B), two $\overline{PSEN}$s are skipped, because the address and data bus are being used for the Data Memory access.

Note that a Data Memory bus cycle takes twice as much time as a Program Memory bus cycle. Figure 16 shows the relative timing of the addresses being emitted at Ports 0 and 2, and of ALE and $\overline{PSEN}$. ALE is used to latch the low address byte from P0 into the address latch.

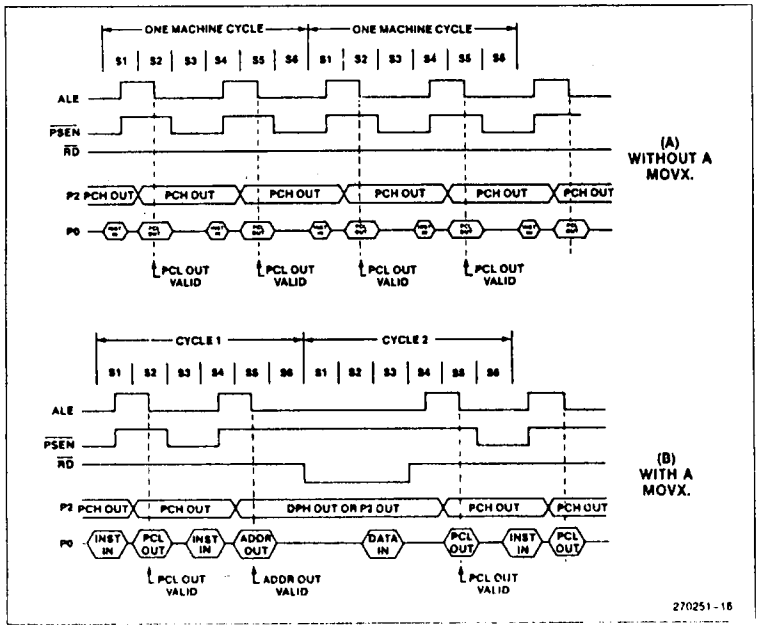


Figure 16. Bus Cycles in MCS®-51 Devices Executing from External Program Memory

When the CPU is executing from internal Program Memory, $\overline{PSEN}$ is not activated, and program addresses are not emitted. However, ALE continues to be activated twice per machine cycle and so is available as a clock output signal. Note, however, that one ALE is skipped during the execution of the MOVX instruction.

Interrupt Structure

The 8051 core provides 5 interrupt sources: 2 external interrupts, 2 timer interrupts, and the serial port interrupt. What follows is an overview of the interrupt structure for the 8051. Other MCS-51 devices have additional interrupt sources and vectors as shown in Table 1. Refer to the appropriate chapters on other devices for further information on their interrupts.

INTERRUPT ENABLES

Each of the interrupt sources can be individually enabled or disabled by setting or clearing a bit in the SFR named IE (Interrupt Enable). This register also contains a global disable bit, which can be cleared to disable all interrupts at once. Figure 17 shows the IE register for the 8051.

(MSB)							(LSB)
EA	–	–	ES	ET1	EX1	ET0	EX0

Enable bit = 1 enables the interrupt.
Enable bit = 0 disables it.

Symbol	Position	Function
EA	IE.7	disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
–	IE.6	reserved*
–	IE.5	reserved*
ES	IE.4	Serial Port interrupt enable bit.
ET1	IE.3	Timer 1 Overflow interrupt enable bit.
EX1	IE.2	External interrupt 1 enable bit.
ET0	IE.1	Timer 0 Overflow interrupt enable bit.
EX0	IE.0	External interrupt 0 enable bit.

*These reserved bits are used in other MCS-51 devices.

Figure 17. IE (Interrupt Enable) Register in the 8051

INTERRUPT PRIORITIES

Each interrupt source can also be individually programmed to one of two priority levels by setting or clearing a bit in the SFR named IP (Interrupt Priority). Figure 18 shows the IP register in the 8051.

A low-priority interrupt can be interrupted by a high-priority interrupt, but not by another low-priority interrupt. A high-priority interrupt can't be interrupted by any other interrupt source.

If two interrupt requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence.

Figure 19 shows, for the 8051, how the IE and IP registers and the polling sequence work to determine which if any interrupt will be serviced.

(MSB)							(LSB)
–	–	–	PS	PT1	PX1	PT0	PX0

Priority bit = 1 assigns high priority.
Priority bit = 0 assigns low priority.

Symbol	Position	Function
–	IP.7	reserved*
–	IP.6	reserved*
–	IP.5	reserved*
PS	IP.4	Serial Port interrupt priority bit.
PT1	IP.3	Timer 1 interrupt priority bit.
PX1	IP.2	External interrupt 1 priority bit.
PT0	IP.1	Timer 0 interrupt priority bit.
PX0	IP.0	External interrupt 0 priority bit.

*These reserved bits are used in other MCS-51 devices.

Figure 18. IP (Interrupt Priority) Register in the 8051

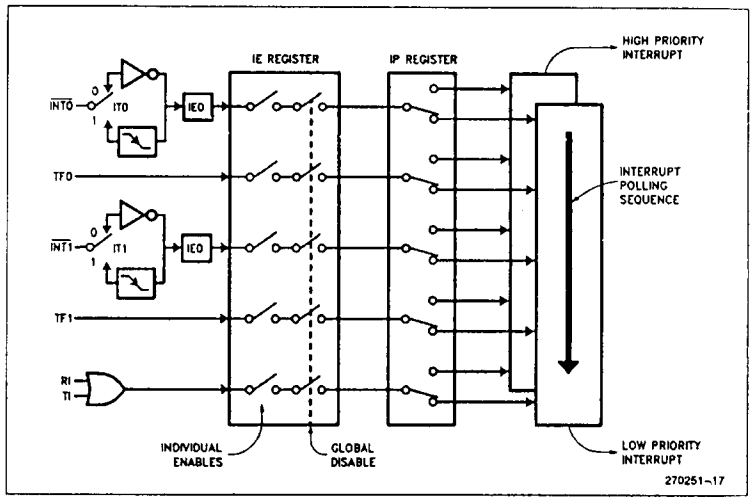


Figure 19. 8051 Interrupt Control System

In operation, all the interrupt flags are latched into the interrupt control system during State 5 of every machine cycle. The samples are polled during the following machine cycle. If the flag for an enabled interrupt is found to be set (1), the interrupt system generates an LCALL to the appropriate location in Program Memory, unless some other condition blocks the interrupt. Several conditions can block an interrupt, among them that an interrupt of equal or higher priority level is already in progress.

The hardware-generated LCALL causes the contents of the Program Counter to be pushed onto the stack, and reloads the PC with the beginning address of the service routine. As previously noted (Figure 3), the service routine for each interrupt begins at a fixed location.

Only the Program Counter is automatically pushed onto the stack, not the PSW or any other register. Having only the PC be automatically saved allows the programmer to decide how much time to spend saving which other registers. This enhances the interrupt response time, albeit at the expense of increasing the programmer's burden of responsibility. As a result, many interrupt functions that are typical in control applications—toggling a port pin, for example, or reloading a timer, or unloading a serial buffer—can often be completed in less time than it takes other architectures to commence them.

SIMULATING A THIRD PRIORITY LEVEL IN SOFTWARE

Some applications require more than the two priority levels that are provided by on-chip hardware in MCS-51 devices. In these cases, relatively simple software can be written to produce the same effect as a third priority level.

First, interrupts that are to have higher priority than 1 are assigned to priority 1 in the IP (Interrupt Priority) register. The service routines for priority 1 interrupts that are supposed to be interruptible by "priority 2" interrupts are written to include the following code:

```
        PUSH    IE
        MOV     IE, #MASK
        CALL    LABEL
        .......
(execute service routine)
        .......
        POP     IE
        RET
LABEL:  RETI
```


As soon as any priority 1 interrupt is acknowledged, the IE (Interrupt Enable) register is re-defined so as to disable all but "priority 2" interrupts. Then, a CALL to LABEL executes the RETI instruction, which clears the priority 1 interrupt-in-progress flip-flop. At this point any priority 1 interrupt that is enabled can be serviced, but only "priority 2" interrupts are enabled.

POPping IE restores the original enable byte. Then a normal RET (rather than another RETI) is used to terminate the service routine. The additional software adds 10 μs (at 12 MHz) to priority 1 interrupts.

ADDITIONAL REFERENCES

The following application notes are found in the *Embedded Control Applications* handbook. (Order Number: 270648)

1. AP-69 "An Introduction to the Intel MCS®-51 Single-Chip Microcomputer Family"
2. AP-70 "Using the Intel MCS®-51 Boolean Processing Capabilities"

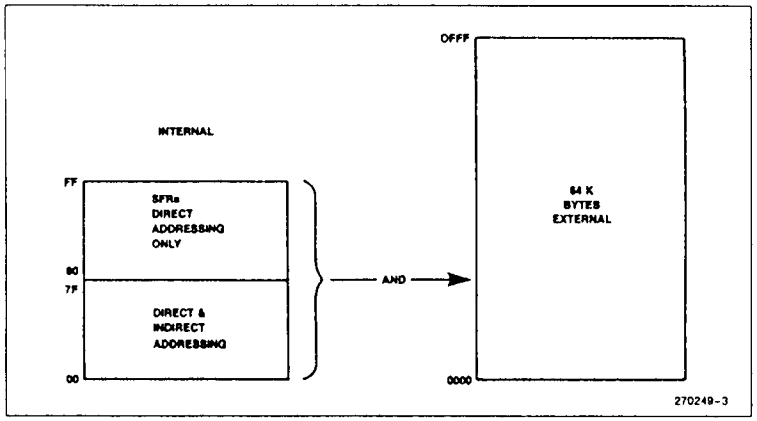


Figure 3a. The 8051 Data Memory

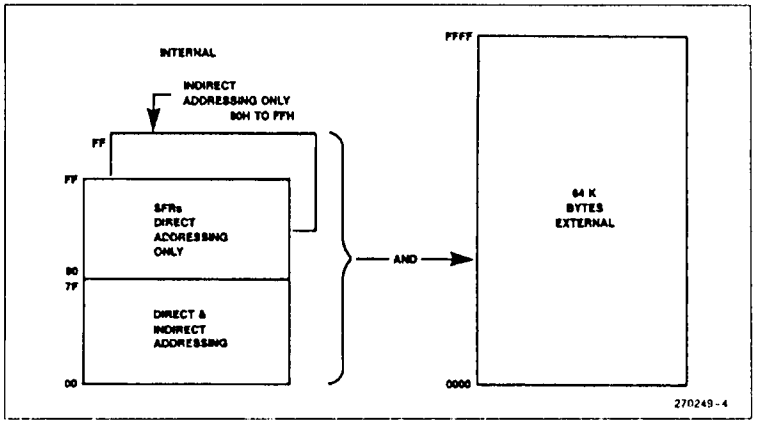


Figure 3b. The 8052 Data Memory

INDIRECT ADDRESS AREA:

Note that in Figure 3b the SFRs and the indirect address RAM have the same addresses (80H–0FFH). Nevertheless, they are two separate areas and are accessed in two different ways.

For example the instruction

```
MOV    80H,#0AAH
```

writes 0AAH to Port 0 which is one of the SFRs and the instruction

```
MOV    R0,#80H
MOV    @R0,#0BBH
```

writes 0BBH in location 80H of the data RAM. Thus, after execution of both of the above instructions Port 0 will contain 0AAH and location 80 of the RAM will contain 0BBH.

Note that the stack operations are examples of indirect addressing, so the upper 128 bytes of data RAM are available as stack space in those devices which implement 256 bytes of internal RAM.

DIRECT AND INDIRECT ADDRESS AREA:

The 128 bytes of RAM which can be accessed by both direct and indirect addressing can be divided into 3 segments as listed below and shown in Figure 4.

1. **Register Banks 0-3:** Locations 0 through 1FH (32 bytes). ASM-51 and the device after reset default to register bank 0. To use the other register banks the user must select them in the software (refer to the MCS-51 Micro Assembler User's Guide). Each register bank contains 8 one-byte registers, 0 through 7.

Reset initializes the Stack Pointer to location 07H and it is incremented once to start from location 08H which is the first register (R0) of the second register bank. Thus, in order to use more than one register bank, the SP should be intialized to a different location of the RAM where it is not used for data storage (ie, higher part of the RAM).

2. **Bit Addressable Area:** 16 bytes have been assigned for this segment, 20H-2FH. Each one of the 128 bits of this segment can be directly addressed (0-7FH).

The bits can be referred to in two ways both of which are acceptable by the ASM-51. One way is to refer to their addresses, ie. 0 to 7FH. The other way is with reference to bytes 20H to 2FH. Thus, bits 0–7 can also be referred to as bits 20.0–20.7, and bits 8-FH are the same as 21.0–21.7 and so on.

Each of the 16 bytes in this segment can also be addressed as a byte.

3. **Scratch Pad Area:** Bytes 30H through 7FH are available to the user as data RAM. However, if the stack pointer has been initialized to this area, enough number of bytes should be left aside to prevent SP data destruction.

Figure 4 shows the different segments of the on-chip RAM.

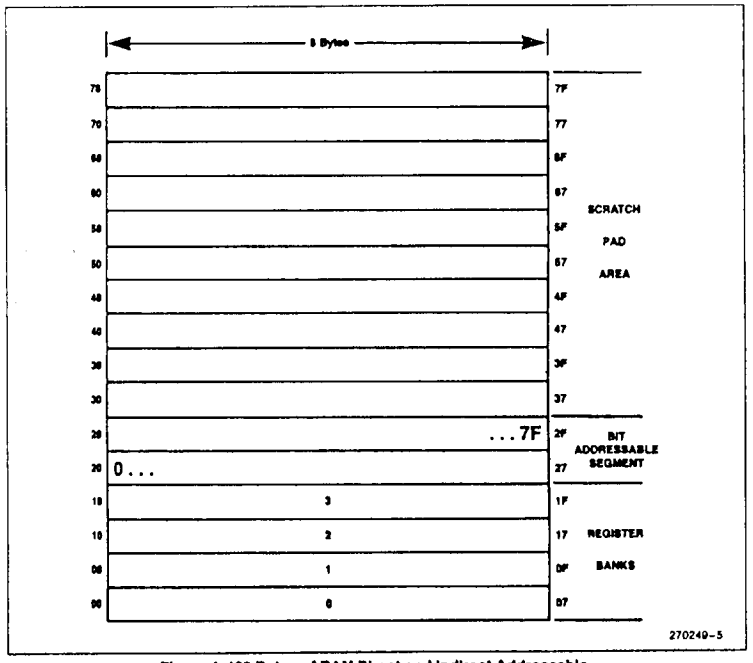


Figure 4. 128 Bytes of RAM Direct and Indirect Addressable

SPECIAL FUNCTION REGISTERS:

Table 1 contains a list of all the SFRs and their addresses.

Comparing Table 1 and Figure 5 shows that all of the SFRs that are byte and bit addressable are located on the first column of the diagram in Figure 5.

Table 1

Symbol	Name	Address
*ACC	Accumulator	0E0H
*B	B Register	0F0H
*PSW	Program Status Word	0D0H
SP	Stack Pointer	81H
DPTR	Data Pointer 2 Bytes	
DPL	Low Byte	82H
DPH	High Byte	83H
*P0	Port 0	80H
*P1	Port 1	90H
*P2	Port 2	0A0H
*P3	Port 3	0B0H
*IP	Interrupt Priority Control	0B8H
*IE	Interrupt Enable Control	0A8H
TMOD	Timer/Counter Mode Control	89H
*TCON	Timer/Counter Control	88H
*+T2CON	Timer/Counter 2 Control	0C8H
TH0	Timer/Counter 0 High Byte	8CH
TL0	Timer/Counter 0 Low Byte	8AH
TH1	Timer/Counter 1 High Byte	8DH
TL1	Timer/Counter 1 Low Byte	8BH
+TH2	Timer/Counter 2 High Byte	0CDH
+TL2	Timer/Counter 2 Low Byte	0CCH
+RCAP2H	T/C 2 Capture Reg. High Byte	0CBH
+RCAP2L	T/C 2 Capture Reg. Low Byte	0CAH
*SCON	Serial Control	98H
SBUF	Serial Data Buffer	99H
PCON	Power Control	87H

* = Bit addressable
+ = 8052 only

WHAT DO THE SFRs CONTAIN JUST AFTER POWER-ON OR A RESET?

Table 2 lists the contents of each SFR after power-on or a hardware reset.

Table 2. Contents of the SFRs after reset

Register	Value In Binary
*ACC	00000000
*B	00000000
*PSW	00000000
SP	00000111
DPTR	
DPH	00000000
DPL	00000000
*P0	11111111
*P1	11111111
*P2	11111111
*P3	11111111
*IP	8051 XXX00000, 8052 XX000000
*IE	8051 0XX00000, 8052 0X000000
TMOD	00000000
*TCON	00000000
*+T2CON	00000000
TH0	00000000
TL0	00000000
TH1	00000000
TL1	00000000
+TH2	00000000
+TL2	00000000
+RCAP2H	00000000
+RCAP2L	00000000
*SCON	00000000
SBUF	Indeterminate
PCON	HMOS 0XXXXXXX CHMOS 0XXX0000

X = Undefined
* = Bit Addressable
+ = 8052 only

SFR MEMORY MAP

	8 Bytes								
F8									FF
F0	B								F7
E8									EF
E0	ACC								E7
D8									DF
D0	PSW								D7
C8	T2CON		RCAP2L	RCAP2H	TL2	TH2			CF
C0									C7
B8	IP								BF
B0	P3								B7
A8	IE								AF
A0	P2								A7
98	SCON	SBUF							9F
90	P1								97
88	TCON	TMOD	TL0	TL1	TH0	TH1			8F
80	P0	SP	DPL	DPH				PCON	87

↑ Bit Addressable

Figure 5

Those SFRs that have their bits assigned for various functions are listed in this section. A brief description of each bit is provided for quick reference. For more detailed information refer to the Architecture Chapter of this book.

PSW: PROGRAM STATUS WORD. BIT ADDRESSABLE.

CY	AC	F0	RS1	RS0	OV	—	P

CY	PSW.7	Carry Flag.
AC	PSW.6	Auxiliary Carry Flag.
F0	PSW.5	Flag 0 available to the user for general purpose.
RS1	PSW.4	Register Bank selector bit 1 (SEE NOTE 1).
RS0	PSW.3	Register Bank selector bit 0 (SEE NOTE 1).
OV	PSW.2	Overflow Flag.
—	PSW.1	User definable flag.
P	PSW.0	Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even number of '1' bits in the accumulator.

NOTE:
1. The value presented by RS0 and RS1 selects the corresponding register bank.

RS1	RS0	Register Bank	Address
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

PCON: POWER CONTROL REGISTER. NOT BIT ADDRESSABLE.

SMOD	—	—	—	GF1	GF0	PD	IDL

SMOD	Double baud rate bit. If Timer 1 is used to generate baud rate and SMOD = 1, the baud rate is doubled when the Serial Port is used in modes 1, 2, or 3.
—	Not implemented, reserved for future use.*
—	Not implemented, reserved for future use.*
—	Not implemented, reserved for future use.*
GF1	General purpose flag bit.
GF0	General purpose flag bit.
PD	Power Down bit. Setting this bit activates Power Down operation in the 80C51BH. (Available only in CHMOS).
IDL	Idle Mode bit. Setting this bit activates Idle Mode operation in the 80C51BH. (Available only in CHMOS).

If 1s are written to PD and IDL at the same time, PD takes precedence.

*User software should not write 1s to reserved bits. These bits may be used in future MCS-51 products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1.

INTERRUPTS:

In order to use any of the interrupts in the MCS-51, the following three steps must be taken.

1. Set the EA (enable all) bit in the IE register to 1.
2. Set the corresponding individual interrupt enable bit in the IE register to 1.
3. Begin the interrupt service routine at the corresponding Vector Address of that interrupt. See Table below.

Interrupt Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI & TI	0023H
TF2 & EXF2	002BH

In addition, for external interrupts, pins $\overline{INT0}$ and $\overline{INT1}$ (P3.2 and P3.3) must be set to 1, and depending on whether the interrupt is to be level or transition activated, bits IT0 or IT1 in the TCON register may need to be set to 1.

ITx = 0 level activated

ITx = 1 transition activated

IE: INTERRUPT ENABLE REGISTER. BIT ADDRESSABLE.

If the bit is 0, the corresponding interrupt is disabled. If the bit is 1, the corresponding interrupt is enabled.

EA	—	ET2	ES	ET1	EX1	ET0	EX0

EA	IE.7	Disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
—	IE.6	Not implemented, reserved for future use.*
ET2	IE.5	Enable or disable the Timer 2 overflow or capture interrupt (8052 only).
ES	IE.4	Enable or disable the serial port interrupt.
ET1	IE.3	Enable or disable the Timer 1 overflow interrupt.
EX1	IE.2	Enable or disable External Interrupt 1.
ET0	IE.1	Enable or disable the Timer 0 overflow interrupt.
EX0	IE.0	Enable or disable External Interrupt 0.

*User software should not write 1s to reserved bits. These bits may be used in future MCS-51 products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1.

ASSIGNING HIGHER PRIORITY TO ONE OR MORE INTERRUPTS:

In order to assign higher priority to an interrupt the corresponding bit in the IP register must be set to 1.

Remember that while an interrupt service is in progress, it cannot be interrupted by a lower or same level interrupt.

PRIORITY WITHIN LEVEL:

Priority within level is only to resolve simultaneous requests of the same priority level.

From high to low, interrupt sources are listed below:

IE0
TF0
IE1
TF1
RI or TI
TF2 or EXF2

IP: INTERRUPT PRIORITY REGISTER. BIT ADDRESSABLE.

If the bit is 0, the corresponding interrupt has a lower priority and if the bit is 1 the corresponding interrupt has a higher priority.

—	—	PT2	PS	PT1	PX1	PT0	PX0

— IP. 7 Not implemented, reserved for future use.*

— IP. 6 Not implemented, reserved for future use.*

PT2 IP. 5 Defines the Timer 2 interrupt priority level (8052 only).

PS IP. 4 Defines the Serial Port interrupt priority level.

PT1 IP. 3 Defines the Timer 1 interrupt priority level.

PX1 IP. 2 Defines External Interrupt 1 priority level.

PT0 IP. 1 Defines the Timer 0 interrupt priority level.

PX0 IP. 0 Defines the External Interrupt 0 priority level.

*User software should not write 1s to reserved bits. These bits may be used in future MCS-51 products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1.

TCON: TIMER/COUNTER CONTROL REGISTER. BIT ADDRESSABLE.

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1 TCON. 7 Timer 1 overflow flag. Set by hardware when the Timer/Counter 1 overflows. Cleared by hardware as processor vectors to the interrupt service routine.

TR1 TCON. 6 Timer 1 run control bit. Set/cleared by software to turn Timer/Counter 1 ON/OFF.

TF0 TCON. 5 Timer 0 overflow flag. Set by hardware when the Timer/Counter 0 overflows. Cleared by hardware as processor vectors to the service routine.

TR0 TCON. 4 Timer 0 run control bit. Set/cleared by software to turn Timer/Counter 0 ON/OFF.

IE1 TCON. 3 External Interrupt 1 edge flag. Set by hardware when External Interrupt edge is detected. Cleared by hardware when interrupt is processed.

IT1 TCON. 2 Interrupt 1 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt.

IE0 TCON. 1 External Interrupt 0 edge flag. Set by hardware when External Interrupt edge detected. Cleared by hardware when interrupt is processed.

IT0 TCON. 0 Interrupt 0 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt.

TMOD: TIMER/COUNTER MODE CONTROL REGISTER. NOT BIT ADDRESSABLE.

GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
TIMER 1				TIMER 0			

GATE When TRx (in TCON) is set and GATE = 1, TIMER/COUNTERx will run only while INTx pin is high (hardware control). When GATE = 0, TIMER/COUNTERx will run only while TRx = 1 (software control).

C/$\overline{T}$ Timer or Counter selector. Cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin).

M1 Mode selector bit. (NOTE 1)

M0 Mode selector bit. (NOTE 1)

NOTE 1:

M1	M0	Operating Mode	
0	0	0	13-bit Timer (MCS-48 compatible)
0	1	1	16-bit Timer/Counter
1	0	2	8-bit Auto-Reload Timer/Counter
1	1	3	(Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits, TH0 is an 8-bit Timer and is controlled by Timer 1 control bits.
1	1	3	(Timer 1) Timer/Counter 1 stopped.

TIMER SET-UP

Tables 3 through 6 give some values for TMOD which can be used to set up Timer 0 in different modes.

It is assumed that only one timer is being used at a time. If it is desired to run Timers 0 and 1 simultaneously, in any mode, the value in TMOD for Timer 0 must be ORed with the value shown for Timer 1 (Tables 5 and 6).

For example, if it is desired to run Timer 0 in mode 1 GATE (external control), and Timer 1 in mode 2 COUNTER, then the value that must be loaded into TMOD is 69H (09H from Table 3 ORed with 60H from Table 6).

Moreover, it is assumed that the user, at this point, is not ready to turn the timers on and will do that at a different point in the program by setting bit TR.x (in TCON) to 1.

TIMER/COUNTER 0

As a Timer:

Table 3

MODE	TIMER 0 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	00H	08H
1	16-bit Timer	01H	09H
2	8-bit Auto-Reload	02H	0AH
3	two 8-bit Timers	03H	0BH

As a Counter:

Table 4

MODE	COUNTER 0 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	04H	0CH
1	16-bit Timer	05H	0DH
2	8-bit Auto-Reload	06H	0EH
3	one 8-bit Counter	07H	0FH

NOTES:
1. The Timer is turned ON/OFF by setting/clearing bit TR0 in the software.
2. The Timer is turned ON/OFF by the 1 to 0 transition on $\overline{INT0}$ (P3.2) when TR0 = 1 (hardware control).

TIMER/COUNTER 1

As a Timer:

Table 5

MODE	TIMER 1 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	00H	80H
1	16-bit Timer	10H	90H
2	8-bit Auto-Reload	20H	A0H
3	does not run	30H	B0H

As a Counter:

Table 6

MODE	COUNTER 1 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	40H	C0H
1	16-bit Timer	50H	D0H
2	8-bit Auto-Reload	60H	E0H
3	not available	—	—

NOTES:
1. The Timer is turned ON/OFF by setting/clearing bit TR1 in the software.
2. The Timer is turned ON/OFF by the 1 to 0 transition on $\overline{INT1}$ (P3.3) when TR1 = 1 (hardware control).

T2CON: TIMER/COUNTER 2 CONTROL REGISTER, BIT ADDRESSABLE

8052 Only

TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$

TF2 T2CON. 7 Timer 2 overflow flag set by hardware and cleared by software. TF2 cannot be set when either RCLK = 1 or CLK = 1

EXF2 T2CON. 6 Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX, and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software.

RCLK T2CON. 5 Receive clock flag. When set, causes the Serial Port to use Timer 2 overflow pulses for its receive clock in modes 1 & 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.

TLCK T2CON. 4 Transmit clock flag. When set, causes the Serial Port to use Timer 2 overflow pulses for its transmit clock in modes 1 & 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.

EXEN2 T2CON. 3 Timer 2 external enable flag. When set, allows a capture or reload to occur as a result of negative transition on T2EX if Timer 2 is not being used to clock the Serial Port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.

TR2 T2CON. 2 Software START/STOP control for Timer 2. A logic 1 starts the Timer.

C/$\overline{T2}$ T2CON. 1 Timer or Counter select.

0 = Internal Timer. 1 = External Event Counter (falling edge triggered).

CP/$\overline{RL2}$ T2CON. 0 Capture/Reload flag. When set, captures will occur on negative transitions at T2EX if EXEN2 = 1. When cleared, Auto-Reloads will occur either with Timer 2 overflows or negative transitions at T2EX when EXEN2 = 1. When either RCLK = 1 or TCLK = 1, this bit is ignored and the Timer is forced to Auto-Reload on Timer 2 overflow.

TIMER/COUNTER 2 SET-UP

Except for the baud rate generator mode, the values given for T2CON do not include the setting of the TR2 bit. Therefore, bit TR2 must be set, separately, to turn the Timer on.

As a Timer:

Table 7

MODE	T2CON	
	INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
16-bit Auto-Reload	00H	08H
16-bit Capture	01H	09H
BAUD rate generator receive & transmit same baud rate	34H	36H
receive only	24H	26H
transmit only	14H	16H

As a Counter:

Table 8

MODE	TMOD	
	INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
16-bit Auto-Reload	02H	0AH
16-bit Capture	03H	0BH

NOTES:
1. Capture/Reload occurs only on Timer/Counter overflow.
2. Capture/Reload occurs on Timer/Counter overflow and a 1 to 0 transition on T2EX (P1.1) pin except when Timer 2 is used in the baud rate generating mode.

SCON: SERIAL PORT CONTROL REGISTER. BIT ADDRESSABLE.

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0 SCON. 7 Serial Port mode specifier. (NOTE 1).

SM1 SCON. 6 Serial Port mode specifier. (NOTE 1).

SM2 SCON. 5 Enables the multiprocessor communication feature in modes 2 & 3. In mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In mode 0, SM2 should be 0. (See Table 9).

REN SCON. 4 Set/Cleared by software to Enable/Disable reception.

TB8 SCON. 3 The 9th bit that will be transmitted in modes 2 & 3. Set/Cleared by software.

RB8 SCON. 2 In modes 2 & 3, is the 9th data bit that was received. In mode 1, if SM2 = 0, RB8 is the stop bit that was received. In mode 0, RB8 is not used.

TI SCON. 1 Transmit interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or at the beginning of the stop bit in the other modes. Must be cleared by software.

RI SCON. 0 Receive interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or halfway through the stop bit time in the other modes (except see SM2). Must be cleared by software.

NOTE 1:

SM0	SM1	Mode	Description	Baud Rate
0	0	0	SHIFT REGISTER	Fosc./12
0	1	1	8-Bit UART	Variable
1	0	2	9-Bit UART	Fosc./64 OR Fosc./32
1	1	3	9-Bit UART	Variable

SERIAL PORT SET-UP:

Table 9

MODE	SCON	SM2 VARIATION
0	10H	Single Processor Environment (SM2 = 0)
1	50H	
2	90H	
3	D0H	
0	NA	Multiprocessor Environment (SM2 = 1)
1	70H	
2	B0H	
3	F0H	

GENERATING BAUD RATES

Serial Port in Mode 0:

Mode 0 has a fixed baud rate which is 1/12 of the oscillator frequency. To run the serial port in this mode none of the Timer/Counters need to be set up. Only the SCON register needs to be defined.

$$\text{Baud Rate} = \frac{\text{Osc Freq}}{12}$$

Serial Port in Mode 1:

Mode 1 has a variable baud rate. The baud rate can be generated by either Timer 1 or Timer 2 (8052 only).

USING TIMER/COUNTER 1 TO GENERATE BAUD RATES:

For this purpose, Timer 1 is used in mode 2 (Auto-Reload). Refer to Timer Setup section of this chapter.

$$\text{Baud Rate} = \frac{\text{K x Oscillator Freq}}{32 \times 12 \times [256 - (\text{TH1})]}$$

If SMOD = 0, then K = 1.
If SMOD = 1, then K = 2. (SMOD is the PCON register).

Most of the time the user knows the baud rate and needs to know the reload value for TH1. Therefore, the equation to calculate TH1 can be written as:

$$\text{TH1} = 256 - \frac{\text{K x Osc Freq}}{384 \times \text{baud rate}}$$

TH1 must be an integer value. Rounding off TH1 to the nearest integer may not produce the desired baud rate. In this case, the user may have to choose another crystal frequency.

Since the PCON register is not bit addressable, one way to set the bit is logical ORing the PCON register. (ie, ORL PCON, #80H). The address of PCON is 87H.

USING TIMER/COUNTER 2 TO GENERATE BAUD RATES:

For this purpose, Timer 2 must be used in the baud rate generating mode. Refer to Timer 2 Setup Table in this chapter. If Timer 2 is being clocked through pin T2 (P1.0) the baud rate is:

$$\text{Baud Rate} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

And if it is being clocked internally the baud rate is:

$$\text{Baud Rate} = \frac{\text{Osc Freq}}{32 \times [65536 - (\text{RCAP2H, RCAP2L})]}$$

To obtain the reload value for RCAP2H and RCAP2L the above equation can be rewritten as:

$$\text{RCAP2H, RCAP2L} = 65536 - \frac{\text{Osc Freq}}{32 \times \text{Baud Rate}}$$

SERIAL PORT IN MODE 2:

The baud rate is fixed in this mode and is $1/_{32}$ or $1/_{64}$ of the oscillator frequency depending on the value of the SMOD bit in the PCON register.

In this mode none of the Timers are used and the clock comes from the internal phase 2 clock.

SMOD = 1, Baud Rate = $1/_{32}$ Osc Freq.

SMOD = 0, Baud Rate = $1/_{64}$ Osc Freq.

To set the SMOD bit: ORL PCON, #80H. The address of PCON is 87H.

SERIAL PORT IN MODE 3:

The baud rate in mode 3 is variable and sets up exactly the same as in mode 1.

MCS®-51 INSTRUCTION SET

Table 10. 8051 Instruction Set Summary

Interrupt Response Time: Refer to Hardware Description Chapter.

Instructions that Affect Flag Settings(1)

Instruction	Flag C	OV	AC	Instruction	Flag C	OV	AC
ADD	X	X	X	CLR C	0		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C,bit	X		
MUL	0	X		ANL C,/bit	X		
DIV	0	X		ORL C,bit	X		
DA	X			ORL C,bit	X		
RRC	X			MOV C,bit	X		
RLC	X			CJNE	X		
SETB C	1						

(1)Note that operations on SFR byte address 208 or bit addresses 209-215 (i.e., the PSW or bits in the PSW) will also affect flag settings.

Note on instruction set and addressing modes:

Rn — Register R7–R0 of the currently selected Register Bank.

direct — 8-bit internal data location's address. This could be an Internal Data RAM location (0–127) or a SFR [i.e., I/O port, control register, status register, etc. (128–255)].

@Ri — 8-bit internal data RAM location (0–255) addressed indirectly through register R1 or R0.

#data — 8-bit constant included in instruction.

#data 16 — 16-bit constant included in instruction.

addr 16 — 16-bit destination address. Used by LCALL & LJMP. A branch can be anywhere within the 64K-byte Program Memory address space.

addr 11 — 11-bit destination address. Used by ACALL & AJMP. The branch will be within the same 2K-byte page of program memory as the first byte of the following instruction.

rel — Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is −128 to +127 bytes relative to first byte of the following instruction.

bit — Direct Addressed bit in Internal Data RAM or Special Function Register.

Mnemonic		Description	Byte	Oscillator Period
ARITHMETIC OPERATIONS				
ADD	A,Rn	Add register to Accumulator	1	12
ADD	A,direct	Add direct byte to Accumulator	2	12
ADD	A,@Ri	Add indirect RAM to Accumulator	1	12
ADD	A,#data	Add immediate data to Accumulator	2	12
ADDC	A,Rn	Add register to Accumulator with Carry	1	12
ADDC	A,direct	Add direct byte to Accumulator with Carry	2	12
ADDC	A,@Ri	Add indirect RAM to Accumulator with Carry	1	12
ADDC	A,#data	Add immediate data to Acc with Carry	2	12
SUBB	A,Rn	Subtract Register from Acc with borrow	1	12
SUBB	A,direct	Subtract direct byte from Acc with borrow	2	12
SUBB	A,@Ri	Subtract indirect RAM from ACC with borrow	1	12
SUBB	A,#data	Subtract immediate data from Acc with borrow	2	12
INC	A	Increment Accumulator	1	12
INC	Rn	Increment register	1	12
INC	direct	Increment direct byte	2	12
INC	@Ri	Increment direct RAM	1	12
DEC	A	Decrement Accumulator	1	12
DEC	Rn	Decrement Register	1	12
DEC	direct	Decrement direct byte	2	12
DEC	@Ri	Decrement indirect RAM	1	12

Table 10. 8051 Instruction Set Summary (Continued)

Mnemonic		Description	Byte	Oscillator Period
ARITHMETIC OPERATIONS (Continued)				
INC	DPTR	Increment Data Pointer	1	24
MUL	AB	Multiply A & B	1	48
DIV	AB	Divide A by B	1	48
DA	A	Decimal Adjust Accumulator	1	12
LOGICAL OPERATIONS				
ANL	A,Rn	AND Register to Accumulator	1	12
ANL	A,direct	AND direct byte to Accumulator	2	12
ANL	A,@Ri	AND indirect RAM to Accumulator	1	12
ANL	A,#data	AND immediate data to Accumulator	2	12
ANL	direct,A	AND Accumulator to direct byte	2	12
ANL	direct,#data	AND immediate data to direct byte	3	24
ORL	A,Rn	OR register to Accumulator	1	12
ORL	A,direct	OR direct byte to Accumulator	2	12
ORL	A,@Ri	OR indirect RAM to Accumulator	1	12
ORL	A,#data	OR immediate data to Accumulator	2	12
ORL	direct,A	OR Accumulator to direct byte	2	12
ORL	direct,#data	OR immediate data to direct byte	3	24
XRL	A,Rn	Exclusive-OR register to Accumulator	1	12
XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	12
XRL	A,@Ri	Exclusive-OR indirect RAM to Accumulator	1	12
XRL	A,#data	Exclusive-OR immediate data to Accumulator	2	12
XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	12
XRL	direct,#data	Exclusive-OR immediate data to direct byte	3	24
CLR	A	Clear Accumulator	1	12
CPL	A	Complement Accumulator	1	12

Mnemonic		Description	Byte	Oscillator Period
LOGICAL OPERATIONS (Continued)				
RL	A	Rotate Accumulator Left	1	12
RLC	A	Rotate Accumulator Left through the Carry	1	12
RR	A	Rotate Accumulator Right	1	12
RRC	A	Rotate Accumulator Right through the Carry	1	12
SWAP	A	Swap nibbles within the Accumulator	1	12
DATA TRANSFER				
MOV	A,Rn	Move register to Accumulator	1	12
MOV	A,direct	Move direct byte to Accumulator	2	12
MOV	A,@Ri	Move indirect RAM to Accumulator	1	12
MOV	A,#data	Move immediate data to Accumulator	2	12
MOV	Rn,A	Move Accumulator to register	1	12
MOV	Rn,direct	Move direct byte to register	2	24
MOV	Rn,#data	Move immediate data to register	2	12
MOV	direct,A	Move Accumulator to direct byte	2	12
MOV	direct,Rn	Move register to direct byte	2	24
MOV	direct,direct	Move direct byte to direct	3	24
MOV	direct,@Ri	Move indirect RAM to direct byte	2	24
MOV	direct,#data	Move immediate data to direct byte	3	24
MOV	@Ri,A	Move Accumulator to indirect RAM	1	12

Table 10. 8051 Instruction Set Summary (Continued)

Mnemonic		Description	Byte	Oscillator Period
DATA TRANSFER (Continued)				
MOV	@Ri,direct	Move direct byte to indirect RAM	2	24
MOV	@Ri, #data	Move immediate data to indirect RAM	2	12
MOV	DPTR, #data16	Load Data Pointer with a 16-bit constant	3	24
MOVC	A,@A + DPTR	Move Code byte relative to DPTR to Acc	1	24
MOVC	A,@A + PC	Move Code byte relative to PC to Acc	1	24
MOVX	A,@Ri	Move External RAM (8-bit addr) to Acc	1	24
MOVX	A,@DPTR	Move External RAM (16-bit addr) to Acc	1	24
MOVX	@Ri,A	Move Acc to External RAM (8-bit addr)	1	24
MOVX	@DPTR,A	Move Acc to External RAM (16-bit addr)	1	24
PUSH	direct	Push direct byte onto stack	2	24
POP	direct	Pop direct byte from stack	2	24
XCH	A,Rn	Exchange register with Accumulator	1	12
XCH	A,direct	Exchange direct byte with Accumulator	2	12
XCH	A,@Ri	Exchange indirect RAM with Accumulator	1	12
XCHD	A,@Ri	Exchange low-order Digit indirect RAM with Acc	1	12

Mnemonic		Description	Byte	Oscillator Period
BOOLEAN VARIABLE MANIPULATION				
CLR	C	Clear Carry	1	12
CLR	bit	Clear direct bit	2	12
SETB	C	Set Carry	1	12
SETB	bit	Set direct bit	2	12
CPL	C	Complement Carry	1	12
CPL	bit	Complement direct bit	2	12
ANL	C,bit	AND direct bit to CARRY	2	24
ANL	C,/bit	AND complement of direct bit to Carry	2	24
ORL	C,bit	OR direct bit to Carry	2	24
ORL	C,/bit	OR complement of direct bit to Carry	2	24
MOV	C,bit	Move direct bit to Carry	2	12
MOV	bit,C	Move Carry to direct bit	2	24
JC	rel	Jump if Carry is set	2	24
JNC	rel	Jump if Carry not set	2	24
JB	bit,rel	Jump if direct Bit is set	3	24
JNB	bit,rel	Jump if direct Bit is Not set	3	24
JBC	bit,rel	Jump if direct Bit is set & clear bit	3	24
PROGRAM BRANCHING				
ACALL	addr11	Absolute Subroutine Call	2	24
LCALL	addr16	Long Subroutine Call	3	24
RET		Return from Subroutine	1	24
RETI		Return from interrupt	1	24
AJMP	addr11	Absolute Jump	2	24
LJMP	addr16	Long Jump	3	24
SJMP	rel	Short Jump (relative addr)	2	24

Table 10. 8051 Instruction Set Summary (Continued)

Mnemonic		Description	Byte	Oscillator Period
PROGRAM BRANCHING (Continued)				
JMP	@A + DPTR	Jump indirect relative to the DPTR	1	24
JZ	rel	Jump if Accumulator is Zero	2	24
JNZ	rel	Jump if Accumulator is Not Zero	2	24
CJNE	A,direct,rel	Compare direct byte to Acc and Jump if Not Equal	3	24
CJNE	A, #data,rel	Compare immediate to Acc and Jump if Not Equal	3	24

Mnemonic		Description	Byte	Oscillator Period
PROGRAM BRANCHING (Continued)				
CJNE	Rn, #data,rel	Compare immediate to register and Jump if Not Equal	3	24
CJNE	@Ri, #data,rel	Compare immediate to indirect and Jump if Not Equal	3	24
DJNZ	Rn,rel	Decrement register and Jump if Not Zero	2	24
DJNZ	direct,rel	Decrement direct byte and Jump if Not Zero	3	24
NOP		No Operation	1	12

Table 11. Instruction Opcodes in Hexadecimal Order

Hex Code	Number of Bytes	Mnemonic	Operands
00	1	NOP	
01	2	AJMP	code addr
02	3	LJMP	code addr
03	1	RR	A
04	1	INC	A
05	2	INC	data addr
06	1	INC	@R0
07	1	INC	@R1
08	1	INC	R0
09	1	INC	R1
0A	1	INC	R2
0B	1	INC	R3
0C	1	INC	R4
0D	1	INC	R5
0E	1	INC	R6
0F	1	INC	R7
10	3	JBC	bit addr, code addr
11	2	ACALL	code addr
12	3	LCALL	code addr
13	1	RRC	A
14	1	DEC	A
15	2	DEC	data addr
16	1	DEC	@R0
17	1	DEC	@R1
18	1	DEC	R0
19	1	DEC	R1
1A	1	DEC	R2
1B	1	DEC	R3
1C	1	DEC	R4
1D	1	DEC	R5
1E	1	DEC	R6
1F	1	DEC	R7
20	3	JB	bit addr, code addr
21	2	AJMP	code addr
22	1	RET	
23	1	RL	A
24	2	ADD	A,#data
25	2	ADD	A,data addr
26	1	ADD	A,@R0
27	1	ADD	A,@R1
28	1	ADD	A,R0
29	1	ADD	A,R1
2A	1	ADD	A,R2
2B	1	ADD	A,R3
2C	1	ADD	A,R4
2D	1	ADD	A,R5
2E	1	ADD	A,R6
2F	1	ADD	A,R7
30	3	JNB	bit addr, code addr
31	2	ACALL	code addr
32	1	RETI	
33	1	RLC	A
34	2	ADDC	A,#data
35	2	ADDC	A,data addr
36	1	ADDC	A,@R0
37	1	ADDC	A,@R1
38	1	ADDC	A,R0
39	1	ADDC	A,R1
3A	1	ADDC	A,R2
3B	1	ADDC	A,R3
3C	1	ADDC	A,R4
3D	1	ADDC	A,R5
3E	1	ADDC	A,R6
3F	1	ADDC	A,R7
40	2	JC	code addr
41	2	AJMP	code addr
42	2	ORL	data addr,A
43	3	ORL	data addr,#data
44	2	ORL	A,#data
45	2	ORL	A,data addr
46	1	ORL	A,@R0
47	1	ORL	A,@R1
48	1	ORL	A,R0
49	1	ORL	A,R1
4A	1	ORL	A,R2
4B	1	ORL	A,R3
4C	1	ORL	A,R4
4D	1	ORL	A,R5
4E	1	ORL	A,R6
4F	1	ORL	A,R7
50	2	JNC	code addr
51	2	ACALL	code addr
52	2	ANL	data addr,A
53	3	ANL	data addr,#data
54	2	ANL	A,#data
55	2	ANL	A,data addr
56	1	ANL	A,@R0
57	1	ANL	A,@R1
58	1	ANL	A,R0
59	1	ANL	A,R1
5A	1	ANL	A,R2
5B	1	ANL	A,R3
5C	1	ANL	A,R4
5D	1	ANL	A,R5
5E	1	ANL	A,R6
5F	1	ANL	A,R7
60	2	JZ	code addr
61	2	AJMP	code addr
62	2	XRL	data addr,A
63	3	XRL	data addr,#data
64	2	XRL	A,#data
65	2	XRL	A,data addr

Table 11. Instruction Opcodes in Hexadecimal Order (Continued)

Hex Code	Number of Bytes	Mnemonic	Operands
66	1	XRL	A,@R0
67	1	XRL	A,@R1
68	1	XRL	A,R0
69	1	XRL	A,R1
6A	1	XRL	A,R2
6B	1	XRL	A,R3
6C	1	XRL	A,R4
6D	1	XRL	A,R5
6E	1	XRL	A,R6
6F	1	XRL	A,R7
70	2	JNZ	code addr
71	2	ACALL	code addr
72	2	ORL	C,bit addr
73	1	JMP	@A+DPTR
74	2	MOV	A,#data
75	3	MOV	data addr,#data
76	2	MOV	@R0,#data
77	2	MOV	@R1,#data
78	2	MOV	R0,#data
79	2	MOV	R1,#data
7A	2	MOV	R2,#data
7B	2	MOV	R3,#data
7C	2	MOV	R4,#data
7D	2	MOV	R5,#data
7E	2	MOV	R6,#data
7F	2	MOV	R7,#data
80	2	SJMP	code addr
81	2	AJMP	code addr
82	2	ANL	C,bit addr
83	1	MOVC	A,@A+PC
84	1	DIV	AB
85	3	MOV	data addr, data addr
86	2	MOV	data addr,@R0
87	2	MOV	data addr,@R1
88	2	MOV	data addr,R0
89	2	MOV	data addr,R1
8A	2	MOV	data addr,R2
8B	2	MOV	data addr,R3
8C	2	MOV	data addr,R4
8D	2	MOV	data addr,R5
8E	2	MOV	data addr,R6
8F	2	MOV	data addr,R7
90	3	MOV	DPTR,#data
91	2	ACALL	code addr
92	2	MOV	bit addr,C
93	1	MOVC	A,@A+DPTR
94	2	SUBB	A,#data
95	2	SUBB	A,data addr
96	1	SUBB	A,@R0
97	1	SUBB	A,@R1
98	1	SUBB	A,R0
99	1	SUBB	A,R1
9A	1	SUBB	A,R2
9B	1	SUBB	A,R3
9C	1	SUBB	A,R4
9D	1	SUBB	A,R5
9E	1	SUBB	A,R6
9F	1	SUBB	A,R7
A0	2	ORL	C,/bit addr
A1	2	AJMP	code addr
A2	2	MOV	C,bit addr
A3	1	INC	DPTR
A4	1	MUL	AB
A5		reserved	
A6	2	MOV	@R0,data addr
A7	2	MOV	@R1,data addr
A8	2	MOV	R0,data addr
A9	2	MOV	R1,data addr
AA	2	MOV	R2,data addr
AB	2	MOV	R3,data addr
AC	2	MOV	R4,data addr
AD	2	MOV	R5,data addr
AE	2	MOV	R6,data addr
AF	2	MOV	R7,data addr
B0	2	ANL	C,/bit addr
B1	2	ACALL	code addr
B2	2	CPL	bit addr
B3	1	CPL	C
B4	3	CJNE	A,#data,code addr
B5	3	CJNE	A,data addr,code addr
B6	3	CJNE	@R0,#data,code addr
B7	3	CJNE	@R1,#data,code addr
B8	3	CJNE	R0,#data,code addr
B9	3	CJNE	R1,#data,code addr
BA	3	CJNE	R2,#data,code addr
BB	3	CJNE	R3,#data,code addr
BC	3	CJNE	R4,#data,code addr
BD	3	CJNE	R5,#data,code addr
BE	3	CJNE	R6,#data,code addr
BF	3	CJNE	R7,#data,code addr
C0	2	PUSH	data addr
C1	2	AJMP	code addr
C2	2	CLR	bit addr
C3	1	CLR	C
C4	1	SWAP	A
C5	2	XCH	A,data addr
C6	1	XCH	A,@R0
C7	1	XCH	A,@R1
C8	1	XCH	A,R0
C9	1	XCH	A,R1
CA	1	XCH	A,R2
CB	1	XCH	A,R3

Table 11. Instruction Opcodes in Hexadecimal Order (Continued)

Hex Code	Number of Bytes	Mnemonic	Operands
CC	1	XCH	A,R4
CD	1	XCH	A,R5
CE	1	XCH	A,R6
CF	1	XCH	A,R7
D0	2	POP	data addr
D1	2	ACALL	code addr
D2	2	SETB	bit addr
D3	1	SETB	C
D4	1	DA	A
D5	3	DJNZ	data addr,code addr
D6	1	XCHD	A,@R0
D7	1	XCHD	A,@R1
D8	2	DJNZ	R0,code addr
D9	2	DJNZ	R1,code addr
DA	2	DJNZ	R2,code addr
DB	2	DJNZ	R3,code addr
DC	2	DJNZ	R4,code addr
DD	2	DJNZ	R5,code addr
DE	2	DJNZ	R6,code addr
DF	2	DJNZ	R7,code addr
E0	1	MOVX	A,@DPTR
E1	2	AJMP	code addr
E2	1	MOVX	A,@R0
E3	1	MOVX	A,@R1
E4	1	CLR	A
E5	2	MOV	A,data addr
E6	1	MOV	A,@R0
E7	1	MOV	A,@R1
E8	1	MOV	A,R0
E9	1	MOV	A,R1
EA	1	MOV	A,R2
EB	1	MOV	A,R3
EC	1	MOV	A,R4
ED	1	MOV	A,R5
EE	1	MOV	A,R6
EF	1	MOV	A,R7
F0	1	MOVX	@DPTR,A
F1	2	ACALL	code addr
F2	1	MOVX	@R0,A
F3	1	MOVX	@R1,A
F4	1	CPL	A
F5	2	MOV	data addr,A
F6	1	MOV	@R0,A
F7	1	MOV	@R1,A
F8	1	MOV	R0,A
F9	1	MOV	R1,A
FA	1	MOV	R2,A
FB	1	MOV	R3,A
FC	1	MOV	R4,A
FD	1	MOV	R5,A
FE	1	MOV	R6,A
FF	1	MOV	R7,A

INSTRUCTION DEFINITIONS

ACALL addr11

Function: Absolute Call

Description: ACALL unconditionally calls a subroutine located at the indicated address. The instruction increments the PC twice to obtain the address of the following instruction, then pushes the 16-bit result onto the stack (low-order byte first) and increments the Stack Pointer twice. The destination address is obtained by successively concatenating the five high-order bits of the incremented PC, opcode bits 7-5, and the second byte of the instruction. The subroutine called must therefore start within the same 2K block of the program memory as the first byte of the instruction following ACALL. No flags are affected.

Example: Initially SP equals 07H. The label "SUBRTN" is at program memory location 0345 H. After executing the instruction,

ACALL SUBRTN

at location 0123H, SP will contain 09H, internal RAM locations 08H and 09H will contain 25H and 01H, respectively, and the PC will contain 0345H.

Bytes: 2

Cycles: 2

Encoding:

a10 a9 a8 1	0 0 0 1	a7 a6 a5 a4	a3 a2 a1 a0

Operation: ACALL
$(PC) \leftarrow (PC) + 2$
$(SP) \leftarrow (SP) + 1$
$((SP)) \leftarrow (PC_{7\text{-}0})$
$(SP) \leftarrow (SP) + 1$
$((SP)) \leftarrow (PC_{15\text{-}8})$
$(PC_{10\text{-}0}) \leftarrow$ page address

6

ADDC A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 0 1 1	1 r r r

Operation: ADDC
(A) ← (A) + (C) + (R_n)

ADDC A,direct

Bytes: 2

Cycles: 1

Encoding:

0 0 1 1	0 1 0 1	direct address

Operation: ADDC
(A) ← (A) + (C) + (direct)

ADDC A,@Ri

Bytes: 1

Cycles: 1

Encoding:

0 0 1 1	0 1 1 i

Operation: ADDC
(A) ← (A) + (C) + ((R_i))

ADDC A,#data

Bytes: 2

Cycles: 1

Encoding:

0 0 1 1	0 1 0 0	immediate data

Operation: ADDC
(A) ← (A) + (C) + #data

AJMP addr11

Function: Absolute Jump

Description: AJMP transfers program execution to the indicated address, which is formed at run-time by concatenating the high-order five bits of the PC (*after* incrementing the PC twice), opcode bits 7-5, and the second byte of the instruction. The destination must therefore be within the same 2K block of program memory as the first byte of the instruction following AJMP.

Example: The label "JMPADR" is at program memory location 0123H. The instruction,

AJMP JMPADR

is at location 0345H and will load the PC with 0123H.

Bytes: 2

Cycles: 2

Encoding:

a10 a9 a8 0	0 0 0 1	a7 a6 a5 a4	a3 a2 a1 a0

Operation: AJMP
(PC) ← (PC) + 2
(PC_{10-0}) ← page address

ANL <dest-byte>,<src-byte>

Function: Logical-AND for byte variables

Description: ANL performs the bitwise logical-AND operation between the variables indicated and stores the results in the destination variable. No flags are affected.

The two operands allow six addressing mode combinations. When the destination is the Accumulator, the source can use register, direct, register-indirect, or immediate addressing; when the destination is a direct address, the source can be the Accumulator or immediate data.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: If the Accumulator holds 0C3H (11000011B) and register 0 holds 55H (01010101B) then the instruction,

ANL A,R0

will leave 41H (01000001B) in the Accumulator.

When the destination is a directly addressed byte, this instruction will clear combinations of bits in any RAM location or hardware register. The mask byte determining the pattern of bits to be cleared would either be a constant contained in the instruction or a value computed in the Accumulator at run-time. The instruction,

ANL P1,#01110011B

will clear bits 7, 3, and 2 of output port 1.

ADD A,<src-byte>

Function: Add

Description: ADD adds the byte variable indicated to the Accumulator, leaving the result in the Accumulator. The carry and auxiliary-carry flags are set, respectively, if there is a carry-out from bit 7 or bit 3, and cleared otherwise. When adding unsigned integers, the carry flag indicates an overflow occured.

OV is set if there is a carry-out of bit 6 but not out of bit 7, or a carry-out of bit 7 but not bit 6; otherwise OV is cleared. When adding signed integers, OV indicates a negative number produced as the sum of two positive operands, or a positive sum from two negative operands.

Four source operand addressing modes are allowed: register, direct, register-indirect, or immediate.

Example: The Accumulator holds 0C3H (11000011B) and register 0 holds 0AAH (10101010B). The instruction,

ADD A,R0

will leave 6DH (01101101B) in the Accumulator with the AC flag cleared and both the carry flag and OV set to 1.

ADD A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 0 1 0	1 r r r

Operation: ADD
(A) ← (A) + (Rn)

ADD A,direct

Bytes: 2

Cycles: 1

Encoding:

0 0 1 0	0 1 0 1	direct address

Operation: ADD
(A) ← (A) + (direct)

ADD A,@Ri

Bytes: 1

Cycles: 1

Encoding:

0 0 1 0	0 1 1 i

Operation: ADD
(A) ← (A) + ((Ri))

ADD A,#data

Bytes: 2

Cycles: 1

Encoding:

0 0 1 0	0 1 0 0	immediate data

Operation: ADD
(A) ← (A) + #data

ADDC A,<src-byte>

Function: Add with Carry

Description: ADDC simultaneously adds the byte variable indicated, the carry flag and the Accumulator contents, leaving the result in the Accumulator. The carry and auxiliary-carry flags are set, respectively, if there is a carry-out from bit 7 or bit 3, and cleared otherwise. When adding unsigned integers, the carry flag indicates an overflow occured.

OV is set if there is a carry-out of bit 6 but not out of bit 7, or a carry-out of bit 7 but not out of bit 6; otherwise OV is cleared. When adding signed integers, OV indicates a negative number produced as the sum of two positive operands or a positive sum from two negative operands.

Four source operand addressing modes are allowed: register, direct, register-indirect, or immediate.

Example: The Accumulator holds 0C3H (11000011B) and register 0 holds 0AAH (10101010B) with the carry flag set. The instruction,

ADDC A,R0

will leave 6EH (01101110B) in the Accumulator with AC cleared and both the Carry flag and OV set to 1.

ANL A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 1 0 1	1 r r r

Operation: ANL
(A) ← (A) ∧ (Rn)

ANL A,direct

Bytes: 2

Cycles: 1

Encoding:

0 1 0 1	0 1 0 1	direct address

Operation: ANL

(A) ← (A) ∧ (direct)

ANL A,@Ri

Bytes: 1

Cycles: 1

Encoding:

0 1 0 1	0 1 1 i

Operation: ANL

(A) ← (A) ∧ ((Ri))

ANL A,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 0 1	0 1 0 0	immediate data

Operation: ANL
(A) ← (A) ∧ #data

ANL direct,A

Bytes: 2

Cycles: 1

Encoding:

0 1 0 1	0 0 1 0	direct address

Operation: ANL
(direct) ← (direct) ∧ (A)

ANL direct,#data

Bytes: 3

Cycles: 2

Encoding:

0 1 0 1	0 0 1 1	direct address	immediate data

Operation: ANL
(direct) ← (direct) ∧ #data

ANL C,<src-bit>

Function: Logical-AND for bit variables

Description: If the Boolean value of the source bit is a logical 0 then clear the carry flag; otherwise leave the carry flag in its current state. A slash ("/") preceding the operand in the assembly language indicates that the logical complement of the addressed bit is used as the source value, *but the source bit itself is not affected.* No other flags are affected.

Example: Only direct addressing is allowed for the source operand.
Set the carry flag if, and only if, P1.0 = 1, ACC. 7 = 1, and OV = 0:

```
MOV  C,P1.0     ;LOAD CARRY WITH INPUT PIN STATE
ANL  C,ACC.7    ;AND CARRY WITH ACCUM. BIT 7
ANL  C,/OV      ;AND WITH INVERSE OF OVERFLOW FLAG
```

ANL C,bit

Bytes: 2

Cycles: 2

Encoding:

1 0 0 0	0 0 1 0	bit address

Operation: ANL
(C) ← (C) ∧ (bit)

ANL C,/bit

Bytes: 2

Cycles: 2

Encoding:

1 0 1 1	0 0 0 0	bit address

Operation: ANL
(C) ← (C) ∧ ¬(bit)

6

CJNE <dest-byte>,<src-byte>,rel

Function: Compare and Jump if Not Equal.

Description: CJNE compares the magnitudes of the first two operands, and branches if their values are not equal. The branch destination is computed by adding the signed relative-displacement in the last instruction byte to the PC, after incrementing the PC to the start of the next instruction. The carry flag is set if the unsigned integer value of <dest-byte> is less than the unsigned integer value of <src-byte>; otherwise, the carry is cleared. Neither operand is affected.

The first two operands allow four addressing mode combinations: the Accumulator may be compared with any directly addressed byte or immediate data, and any indirect RAM location or working register can be compared with an immediate constant.

Example: The Accumulator contains 34H. Register 7 contains 56H. The first instruction in the sequence,

```
                CJNE   R7,#60H, NOT_EQ
;               ...    .....          ; R7 = 60H.
NOT_EQ:         JC     REQ_LOW        ; IF R7 < 60H.
;               ...    .....          ; R7 > 60H.
```

sets the carry flag and branches to the instruction at label NOT_EQ. By testing the carry flag, this instruction determines whether R7 is greater or less than 60H.

If the data being presented to Port 1 is also 34H, then the instruction,

```
WAIT:  CJNE  A,P1,WAIT
```

clears the carry flag and continues with the next instruction in sequence, since the Accumulator does equal the data read from P1. (If some other value was being input on P1, the program will loop at this point until the P1 data changes to 34H.)

CJNE A,direct,rel

Bytes: 3

Cycles: 2

Encoding:

1 0 1 1	0 1 0 1	direct address	rel. address

Operation:

```
(PC) ← (PC) + 3
IF (A) <> (direct)
THEN
        (PC) ← (PC) + relative offset

IF (A) < (direct)
THEN
        (C) ← 1
ELSE
        (C) ← 0
```


CJNE A,#data,rel

Bytes: 3

Cycles: 2

Encoding:

1 0 1 1	0 1 0 0	immediate data	rel. address

Operation:

```
(PC) ← (PC) + 3
IF (A) <> data
THEN
        (PC) ← (PC) + relative offset

IF (A) < data
THEN
        (C) ← 1
ELSE
        (C) ← 0
```

CJNE Rn,#data,rel

Bytes: 3

Cycles: 2

Encoding:

1 0 1 1	1 r r r	immediate data	rel. address

Operation:

```
(PC) ← (PC) + 3
IF (Rn) <> data
THEN
        (PC) ← (PC) + relative offset

IF (Rn) < data
THEN
        (C) ← 1
ELSE
        (C) ← 0
```

CJNE @Ri,#data,rel

Bytes: 3

Cycles: 2

Encoding:

1 0 1 1	0 1 1 i	immediate data	rel. address

Operation:

```
(PC) ← (PC) + 3
IF ((Ri)) <> data
THEN
        (PC) ← (PC) + relative offset

IF ((Ri)) < data
THEN
        (C) ← 1
ELSE
        (C) ← 0
```

CLR A

Function: Clear Accumulator

Description: The Accumulator is cleared (all bits set on zero). No flags are affected.

Example: The Accumulator contains 5CH (01011100B). The instruction,

CLR A

will leave the Accumulator set to 00H (00000000B).

Bytes: 1

Cycles: 1

Encoding:

1 1 1 0	0 1 0 0

Operation:
CLR
(A) ← 0

CLR bit

Function: Clear bit

Description: The indicated bit is cleared (reset to zero). No other flags are affected. CLR can operate on the carry flag or any directly addressable bit.

Example: Port 1 has previously been written with 5DH (01011101B). The instruction,

CLR P1.2

will leave the port set to 59H (01011001B).

CLR C

Bytes: 1

Cycles: 1

Encoding:

1 1 0 0	0 0 1 1

Operation:
CLR
(C) ← 0

CLR bit

Bytes: 2

Cycles: 1

Encoding:

1 1 0 0	0 0 1 0	bit address

Operation:
CLR
(bit) ← 0

CPL A

Function: Complement Accumulator

Description: Each bit of the Accumulator is logically complemented (one's complement). Bits which previously contained a one are changed to a zero and vice-versa. No flags are affected.

Example: The Accumulator contains 5CH (01011100B). The instruction,

CPL A

will leave the Accumulator set to 0A3H (10100011B).

Bytes: 1

Cycles: 1

Encoding:

1 1 1 1	0 1 0 0

Operation:
CPL
(A) ← ¬(A)

CPL bit

Function: Complement bit

Description: The bit variable specified is complemented. A bit which had been a one is changed to zero and vice-versa. No other flags are affected. CLR can operate on the carry or any directly addressable bit.

Note: When this instruction is used to modify an output pin, the value used as the original data will be read from the output data latch, *not* the input pin.

Example: Port 1 has previously been written with 5BH (01011101B). The instruction sequence,

CPL P1.1

CPL P1.2

will leave the port set to 5BH (01011011B).

CPL C

Bytes: 1

Cycles: 1

Encoding:

1 0 1 1	0 0 1 1

Operation:
CPL
(C) ← ¬(C)

CPL bit

Bytes: 2

Cycles: 1

Encoding:

1 0 1 1	0 0 1 0	bit address

Operation: CPL
(bit) ← ¬ (bit)

DA A

Function: Decimal-adjust Accumulator for Addition

Description: DA A adjusts the eight-bit value in the Accumulator resulting from the earlier addition of two variables (each in packed-BCD format), producing two four-bit digits. Any ADD or ADDC instruction may have been used to perform the addition.

If Accumulator bits 3-0 are greater than nine (xxxx1010-xxxx1111), or if the AC flag is one, six is added to the Accumulator producing the proper BCD digit in the low-order nibble. This internal addition would set the carry flag if a carry-out of the low-order four-bit field propagated through all high-order bits, but it would not clear the carry flag otherwise.

If the carry flag is now set, or if the four high-order bits now exceed nine (1010xxxx-111xxxx), these high-order bits are incremented by six, producing the proper BCD digit in the high-order nibble. Again, this would set the carry flag if there was a carry-out of the high-order bits, but wouldn't clear the carry. The carry flag thus indicates if the sum of the original two BCD variables is greater than 100, allowing multiple precision decimal addition. OV is not affected.

All of this occurs during the one instruction cycle. Essentially, this instruction performs the decimal conversion by adding 00H, 06H, 60H, or 66H to the Accumulator, depending on initial Accumulator and PSW conditions.

Note: DA A *cannot* simply convert a hexadecimal number in the Accumulator to BCD notation, nor does DA A apply to decimal subtraction.

Example: The Accumulator holds the value 56H (01010110B) representing the packed BCD digits of the decimal number 56. Register 3 contains the value 67H (01100111B) representing the packed BCD digits of the decimal number 67. The carry flag is set. The instruction sequence.

```
ADDC  A,R3
DA    A
```

will first perform a standard twos-complement binary addition, resulting in the value 0BEH (10111110) in the Accumulator. The carry and auxiliary carry flags will be cleared.

The Decimal Adjust instruction will then alter the Accumulator to the value 24H (00100100B), indicating the packed BCD digits of the decimal number 24, the low-order two digits of the decimal sum of 56, 67, and the carry-in. The carry flag will be set by the Decimal Adjust instruction, indicating that a decimal overflow occurred. The true sum 56, 67, and 1 is 124.

BCD variables can be incremented or decremented by adding 01H or 99H. If the Accumulator initially holds 30H (representing the digits of 30 decimal), then the instruction sequence,

```
ADD   A,#99H

DA    A
```

will leave the carry set and 29H in the Accumulator, since 30 + 99 = 129. The low-order byte of the sum can be interpreted to mean 30 − 1 = 29.

Bytes: 1

Cycles: 1

Encoding:

1 1 0 1	0 1 0 0

Operation: DA
-contents of Accumulator are BCD
IF [[(A_{3-0}) > 9] ∨ [(AC) = 1]]
THEN(A_{3-0}) ← (A_{3-0}) + 6
AND

IF [[(A_{7-4}) > 9] ∨ [(C) = 1]]
THEN (A_{7-4}) ← (A_{7-4}) + 6

6

DEC byte

Function: Decrement

Description: The variable indicated is decremented by 1. An original value of 00H will underflow to 0FFH. No flags are affected. Four operand addressing modes are allowed: accumulator, register, direct, or register-indirect.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: Register 0 contains 7FH (01111111B). Internal RAM locations 7EH and 7FH contain 00H and 40H, respectively. The instruction sequence,

```
DEC  @R0
DEC  R0
DEC  @R0
```

will leave register 0 set to 7EH and internal RAM locations 7EH and 7FH set to 0FFH and 3FH.

DEC A

Bytes: 1

Cycles: 1

Encoding:

0 0 0 1	0 1 0 0

Operation: DEC
(A) ← (A) − 1

DEC Rn

Bytes: 1

Cycles: 1

Encoding:

0 0 0 1	1 r r r

Operation: DEC
(Rn) ← (Rn) − 1

DEC direct

Bytes: 2

Cycles: 1

Encoding:

0 0 0 1	0 1 0 1	direct address

Operation: DEC
(direct) ← (direct) − 1

DEC @Ri

Bytes: 1

Cycles: 1

Encoding:

0 0 0 1	0 1 1 i

Operation: DEC
((Ri)) ← ((Ri)) − 1

DIV AB

Function: Divide

Description: DIV AB divides the unsigned eight-bit integer in the Accumulator by the unsigned eight-bit integer in register B. The Accumulator receives the integer part of the quotient; register B receives the integer remainder. The carry and OV flags will be cleared.

Exception: if B had originally contained 00H, the values returned in the Accumulator and B-register will be undefined and the overflow flag will be set. The carry flag is cleared in any case.

Example: The Accumulator contains 251 (0FBH or 11111011B) and B contains 18 (12H or 00010010B). The instruction,

```
DIV  AB
```

will leave 13 in the Accumulator (0DH or 00001101B) and the value 17 (11H or 00010001B) in B, since 251 = (13 X 18) + 17. Carry and OV will both be cleared.

Bytes: 1

Cycles: 4

Encoding:

1 0 0 0	0 1 0 0

Operation: DIV
$(A)_{15-8}$
$(B)_{7-0}$ ← (A)/(B)

6

DJNZ <byte>,<rel-addr>

Function: Decrement and Jump if Not Zero

Description: DJNZ decrements the location indicated by 1, and branches to the address indicated by the second operand if the resulting value is not zero. An original value of 00H will underflow to 0FFH. No flags are affected. The branch destination would be computed by adding the signed relative-displacement value in the last instruction byte to the PC, after incrementing the PC to the first byte of the following instruction.

The location decremented may be a register or directly addressed byte.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: Internal RAM locations 40H, 50H, and 60H contain the values 01H, 70H, and 15H, respectively. The instruction sequence,

```
DJNZ  40H,LABEL_1
DJNZ  50H,LABEL_2
DJNZ  60H,LABEL_3
```

will cause a jump to the instruction at label LABEL_2 with the values 00H, 6FH, and 15H in the three RAM locations. The first jump was *not* taken because the result was zero.

This instruction provides a simple way of executing a program loop a given number of times, or for adding a moderate time delay (from 2 to 512 machine cycles) with a single instruction. The instruction sequence,

```
        MOV     R2,#8
TOGGLE: CPL     P1.7
        DJNZ    R2,TOGGLE
```

will toggle P1.7 eight times, causing four output pulses to appear at bit 7 of output Port 1. Each pulse will last three machine cycles; two for DJNZ and one to alter the pin.

DJNZ Rn,rel

Bytes: 2

Cycles: 2

Encoding:

1 1 0 1	1 r r r	rel. address

Operation:
```
DJNZ
(PC) ← (PC) + 2
(Rn) ← (Rn) − 1
IF  (Rn) > 0 or (Rn) < 0
    THEN
        (PC) ← (PC) + rel
```


DJNZ direct,rel

Bytes: 3

Cycles: 2

Encoding:

1 1 0 1	0 1 0 1	direct address	rel. address

Operation:
```
DJNZ
(PC) ← (PC) + 2
(direct) ← (direct) − 1
IF (direct) > 0 or (direct) < 0
   THEN
        (PC) ← (PC) + rel
```

INC <byte>

Function: Increment

Description: INC increments the indicated variable by 1. An original value of 0FFH will overflow to 00H. No flags are affected. Three addressing modes are allowed: register, direct, or register-indirect.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: Register 0 contains 7EH (01111110B). Internal RAM locations 7EH and 7FH contain 0FFH and 40H, respectively. The instruction sequence,

```
INC  @R0
INC  R0
INC  @R0
```

will leave register 0 set to 7FH and internal RAM locations 7EH and 7FH holding (respectively) 00H and 41H.

INC A

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	0 1 0 0

Operation:
```
INC
(A) ← (A) + 1
```

6

INC Rn

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	1 r r r

Operation: INC
(Rn) ← (Rn) + 1

INC direct

Bytes: 2

Cycles: 1

Encoding:

0 0 0 0	0 1 0 1	direct address

Operation: INC
(direct) ← (direct) + 1

INC @Ri

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	0 1 1 i

Operation: INC
((Ri)) ← ((Ri)) + 1

INC DPTR

Function: Increment Data Pointer

Description: Increment the 16-bit data pointer by 1. A 16-bit increment (modulo 2^{16}) is performed; an overflow of the low-order byte of the data pointer (DPL) from 0FFH to 00H will increment the high-order byte (DPH). No flags are affected.

This is the only 16-bit register which can be incremented.

Example: Registers DPH and DPL contain 12H and 0FEH, respectively. The instruction sequence,

```
INC  DPTR
INC  DPTR
INC  DPTR
```

will change DPH and DPL to 13H and 01H.

Bytes: 1

Cycles: 2

Encoding:

1 0 1 0	0 0 1 1

Operation: INC
(DPTR) ← (DPTR) + 1

JB bit,rel

Function: Jump if Bit set

Description: If the indicated bit is a one, jump to the address indicated; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. *The bit tested is not modified.* No flags are affected.

Example: The data present at input port 1 is 11001010B. The Accumulator holds 56 (01010110B). The instruction sequence,

```
JB   P1.2,LABEL1
JB   ACC.2,LABEL2
```

will cause program execution to branch to the instruction at label LABEL2.

Bytes: 3

Cycles: 2

Encoding:

0 0 1 0	0 0 0 0	bit address	rel. address

Operation: JB
(PC) ← (PC) + 3
IF (bit) = 1
THEN
(PC) ← (PC) + rel

JBC bit,rel

Function: Jump if Bit is set and Clear bit

Description: If the indicated bit is one, branch to the address indicated; otherwise proceed with the next instruction. *The bit will not be cleared if it is already a zero.* The branch destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. No flags are affected.

Note: When this instruction is used to test an output pin, the value used as the original data will be read from the output data latch, *not* the input pin.

Example: The Accumulator holds 56H (01010110B). The instruction sequence,

```
JBC  ACC.3,LABEL1
JBC  ACC.2,LABEL2
```

will cause program execution to continue at the instruction identified by the label LABEL2, with the Accumulator modified to 52H (01010010B).

Bytes: 3

Cycles: 2

Encoding:

0 0 0 1	0 0 0 0	bit address	rel. address

Operation:

```
JBC
(PC) ← (PC) + 3
IF  (bit) = 1
    THEN
        (bit) ← 0
        (PC) ← (PC) + rel
```

JC rel

Function: Jump if Carry is set

Description: If the carry flag is set, branch to the address indicated; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice. No flags are affected.

Example: The carry flag is cleared. The instruction sequence,

```
JC    LABEL1
CPL   C
JC    LABEL 2
```

will set the carry and cause program execution to continue at the instruction identified by the label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0 1 0 0	0 0 0 0	rel. address

Operation:

```
JC
(PC) ← (PC) + 2
IF  (C) = 1
    THEN
        (PC) ← (PC) + rel
```


JMP @A+DPTR

Function: Jump indirect

Description: Add the eight-bit unsigned contents of the Accumulator with the sixteen-bit data pointer, and load the resulting sum to the program counter. This will be the address for subsequent instruction fetches. Sixteen-bit addition is performed (modulo 2^{16}): a carry-out from the low-order eight bits propagates through the higher-order bits. Neither the Accumulator nor the Data Pointer is altered. No flags are affected.

Example: An even number from 0 to 6 is in the Accumulator. The following sequence of instructions will branch to one of four AJMP instructions in a jump table starting at JMP_TBL:

```
            MOV   DPTR,#JMP_TBL
            JMP   @A+DPTR
JMP_TBL:    AJMP  LABEL0
            AJMP  LABEL1
            AJMP  LABEL2
            AJMP  LABEL3
```

If the Accumulator equals 04H when starting this sequence, execution will jump to label LABEL2. Remember that AJMP is a two-byte instruction, so the jump instructions start at every other address.

Bytes: 1

Cycles: 2

Encoding:

0 1 1 1	0 0 1 1

Operation:

```
JMP
(PC) ← (A) + (DPTR)
```

6

JNB bit,rel

Function: Jump if Bit Not set

Description: If the indicated bit is a zero, branch to the indicated address; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. *The bit tested is not modified.* No flags are affected.

Example: The data present at input port 1 is 11001010B. The Accumulator holds 56H (01010110B). The instruction sequence,

```
JNB  P1.3,LABEL1
JNB  ACC.3,LABEL2
```

will cause program execution to continue at the instruction at label LABEL2.

Bytes: 3

Cycles: 2

Encoding:

0 0 1 1	0 0 0 0	bit address	rel. address

Operation:

```
JNB
(PC) ← (PC) + 3
IF  (bit) = 0
    THEN (PC) ← (PC) + rel.
```

JNC rel

Function: Jump if Carry not set

Description: If the carry flag is a zero, branch to the address indicated; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice to point to the next instruction. The carry flag is not modified.

Example: The carry flag is set. The instruction sequence,

```
JNC  LABEL1
CPL  C
JNC  LABEL2
```

will clear the carry and cause program execution to continue at the instruction identified by the label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0 1 0 1	0 0 0 0	rel. address

Operation:

```
JNC
(PC) ← (PC) + 2
IF  (C) = 0
    THEN  (PC) ← (PC) + rel
```


JNZ rel

Function: Jump if Accumulator Not Zero

Description: If any bit of the Accumulator is a one, branch to the indicated address; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice. The Accumulator is not modified. No flags are affected.

Example: The Accumulator originally holds 00H. The instruction sequence,

```
JNZ  LABEL1
INC  A
JNZ  LABEL2
```

will set the Accumulator to 01H and continue at label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0 1 1 1	0 0 0 0	rel. address

Operation:

```
JNZ
(PC) ← (PC) + 2
IF   (A) ≠ 0
     THEN  (PC) ← (PC) + rel
```

JZ rel

Function: Jump if Accumulator Zero

Description: If all bits of the Accumulator are zero, branch to the address indicated; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice. The Accumulator is not modified. No flags are affected.

Example: The Accumulator originally contains 01H. The instruction sequence,

```
JZ   LABEL1
DEC  A
JZ   LABEL2
```

will change the Accumulator to 00H and cause program execution to continue at the instruction identified by the label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0 1 1 0	0 0 0 0	rel. address

Operation:

```
JZ
(PC) ← (PC) + 2
IF   (A) = 0
     THEN  (PC) ← (PC) + rel
```

LCALL addr16

Function: Long call

Description: LCALL calls a subroutine located at the indicated address. The instruction adds three to the program counter to generate the address of the next instruction and then pushes the 16-bit result onto the stack (low byte first), incrementing the Stack Pointer by two. The high-order and low-order bytes of the PC are then loaded, respectively, with the second and third bytes of the LCALL instruction. Program execution continues with the instruction at this address. The subroutine may therefore begin anywhere in the full 64K-byte program memory address space. No flags are affected.

Example: Initially the Stack Pointer equals 07H. The label "SUBRTN" is assigned to program memory location 1234H. After executing the instruction,

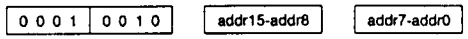
```
LCALL   SUBRTN
```

at location 0123H, the Stack Pointer will contain 09H, internal RAM locations 08H and 09H will contain 26H and 01H, and the PC will contain 1234H.

Bytes: 3

Cycles: 2

Encoding:



Operation:

```
LCALL
(PC) ← (PC) + 3
(SP) ← (SP) + 1
((SP)) ← (PC7-0)
(SP) ← (SP) + 1
((SP)) ← (PC15-8)
(PC) ← addr15-0
```

LJMP addr16

Function: Long Jump

Description: LJMP causes an unconditional branch to the indicated address, by loading the high-order and low-order bytes of the PC (respectively) with the second and third instruction bytes. The destination may therefore be anywhere in the full 64K program memory address space. No flags are affected.

Example: The label "JMPADR" is assigned to the instruction at program memory location 1234H. The instruction,

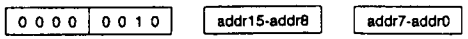
```
LJMP   JMPADR
```

at location 0123H will load the program counter with 1234H.

Bytes: 3

Cycles: 2

Encoding:



Operation:

```
LJMP
(PC) ← addr15-0
```


MOV <dest-byte>,<src-byte>

Function: Move byte variable

Description: The byte variable indicated by the second operand is copied into the location specified by the first operand. The source byte is not affected. No other register or flag is affected.

This is by far the most flexible operation. Fifteen combinations of source and destination addressing modes are allowed.

Example: Internal RAM location 30H holds 40H. The value of RAM location 40H is 10H. The data present at input port 1 is 11001010B (0CAH).

```
MOV   R0,#30H   ;R0 <= 30H
MOV   A,@R0     ;A <= 40H
MOV   R1,A      ;R1 <= 40H
MOV   B,@R1     ;B <= 10H
MOV   @R1,P1    ;RAM (40H) <= 0CAH
MOV   P2,P1     ;P2 #0CAH
```

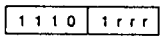
leaves the value 30H in register 0, 40H in both the Accumulator and register 1, 10H in register B, and 0CAH (11001010B) both in RAM location 40H and output on port 2.

MOV A,Rn

Bytes: 1

Cycles: 1

Encoding:



Operation:

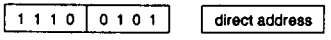
```
MOV
(A) ← (Rn)
```

*MOV A,direct

Bytes: 2

Cycles: 1

Encoding:



Operation:

```
MOV
(A) ← (direct)
```

MOV A,ACC is not a valid instruction.

MOV A,@RI

Bytes: 1

Cycles: 1

Encoding:

1 1 1 0	0 1 1 i

Operation: MOV
(A) ← ((Ri))

MOV A,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 1 1	0 1 0 0	immediate data

Operation: MOV
(A) ← #data

MOV Rn,A

Bytes: 1

Cycles: 1

Encoding:

1 1 1 1	1 r r r

Operation: MOV
(Rn) ← (A)

MOV Rn,direct

Bytes: 2

Cycles: 2

Encoding:

1 0 1 0	1 r r r	direct addr.

Operation: MOV
(Rn) ← (direct)

MOV Rn,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 1 1	1 r r r	immediate data

Operation: MOV
(Rn) ← #data

MOV direct,A

Bytes: 2

Cycles: 1

Encoding:

1 1 1 1	0 1 0 1	direct address

Operation: MOV
(direct) ← (A)

MOV direct,Rn

Bytes: 2

Cycles: 2

Encoding:

1 0 0 0	1 r r r	direct address

Operation: MOV
(direct) ← (Rn)

MOV direct,direct

Bytes: 3

Cycles: 2

Encoding:

1 0 0 0	0 1 0 1	dir. addr. (src)	dir. addr. (dest)

Operation: MOV
(direct) ← (direct)

MOV direct,@RI

Bytes: 2

Cycles: 2

Encoding:

1 0 0 0	0 1 1 i	direct addr.

Operation: MOV
(direct) ← ((Ri))

MOV direct,#data

Bytes: 3

Cycles: 2

Encoding:

0 1 1 1	0 1 0 1	direct address	immediate data

Operation: MOV
(direct) ← #data

MOV @Ri,A

Bytes: 1

Cycles: 1

Encoding: | 1 1 1 1 | 0 1 1 i |

Operation: MOV
((Ri)) ← (A)

MOV @Ri,direct

Bytes: 2

Cycles: 2

Encoding: | 1 0 1 0 | 0 1 1 i | direct addr. |

Operation: MOV
((Ri)) ← (direct)

MOV @Ri,#data

Bytes: 2

Cycles: 1

Encoding: | 0 1 1 1 | 0 1 1 i | immediate data |

Operation: MOV
((Ri)) ← #data

MOV <dest-bit>,<src-bit>

Function: Move bit data

Description: The Boolean variable indicated by the second operand is copied into the location specified by the first operand. One of the operands must be the carry flag; the other may be any directly addressable bit. No other register or flag is affected.

Example: The carry flag is originally set. The data present at input Port 3 is 11000101B. The data previously written to output Port 1 is 35H (00110101B).

```
MOV  P1.3,C
MOV  C,P3.3
MOV  P1.2,C
```

will leave the carry cleared and change Port 1 to 39H (00111001B).

MOV C,bit

Bytes: 2

Cycles: 1

Encoding: | 1 0 1 0 | 0 0 1 0 | bit address |

Operation: MOV
(C) ← (bit)

MOV bit,C

Bytes: 2

Cycles: 2

Encoding: | 1 0 0 1 | 0 0 1 0 | bit address |

Operation: MOV
(bit) ← (C)

MOV DPTR,#data16

Function: Load Data Pointer with a 16-bit constant

Description: The Data Pointer is loaded with the 16-bit constant indicated. The 16-bit constant is loaded into the second and third bytes of the instruction. The second byte (DPH) is the high-order byte, while the third byte (DPL) holds the low-order byte. No flags are affected.

This is the only instruction which moves 16 bits of data at once.

Example: The instruction,

MOV DPTR,#1234H

will load the value 1234H into the Data Pointer: DPH will hold 12H and DPL will hold 34H.

Bytes: 3

Cycles: 2

Encoding: | 1 0 0 1 | 0 0 0 0 | immed. data15-8 | immed. data7-0 |

Operation: MOV
(DPTR) ← $\#data_{15-0}$
DPH ☐ DPL ← $\#data_{15-8}$ ☐ $\#data_{7-0}$

MOVC A,@A+<base-reg>

Function: Move Code byte

Description: The MOVC instructions load the Accumulator with a code byte, or constant from program memory. The address of the byte fetched is the sum of the original unsigned eight-bit Accumulator contents and the contents of a sixteen-bit base register, which may be either the Data Pointer or the PC. In the latter case, the PC is incremented to the address of the following instruction before being added with the Accumulator; otherwise the base register is not altered. Sixteen-bit addition is performed so a carry-out from the low-order eight bits may propagate through higher-order bits. No flags are affected.

Example: A value between 0 and 3 is in the Accumulator. The following instructions will translate the value in the Accumulator to one of four values defined by the DB (define byte) directive.

```
REL_PC: INC   A
        MOVC  A,@A+PC
        RET
        DB    66H
        DB    77H
        DB    88H
        DB    99H
```

If the subroutine is called with the Accumulator equal to 01H, it will return with 77H in the Accumulator. The INC A before the MOVC instruction is needed to "get around" the RET instruction above the table. If several bytes of code separated the MOVC from the table, the corresponding number would be added to the Accumulator instead.

MOVC A,@A+DPTR

Bytes: 1

Cycles: 2

Encoding:

1 0 0 1	0 0 1 1

Operation: MOVC
(A) ← ((A) + (DPTR))

MOVC A,@A + PC

Bytes: 1

Cycles: 2

Encoding:

1 0 0 0	0 0 1 1

Operation: MOVC
(PC) ← (PC) + 1
(A) ← ((A) + (PC))

MOVX <dest-byte>,<src-byte>

Function: Move External

Description: The MOVX instructions transfer data between the Accumulator and a byte of external data memory, hence the "X" appended to MOV. There are two types of instructions, differing in whether they provide an eight-bit or sixteen-bit indirect address to the external data RAM.

In the first type, the contents of R0 or R1 in the current register bank provide an eight-bit address multiplexed with data on P0. Eight bits are sufficient for external I/O expansion decoding or for a relatively small RAM array. For somewhat larger arrays, any output port pins can be used to output higher-order address bits. These pins would be controlled by an output instruction preceding the MOVX.

In the second type of MOVX instruction, the Data Pointer generates a sixteen-bit address. P2 outputs the high-order eight address bits (the contents of DPH) while P0 multiplexes the low-order eight bits (DPL) with data. The P2 Special Function Register retains its previous contents while the P2 output buffers are emitting the contents of DPH. This form is faster and more efficient when accessing very large data arrays (up to 64K bytes), since no additional instructions are needed to set up the output ports.

It is possible in some situations to mix the two MOVX types. A large RAM array with its high-order address lines driven by P2 can be addressed via the Data Pointer, or with code to output high-order address bits to P2 followed by a MOVX instruction using R0 or R1.

Example: An external 256 byte RAM using multiplexed address/data lines (e.g., an Intel 8155 RAM/I/O/Timer) is connected to the 8051 Port 0. Port 3 provides control lines for the external RAM. Ports 1 and 2 are used for normal I/O. Registers 0 and 1 contain 12H and 34H. Location 34H of the external RAM holds the value 56H. The instruction sequence,

```
MOVX  A,@R1
MOVX  @R0,A
```

copies the value 56H into both the Accumulator and external RAM location 12H.

MOVX A,@RI

Bytes: 1

Cycles: 2

Encoding:

1 1 1 0	0 0 1 i

Operation: MOVX
(A) ← ((Ri))

MOVX A,@DPTR

Bytes: 1

Cycles: 2

Encoding:

1 1 1 0	0 0 0 0

Operation: MOVX
(A) ← ((DPTR))

MOVX @RI,A

Bytes: 1

Cycles: 2

Encoding:

1 1 1 1	0 0 1 i

Operation: MOVX
((Ri)) ← (A)

MOVX @DPTR,A

Bytes: 1

Cycles: 2

Encoding:

1 1 1 1	0 0 0 0

Operation: MOVX
(DPTR) ← (A)

MUL AB

Function: Multiply

Description: MUL AB multiplies the unsigned eight-bit integers in the Accumulator and register B. The low-order byte of the sixteen-bit product is left in the Accumulator, and the high-order byte in B. If the product is greater than 255 (0FFH) the overflow flag is set; otherwise it is cleared. The carry flag is always cleared.

Example: Originally the Accumulator holds the value 80 (50H). Register B holds the value 160 (0A0H). The instruction,

MUL AB

will give the product 12,800 (3200H), so B is changed to 32H (00110010B) and the Accumulator is cleared. The overflow flag is set, carry is cleared.

Bytes: 1

Cycles: 4

Encoding:

1 0 1 0	0 1 0 0

Operation: MUL
$(A)_{7-0}$ ← (A) X (B)
$(B)_{15-8}$

NOP

Function: No Operation

Description: Execution continues at the following instruction. Other than the PC, no registers or flags are affected.

Example: It is desired to produce a low-going output pulse on bit 7 of Port 2 lasting exactly 5 cycles. A simple SETB/CLR sequence would generate a one-cycle pulse, so four additional cycles must be inserted. This may be done (assuming no interrupts are enabled) with the instruction sequence,

```
CLR   P2.7
NOP
NOP
NOP
NOP
SETB  P2.7
```

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	0 0 0 0

Operation: NOP
(PC) ← (PC) + 1

ORL <dest-byte> <src-byte>

Function: Logical-OR for byte variables

Description: ORL performs the bitwise logical-OR operation between the indicated variables, storing the results in the destination byte. No flags are affected.

The two operands allow six addressing mode combinations. When the destination is the Accumulator, the source can use register, direct, register-indirect, or immediate addressing; when the destination is a direct address, the source can be the Accumulator or immediate data.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: If the Accumulator holds 0C3H (11000011B) and R0 holds 55H (01010101B) then the instruction,

ORL A,R0

will leave the Accumulator holding the value 0D7H (11010111B).

When the destination is a directly addressed byte, the instruction can set combinations of bits in any RAM location or hardware register. The pattern of bits to be set is determined by a mask byte, which may be either a constant data value in the instruction or a variable computed in the Accumulator at run-time. The instruction,

ORL P1,#00110010B

will set bits 5, 4, and 1 of output Port 1.

ORL A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 1 0 0	1 r r r

Operation: ORL
(A) ← (A) V (Rn)

ORL A,direct

Bytes: 2

Cycles: 1

Encoding:

0 1 0 0	0 1 0 1	direct address

Operation: ORL
(A) ← (A) V (direct)

ORL A,@Ri

Bytes: 1

Cycles: 1

Encoding:

0 1 0 0	0 1 1 i

Operation: ORL
(A) ← (A) V ((Ri))

ORL A,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 0 0	0 1 0 0	immediate data

Operation: ORL
(A) ← (A) V #data

ORL direct,A

Bytes: 2

Cycles: 1

Encoding:

0 1 0 0	0 0 1 0	direct address

Operation: ORL
(direct) ← (direct) V (A)

ORL direct,#data

Bytes: 3

Cycles: 2

Encoding:

0 1 0 0	0 0 1 1	direct addr.	immediate data

Operation: ORL
(direct) ← (direct) V #data

ORL C,<src-bit>

Function: Logical-OR for bit variables

Description: Set the carry flag if the Boolean value is a logical 1; leave the carry in its current state otherwise . A slash ("/") preceding the operand in the assembly language indicates that the logical complement of the addressed bit is used as the source value, but the source bit itself is not affected. No other flags are affected.

Example: Set the carry flag if and only if P1.0 = 1, ACC. 7 = 1, or OV = 0:

```
MOV  C,P1.0     ;LOAD CARRY WITH INPUT PIN P10
ORL  C,ACC.7    ;OR CARRY WITH THE ACC. BIT 7
ORL  C,/OV      ;OR CARRY WITH THE INVERSE OF OV.
```

ORL C,bit

Bytes: 2

Cycles: 2

Encoding:

0 1 1 1	0 0 1 0	bit address

Operation: ORL
(C) ← (C) ∨ (bit)

ORL C,/bit

Bytes: 2

Cycles: 2

Encoding:

1 0 1 0	0 0 0 0	bit address

Operation: ORL
(C) ← (C) ∨ $(\overline{bit})$

POP direct

Function: Pop from stack.

Description: The contents of the internal RAM location addressed by the Stack Pointer is read, and the Stack Pointer is decremented by one. The value read is then transferred to the directly addressed byte indicated. No flags are affected.

Example: The Stack Pointer originally contains the value 32H, and internal RAM locations 30H through 32H contain the values 20H, 23H, and 01H, respectively. The instruction sequence,

```
POP  DPH
POP  DPL
```

will leave the Stack Pointer equal to the value 30H and the Data Pointer set to 0123H. At this point the instruction,

```
POP  SP
```

will leave the Stack Pointer set to 20H. Note that in this special case the Stack Pointer was decremented to 2FH before being loaded with the value popped (20H).

Bytes: 2

Cycles: 2

Encoding:

1 1 0 1	0 0 0 0	direct address

Operation: POP
(direct) ← ((SP))
(SP) ← (SP) − 1

PUSH direct

Function: Push onto stack

Description: The Stack Pointer is incremented by one. The contents of the indicated variable is then copied into the internal RAM location addressed by the Stack Pointer. Otherwise no flags are affected.

Example: On entering an interrupt routine the Stack Pointer contains 09H. The Data Pointer holds the value 0123H. The instruction sequence,

```
PUSH  DPL
PUSH  DPH
```

will leave the Stack Pointer set to 0BH and store 23H and 01H in internal RAM locations 0AH and 0BH, respectively.

Bytes: 2

Cycles: 2

Encoding:

1 1 0 0	0 0 0 0	direct address

Operation: PUSH
(SP) ← (SP) + 1
((SP)) ← (direct)

RET

Function: Return from subroutine

Description: RET pops the high- and low-order bytes of the PC successively from the stack, decrementing the Stack Pointer by two. Program execution continues at the resulting address, generally the instruction immediately following an ACALL or LCALL. No flags are affected.

Example: The Stack Pointer originally contains the value 0BH. Internal RAM locations 0AH and 0BH contain the values 23H and 01H, respectively. The instruction,

RET

will leave the Stack Pointer equal to the value 09H. Program execution will continue at location 0123H.

Bytes: 1

Cycles: 2

Encoding:

0 0 1 0	0 0 1 0

Operation: RET
$(PC_{15-8}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$
$(PC_{7-0}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$

RETI

Function: Return from interrupt

Description: RETI pops the high- and low-order bytes of the PC successively from the stack, and restores the interrupt logic to accept additional interrupts at the same priority level as the one just processed. The Stack Pointer is left decremented by two. No other registers are affected; the PSW is *not* automatically restored to its pre-interrupt status. Program execution continues at the resulting address, which is generally the instruction immediately after the point at which the interrupt request was detected. If a lower- or same-level interrupt had been pending when the RETI instruction is executed, that one instruction will be executed before the pending interrupt is processed.

Example: The Stack Pointer originally contains the value 0BH. An interrupt was detected during the instruction ending at location 0122H. Internal RAM locations 0AH and 0BH contain the values 23H and 01H, respectively. The instruction,

RETI

will leave the Stack Pointer equal to 09H and return program execution to location 0123H.

Bytes: 1

Cycles: 2

Encoding:

0 0 1 1	0 0 1 0

Operation: RETI
$(PC_{15-8}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$
$(PC_{7-0}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$

RL A

Function: Rotate Accumulator Left

Description: The eight bits in the Accumulator are rotated one bit to the left. Bit 7 is rotated into the bit 0 position. No flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B). The instruction,

RL A

leaves the Accumulator holding the value 8BH (10001011B) with the carry unaffected.

Bytes: 1

Cycles: 1

Encoding:

0 0 1 0	0 0 1 1

Operation: RL
$(A_n + 1) \leftarrow (An) \quad n = 0 - 6$
$(A0) \leftarrow (A7)$

RLC A

Function: Rotate Accumulator Left through the Carry flag

Description: The eight bits in the Accumulator and the carry flag are together rotated one bit to the left. Bit 7 moves into the carry flag; the original state of the carry flag moves into the bit 0 position. No other flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B), and the carry is zero. The instruction,

RLC A

leaves the Accumulator holding the value 8BH (10001010B) with the carry set.

Bytes: 1

Cycles: 1

Encoding:

0 0 1 1	0 0 1 1

Operation: RLC
$(An + 1) \leftarrow (An) \quad n = 0 - 6$
$(A0) \leftarrow (C)$
$(C) \leftarrow (A7)$

RR A

Function: Rotate Accumulator Right

Description: The eight bits in the Accumulator are rotated one bit to the right. Bit 0 is rotated into the bit 7 position. No flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B). The instruction,

RR A

leaves the Accumulator holding the value 0E2H (11100010B) with the carry unaffected.

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	0 0 1 1

Operation: RR
(An) ← (A_n + 1) n = 0 – 6
(A7) ← (A0)

RRC A

Function: Rotate Accumulator Right through Carry flag

Description: The eight bits in the Accumulator and the carry flag are together rotated one bit to the right. Bit 0 moves into the carry flag; the original value of the carry flag moves into the bit 7 position. No other flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B), the carry is zero. The instruction,

RRC A

leaves the Accumulator holding the value 62 (01100010B) with the carry set.

Bytes: 1

Cycles: 1

Encoding:

0 0 0 1	0 0 1 1

Operation: RRC
(An) ← (An + 1) n = 0 – 6
(A7) ← (C)
(C) ← (A0)

SETB <bit>

Function: Set Bit

Description: SETB sets the indicated bit to one. SETB can operate on the carry flag or any directly addressable bit. No other flags are affected.

Example: The carry flag is cleared. Output Port 1 has been written with the value 34H (00110100B). The instructions,

SETB C

SETB P1.0

will leave the carry flag set to 1 and change the data output on Port 1 to 35H (00110101B).

SETB C

Bytes: 1

Cycles: 1

Encoding:

1 1 0 1	0 0 1 1

Operation: SETB
(C) ← 1

SETB bit

Bytes: 2

Cycles: 1

Encoding:

1 1 0 1	0 0 1 0	bit address

Operation: SETB
(bit) ← 1

SJMP rel

Function: Short Jump

Description: Program control branches unconditionally to the address indicated. The branch destination is computed by adding the signed displacement in the second instruction byte to the PC, after incrementing the PC twice. Therefore, the range of destinations allowed is from 128 bytes preceding this instruction to 127 bytes following it.

Example: The label "RELADR" is assigned to an instruction at program memory location 0123H. The instruction,

SJMP RELADR

will assemble into location 0100H. After the instruction is executed, the PC will contain the value 0123H.

(*Note:* Under the above conditions the instruction following SJMP will be at 102H. Therefore, the displacement byte of the instruction will be the relative offset (0123H-0102H) = 21H. Put another way, an SJMP with a displacement of 0FEH would be a one-instruction infinite loop.)

Bytes: 2

Cycles: 2

Encoding:

1 0 0 0	0 0 0 0	rel. address

Operation: SJMP
(PC) ← (PC) + 2
(PC) ← (PC) + rel

SUBB A,<src-byte>

Function: Subtract with borrow

Description: SUBB subtracts the indicated variable and the carry flag together from the Accumulator, leaving the result in the Accumulator. SUBB sets the carry (borrow) flag if a borrow is needed for bit 7, and clears C otherwise. (If C was set *before* executing a SUBB instruction, this indicates that a borrow was needed for the previous step in a multiple precision subtraction, so the carry is subtracted from the Accumulator along with the source operand.) AC is set if a borrow is needed for bit 3, and cleared otherwise. OV is set if a borrow is needed into bit 6, but not into bit 7, or into bit 7, but not bit 6.

When subtracting signed integers OV indicates a negative number produced when a negative value is subtracted from a positive value, or a positive result when a positive number is subtracted from a negative number.

The source operand allows four addressing modes: register, direct, register-indirect, or immediate.

Example: The Accumulator holds 0C9H (11001001B), register 2 holds 54H (01010100B), and the carry flag is set. The instruction,

SUBB A,R2

will leave the value 74H (01110100B) in the accumulator, with the carry flag and AC cleared but OV set.

Notice that 0C9H minus 54H is 75H. The difference between this and the above result is due to the carry (borrow) flag being set before the operation. If the state of the carry is not known before starting a single or multiple-precision subtraction, it should be explicitly cleared by a CLR C instruction.

SUBB A,Rn

Bytes: 1

Cycles: 1

Encoding:

1 0 0 1	1 r r r

Operation: SUBB
(A) ← (A) − (C) − (Rn)

6

SUBB A,direct

Bytes: 2

Cycles: 1

Encoding:

1 0 0 1	0 1 0 1	direct address

Operation: SUBB
(A) ← (A) − (C) − (direct)

SUBB A,@Ri

Bytes: 1

Cycles: 1

Encoding:

1 0 0 1	0 1 1 i

Operation: SUBB
(A) ← (A) − (C) − ((Ri))

SUBB A,#data

Bytes: 2

Cycles: 1

Encoding:

1 0 0 1	0 1 0 0	immediate data

Operation: SUBB
(A) ← (A) − (C) − #data

SWAP A

Function: Swap nibbles within the Accumulator

Description: SWAP A interchanges the low- and high-order nibbles (four-bit fields) of the Accumulator (bits 3-0 and bits 7-4). The operation can also be thought of as a four-bit rotate instruction. No flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B). The instruction,

SWAP A

leaves the Accumulator holding the value 5CH (01011100B).

Bytes: 1

Cycles: 1

Encoding:

1 1 0 0	0 1 0 0

Operation: SWAP
$(A_{3-0}) \rightleftarrows (A_{7-4})$

XCH A,<byte>

Function: Exchange Accumulator with byte variable

Description: XCH loads the Accumulator with the contents of the indicated variable, at the same time writing the original Accumulator contents to the indicated variable. The source/destination operand can use register, direct, or register-indirect addressing.

Example: R0 contains the address 20H. The Accumulator holds the value 3FH (00111111B). Internal RAM location 20H holds the value 75H (01110101B). The instruction,

XCH A,@R0

will leave RAM location 20H holding the values 3FH (00111111B) and 75H (01110101B) in the accumulator.

XCH A,Rn

Bytes: 1

Cycles: 1

Encoding:

1 1 0 0	1 r r r

Operation: XCH
(A) $\rightleftarrows$ (Rn)

XCH A,direct

Bytes: 2

Cycles: 1

Encoding:

1 1 0 0	0 1 0 1	direct address

Operation: XCH
(A) $\rightleftarrows$ (direct)

XCH A,@Ri

Bytes: 1

Cycles: 1

Encoding:

1 1 0 0	0 1 1 i

Operation: XCH
(A) $\rightleftarrows$ ((Ri))

XCHD A,@Ri

Function: Exchange Digit

Description: XCHD exchanges the low-order nibble of the Accumulator (bits 3-0), generally representing a hexadecimal or BCD digit, with that of the internal RAM location indirectly addressed by the specified register. The high-order nibbles (bits 7-4) of each register are not affected. No flags are affected.

Example: R0 contains the address 20H. The Accumulator holds the value 36H (00110110B). Internal RAM location 20H holds the value 75H (01110101B). The instruction,

XCHD A,@R0

will leave RAM location 20H holding the value 76H (01110110B) and 35H (00110101B) in the Accumulator.

Bytes: 1

Cycles: 1

Encoding:

1 1 0 1	0 1 1 i

Operation: XCHD
$(A_{3-0}) \rightleftarrows ((Ri_{3-0}))$

XRL <dest-byte>,<src-byte>

Function: Logical Exclusive-OR for byte variables

Description: XRL performs the bitwise logical Exclusive-OR operation between the indicated variables, storing the results in the destination. No flags are affected.

The two operands allow six addressing mode combinations. When the destination is the Accumulator, the source can use register, direct, register-indirect, or immediate addressing; when the destination is a direct address, the source can be the Accumulator or immediate data.

(*Note:* When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.)

Example: If the Accumulator holds 0C3H (11000011B) and register 0 holds 0AAH (10101010B) then the instruction,

XRL A,R0

will leave the Accumulator holding the value 69H (01101001B).

When the destination is a directly addressed byte, this instruction can complement combinations of bits in any RAM location or hardware register. The pattern of bits to be complemented is then determined by a mask byte, either a constant contained in the instruction or a variable computed in the Accumulator at run-time. The instruction,

XRL P1,#00110001B

will complement bits 5, 4, and 0 of output Port 1.

XRL A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 1 1 0	1 r r r

Operation: XRL
(A) ← (A) ∀ (Rn)

XRL A,direct

Bytes: 2

Cycles: 1

Encoding:

0 1 1 0	0 1 0 1	direct address

Operation: XRL
(A) ← (A) ∀ (direct)

XRL A,@Ri

Bytes: 1

Cycles: 1

Encoding:

0 1 1 0	0 1 1 i

Operation: XRL
(A) ← (A) ∀ ((Ri))

XRL A,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 1 0	0 1 0 0	immediate data

Operation: XRL
(A) ← (A) ∀ #data

XRL direct,A

Bytes: 2

Cycles: 1

Encoding:

0 1 1 0	0 0 1 0	direct address

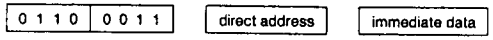
Operation: XRL
(direct) ← (direct) ∀ (A)

XRL direct, #data

Bytes: 3

Cycles: 2

Encoding:



Operation: XRL
(direct) ← (direct) ∀ #data

MCS®-51 PROGRAMMER'S GUIDE AND INSTRUCTION SET

CONTENTS

CONTENTS

CONTENTS

The information presented in this chapter is collected from the MCS®-51 Architectural Overview and the Hardware Description of the 8051, 8052 and 80C51 chapters of this book. The material has been selected and rearranged to form a quick and convenient reference for the programmers of the MCS-51. This guide pertains specifically to the 8051, 8052 and 80C51.

MEMORY ORGANIZATION

PROGRAM MEMORY

The 8051 has separate address spaces for Program Memory and Data Memory. The Program Memory can be up to 64K bytes long. The lower 4K (8K for the 8052) may reside on-chip.

Figure 1 shows a map of the 8051 program memory, and Figure 2 shows a map of the 8052 program memory.

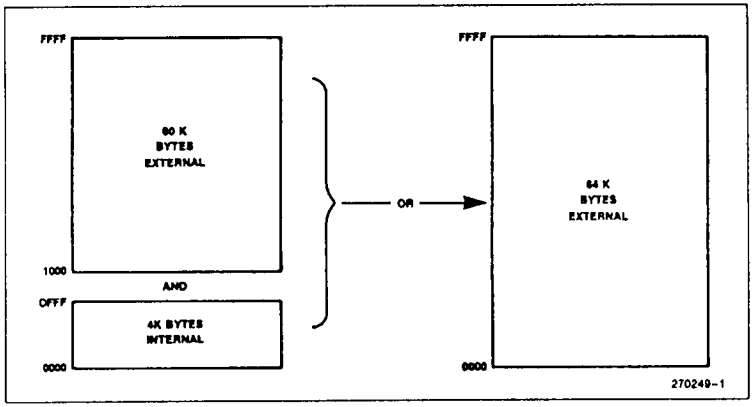


Figure 1. The 8051 Program Memory

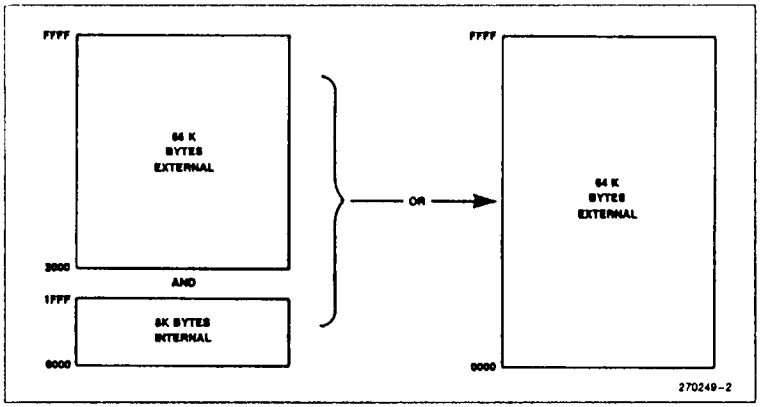


Figure 2. The 8052 Program Memory

Data Memory:

The 8051 can address up to 64K bytes of Data Memory external to the chip. The "MOVX" instruction is used to access the external data memory. (Refer to the MCS-51 Instruction Set, in this chapter, for detailed description of instructions).

The 8051 has 128 bytes of on-chip RAM (256 bytes in the 8052) plus a number of Special Function Registers (SFRs). The lower 128 bytes of RAM can be accessed either by direct addressing (MOV data addr) or by indirect addressing (MOV @Ri). Figure 3 shows the 8051 and the 8052 Data Memory organization.

8051, 8052 AND 80C51 HARDWARE DESCRIPTION

INTRODUCTION

This chapter presents a comprehensive description of the on-chip hardware features of the MCS®-51 microcontrollers. Included in this description are

- The port drivers and how they function both as ports and, for Ports 0 and 2, in bus operations
- The Timer/Counters
- The Serial Interface
- The Interrupt System
- Reset
- The Reduced Power Modes in the CHMOS devices
- The EPROM versions of the 8051AH, 8052AH, and 80C51BH

The devices under consideration are listed in Table 1. As it becomes unwieldy to be constantly referring to each of these devices by their individual names, we will adopt a convention of referring to them generically as 8051s and 8052s, unless a specific member of the group is being referred to, in which case it will be specifically named. The "8051s" include the 8051, 8051AH, and 80C51BH, and their ROMless and EPROM versions. The "8052s" are the 8052AH, 8032AH, and 8752BH.

Figure 1 shows a functional block diagram of the 8051s and 8052s.

Table 1. The MCS-51 Family of Microcontrollers

Device Name	ROMless Version	EPROM Version	ROM Bytes	RAM Bytes	16-bit Timers	Ckt Type
8051	8031	(8751)	4K	128	2	HMOS
8051AH	8031AH	8751H	4K	128	2	HMOS
8052AH	8032AH	8752BH	8K	256	3	HMOS
80C51BH	80C31BH	87C51	4K	128	2	CHMOS

Special Function Registers

A map of the on-chip memory area called SFR (Special Function Register) space is shown in Figure 2. SFRs marked by parentheses are resident in the 8052s but not in the 8051s.

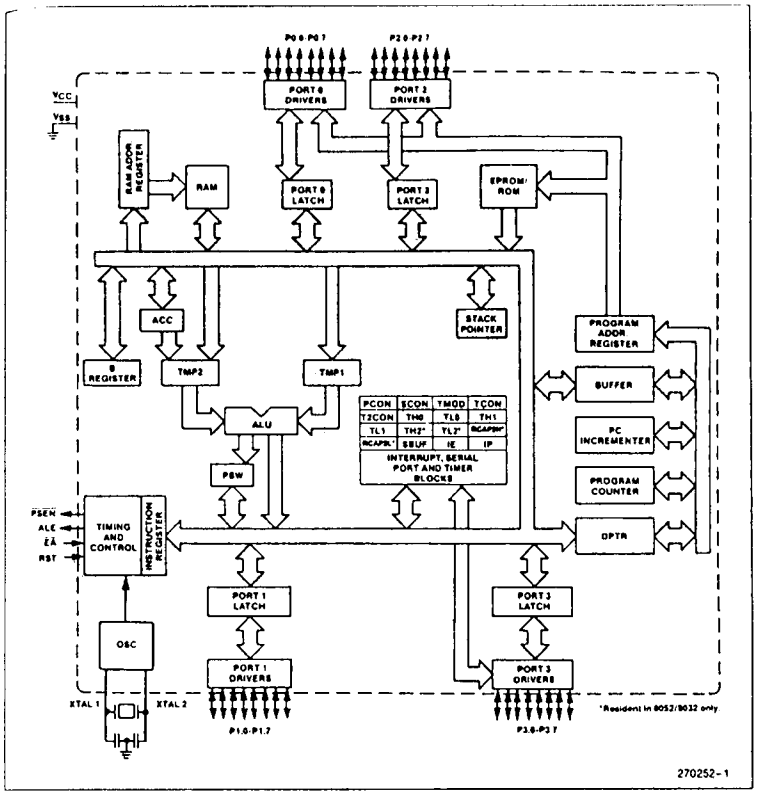


Figure 1. MCS-51 Architectural Block Diagram

	8 Bytes								
F8									FF
F0	B								F7
E8									EF
E0	ACC								E7
D8									DF
D0	PSW								D7
C8	(T2CON)		(RCAP2L)	(RCAP2H)	(TL2)	(TH2)			CF
C0									C7
B8	IP								BF
B0	P3								B7
A8	IE								AF
A0	P2								A7
98	SCON	SBUF							9F
90	P1								97
88	TCON	TMOD	TL0	TL1	TH0	TH1			8F
80	P0	SP	DPL	DPH				PCON	87

Figure 2. SFR Map. (. . .) Indicates Resident in 8052s, not in 8051s

Note that not all of the addresses are occupied. Unoccupied addresses are not implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have no effect.

User software should not write 1s to these unimplemented locations, since they may be used in future MCS-51 products to invoke new features. In that case the reset or inactive values of the new bits will always be 0, and their active values will be 1.

The functions of the SFRs are outlined below.

ACCUMULATOR

ACC is the Accumulator register. The mnemonics for Accumulator-Specific instructions, however, refer to the Accumulator simply as A.

B REGISTER

The B register is used during multiply and divide operations. For other instructions it can be treated as another scratch pad register.

PROGRAM STATUS WORD

The PSW register contains program status information as detailed in Figure 3.

STACK POINTER

The Stack Pointer Register is 8 bits wide. It is incremented before data is stored during PUSH and CALL executions. While the stack may reside anywhere in on-chip RAM, the Stack Pointer is initialized to 07H after a reset. This causes the stack to begin at location 08H.

DATA POINTER

The Data Pointer (DPTR) consists of a high byte (DPH) and a low byte (DPL). Its intended function is to hold a 16-bit address. It may be manipulated as a 16-bit register or as two independent 8-bit registers.

PORTS 0 TO 3

P0, P1, P2 and P3 are the SFR latches of Ports 0, 1, 2 and 3, respectively.

SERIAL DATA BUFFER

The Serial Data Buffer is actually two separate registers, a transmit buffer and a receive buffer register. When data is moved to SBUF, it goes to the transmit buffer where it is held for serial transmission. (Moving a byte to SBUF is what initiates the transmission.) When data is moved from SBUF, it comes from the receive buffer.

TIMER REGISTERS

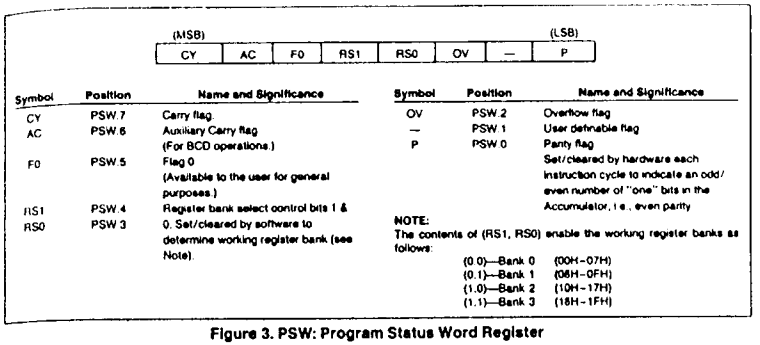
Register pairs (TH0, TL0), (TH1, TL1), and (TH2, TL2) are the 16-bit Counting registers for Timer/Counters 0, 1, and 2, respectively.

CAPTURE REGISTERS

The register pair (RCAP2H, RCAP2L) are the Capture registers for the Timer 2 "Capture Mode." In this mode, in response to a transition at the 8052's T2EX pin, TH2 and TL2 are copied into RCAP2H and RCAP2L. Timer 2 also has a 16-bit auto-reload mode, and RCAP2H and RCAP2L hold the reload value for this mode. More about Timer 2's features in a later section.

CONTROL REGISTERS

Special Function Registers IP, IE, TMOD, TCON, T2CON, SCON, and PCON contain control and status bits for the interrupt system, the Timer/Counters, and the serial port. They are described in later sections



Symbol	Position	Name and Significance
CY	PSW.7	Carry flag.
AC	PSW.6	Auxiliary Carry flag (For BCD operations.)
F0	PSW.5	Flag 0 (Available to the user for general purposes.)
RS1	PSW.4	Register bank select control bits 1 &
RS0	PSW.3	0. Set/cleared by software to determine working register bank (see Note)

Symbol	Position	Name and Significance
OV	PSW.2	Overflow flag
—	PSW.1	User definable flag
P	PSW.0	Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even number of "one" bits in the Accumulator, i.e., even parity

NOTE:
The contents of (RS1, RS0) enable the working register banks as follows:

(0,0)—Bank 0 (00H–07H)
(0,1)—Bank 1 (08H–0FH)
(1,0)—Bank 2 (10H–17H)
(1,1)—Bank 3 (18H–1FH)

Figure 3. PSW: Program Status Word Register

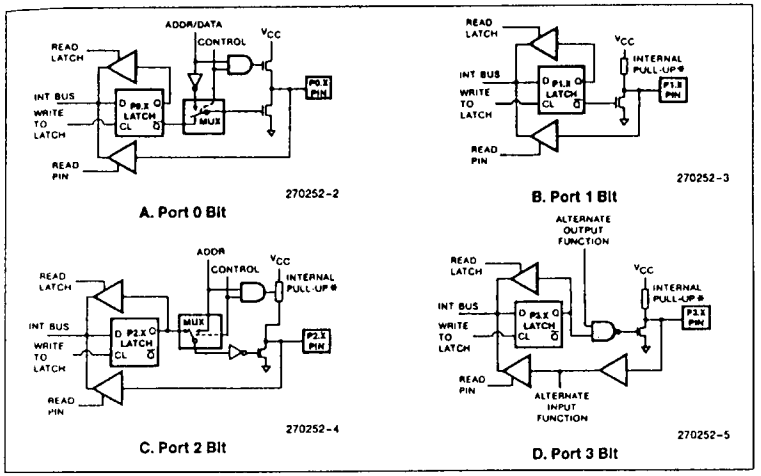


Figure 4. 8051 Port Bit Latches and I/O Buffers
*See Figure 5 for details of the internal pullup.

PORT STRUCTURES AND OPERATION

All four ports in the 8051 are bidirectional. Each consists of a latch (Special Function Registers P0 through P3), an output driver, and an input buffer.

The output drivers of Ports 0 and 2, and the input buffers of Port 0, are used in accesses to external memory. In this application, Port 0 outputs the low byte of the external memory address, time-multiplexed with the byte being written or read. Port 2 outputs the high byte of the external memory address when the address is 16 bits wide. Otherwise the Port 2 pins continue to emit the P2 SFR content.

All the Port 3 pins, and (in the 8052) two Port 1 pins are multifunctional. They are not only port pins, but also serve the functions of various special features as listed on the following page.

Port Pin	Alternate Function
*P1.0	T2 (Timer/Counter 2 external input)
*P1.1	T2EX (Timer/Counter 2 Capture/Reload trigger)
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt)
P3.3	$\overline{INT1}$ (external interrupt)
P3.4	T0 (Timer/Counter 0 external input)
P3.5	T1 (Timer/Counter 1 external input)
P3.6	$\overline{WR}$ (external Data Memory write strobe)
P3.7	$\overline{RD}$ (external Data Memory read strobe)

*P1.0 and P1.1 serve these alternate functions only on the 8052.

The alternate functions can only be activated if the corresponding bit latch in the port SFR contains a 1. Otherwise the port pin is stuck at 0.

I/O Configurations

Figure 4 shows a functional diagram of a typical bit latch and I/O buffer in each of the four ports. The bit latch (one bit in the port's SFR) is represented as a Type D flip-flop, which will clock in a value from the internal bus in response to a "write to latch" signal from the CPU. The Q output of the flip-flop is placed on the internal bus in response to a "read latch" signal from the CPU. The level of the port pin itself is placed on the internal bus in response to a "read pin" signal from the CPU. Some instructions that read a port activate the "read latch" signal, and others activate the "read pin" signal. More about that later.

As shown in Figure 4, the output drivers of Ports 0 and 2 are switchable to an internal ADDR and ADDR/DATA bus by an internal CONTROL signal for use in external memory accesses. During external memory accesses, the P2 SFR remains unchanged, but the P0 SFR gets 1s written to it.

Also shown in Figure 4, is that if a P3 bit latch contains a 1, then the output level is controlled by the signal labeled "alternate output function." The actual P3.X pin level is always available to the pin's alternate input function, if any.

Ports 1, 2, and 3 have internal pullups. Port 0 has open drain outputs. Each I/O line can be independently used as an input or an output. (Ports 0 and 2 may not be used as general purpose I/O when being used as the ADDR/DATA BUS). To be used as an input, the port bit latch must contain a 1, which turns off the output driver FET. Then, for Ports 1, 2, and 3, the pin is pulled high by the internal pullup, but can be pulled low by an external source.

Port 0 differs in not having internal pullups. The pullup FET in the P0 output driver (see Figure 4) is used only when the Port is emitting 1s during external memory accesses. Otherwise the pullup FET is off. Consequently P0 lines that are being used as output port lines are open drain. Writing a 1 to the bit latch leaves both output FETs off, so the pin floats. In that condition it can be used a high-impedance input.

Because Ports 1, 2, and 3 have fixed internal pullups they are sometimes called "quasi-bidirectional" ports. When configured as inputs they pull high and will source current (IIL, in the data sheets) when externally pulled low. Port 0, on the other hand, is considered "true" bidirectional, because when configured as an input it floats.

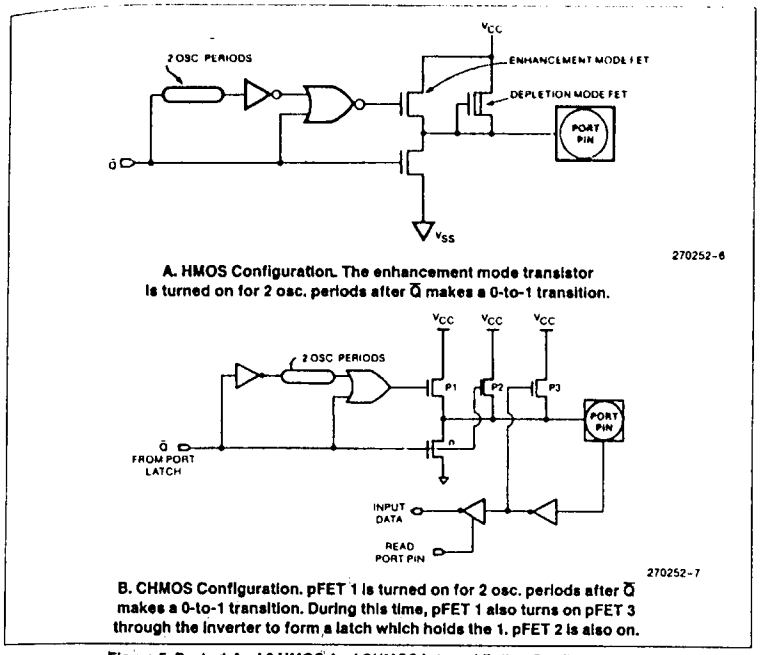
All the port latches in the 8051 have 1s written to them by the reset function. If a 0 is subsequently written to a port latch, it can be reconfigured as an input by writing a 1 to it.

Writing to a Port

In the execution of an instruction that changes the value in a port latch, the new value arrives at the latch during S6P2 of the final cycle of the instruction. However, port latches are in fact sampled by their output buffers only during Phase 1 of any clock period. (During Phase 2 the output buffer holds the value it saw during the previous Phase 1). Consequently, the new value in the port latch won't actually appear at the output pin until the next Phase 1, which will be at S1P1 of the next machine cycle. See Figure 39 in the Internal Timing section.

If the change requires a 0-to-1 transition in Port 1, 2, or 3, an additional pullup is turned on during S1P1 and S1P2 of the cycle in which the transition occurs. This is done to increase the transition speed. The extra pullup can source about 100 times the current that the normal pullup can. It should be noted that the internal pullups are field-effect transistors, not linear resistors. The pullup arrangements are shown in Figure 5.

In HMOS versions of the 8051, the fixed part of the pullup is a depletion-mode transistor with the gate wired to the source. This transistor will allow the pin to source about 0.25 mA when shorted to ground. In parallel with the fixed pullup is an enhancement-mode transistor, which is activated during S1 whenever the port bit does a 0-to-1 transition. During this interval if the port pin is shorted to ground, this extra transistor will allow the pin to source an additional 30 mA.



A. HMOS Configuration. The enhancement mode transistor is turned on for 2 osc. periods after $\overline{Q}$ makes a 0-to-1 transition.

B. CHMOS Configuration. pFET 1 is turned on for 2 osc. periods after $\overline{Q}$ makes a 0-to-1 transition. During this time, pFET 1 also turns on pFET 3 through the inverter to form a latch which holds the 1. pFET 2 is also on.

Figure 5. Ports 1 And 3 HMOS And CHMOS Internal Pullup Configurations. Port 2 Is Similar Except That It Holds The Strong Pullup On While Emitting 1s That Are Address Bits. (See Text, "Accessing External Memory".)

In the CHMOS versions, the pullup consists of three pFETs. It should be noted that an n-channel FET (nFET) is turned on when a logical 1 is applied to its gate, and is turned off when a logical 0 is applied to its gate. A p-channel FET (pFET) is the opposite: it is on when its gate sees a 0, and off when its gate sees a 1.

pFET1 in Figure 5 is the transistor that is turned on for 2 oscillator periods after a 0-to-1 transition in the port latch. While it's on, it turns on pFET3 (a weak pullup), through the inverter. This inverter and pFET form a latch which hold the 1.

Note that if the pin is emitting a 1, a negative glitch on the pin from some external source can turn off pFET3, causing the pin to go into a float state. pFET2 is a very weak pullup which is on whenever the nFET is off, in traditional CMOS style. It's only about $1/_{10}$ the strength of pFET3. Its function is to restore a 1 to the pin in the event the pin *had* a 1 and lost it to a glitch.

Port Loading and Interfacing

The output buffers of Ports 1, 2, and 3 can each drive 4 LS TTL inputs. These ports on HMOS versions can be driven in a normal manner by any TTL or NMOS circuit. Both HMOS and CHMOS pins can be driven by open-collector and open-drain outputs, but note that 0-to-1 transitions will not be fast. In the HMOS device, if the pin is driven by an open-collector output, a 0-to-1 transition will have to be driven by the relatively weak depletion mode FET in Figure 5(A). In the CHMOS device, an input 0 turns off pullup pFET3, leaving only the very weak pullup pFET2 to drive the transition.

In external bus mode, Port 0 output buffers can each drive 8 LS TTL inputs. As port pins, they require external pullups to drive any inputs.

Read-Modify-Write Feature

Some instructions that read a port read the latch and others read the pin. Which ones do which? The instructions that read the latch rather than the pin are the ones that read a value, possibly change it, and then rewrite it to the latch. These are called "read-modify-write" instructions. The instructions listed below are read-modify-write instructions. When the destination operand is a port, or a port bit, these instructions read the latch rather than the pin:

ANL	(logical AND, e.g., ANL P1, A)
ORL	(logical OR, e.g., ORL P2, A)
XRL	(logical EX-OR, e.g., XRL P3, A)
JBC	(jump if bit = 1 and clear bit, e.g., JBC P1.1, LABEL)
CPL	(complement bit, e.g., CPL P3.0)
INC	(increment, e.g., INC P2)
DEC	(decrement, e.g., DEC P2)
DJNZ	(decrement and jump if not zero, e.g., DJNZ P3, LABEL)
MOV, PX.Y, C	(move carry bit to bit Y of Port X)
CLR PX.Y	(clear bit Y of Port X)
SETB PX.Y	(set bit Y of Port X)

It is not obvious that the last three instructions in this list are read-modify-write instructions, but they are. They read the port byte, all 8 bits, modify the addressed bit, then write the new byte back to the latch.

The reason that read-modify-write instructions are directed to the latch rather than the pin is to avoid a possible misinterpretation of the voltage level at the pin. For example, a port bit might be used to drive the base of a transistor. When a 1 is written to the bit, the transistor is turned on. If the CPU then reads the same port bit at the pin rather than the latch, it will read the base voltage of the transistor and interpret it as a 0. Reading the latch rather than the pin will return the correct value of 1.

ACCESSING EXTERNAL MEMORY

Accesses to external memory are of two types: accesses to external Program Memory and accesses to external Data Memory. Accesses to external Program Memory use signal $\overline{PSEN}$ (program store enable) as the read strobe. Accesses to external Data Memory use $\overline{RD}$ or $\overline{WR}$ (alternate functions of P3.7 and P3.6) to strobe the memory. Refer to Figures 36 through 38 in the Internal Timing section.

Fetches from external Program Memory always use a 16-bit address. Accesses to external Data Memory can use either a 16-bit address (MOVX @DPTR) or an 8-bit address (MOVX @Ri).

Whenever a 16-bit address is used, the high byte of the address comes out on Port 2, where it is held for the duration of the read or write cycle. Note that the Port 2 drivers use the strong pullups during the entire time that they are emitting address bits that are 1s. This is during the execution of a MOVX @DPTR instruction. During this time the Port 2 latch (the Special Function Register) does not have to contain 1s, and the contents of the Port 2 SFR are not modified. If the external memory cycle is not immediately followed by another external memory cycle, the undisturbed contents of the Port 2 SFR will reappear in the next cycle.

If an 8-bit address is being used (MOVX @Ri), the contents of the Port 2 SFR remain at the Port 2 pins throughout the external memory cycle. This will facilitate paging.

In any case, the low byte of the address is time-multiplexed with the data byte on Port 0. The ADDR/DATA signal drives both FETs in the Port 0 output buffers. Thus, in this application the Port 0 pins are not open-drain outputs, and do not require external pullups. Signal ALE (Address Latch Enable) should be used to capture the address byte into an external latch. The address byte is valid at the negative transition of ALE. Then, in a write cycle, the data byte to be written appears on Port 0 just before $\overline{WR}$ is activated, and remains there until after $\overline{WR}$ is deactivated. In a read cycle, the incoming byte is accepted at Port 0 just before the read strobe is deactivated.

During any access to external memory, the CPU writes 0FFH to the Port 0 latch (the Special Function Register), thus obliterating whatever information the Port 0 SFR may have been holding. If the user writes to Port 0 during an external memory fetch, the incoming code byte is corrupted. Therefore, do not write to Port 0 if external program memory is used.

External Program Memory is accessed under two conditions:

1) Whenever signal $\overline{EA}$ is active; or
2) Whenever the program counter (PC) contains a number that is larger than 0FFFH (1FFFH for the 8052).

This requires that the ROMless versions have $\overline{EA}$ wired low to enable the lower 4K (8K for the 8032) program bytes to be fetched from external memory.

When the CPU is executing out of external Program Memory, all 8 bits of Port 2 are dedicated to an output function and may not be used for general purpose I/O. During external program fetches they output the high byte of the PC. During this time the Port 2 drivers use the strong pullups to emit PC bits that are 1s.

TIMER/COUNTERS

The 8051 has two 16-bit Timer/Counter registers: Timer 0 and Timer 1. The 8052 has these two plus one more: Timer 2. All three can be configured to operate either as timers or event counters.

In the "Timer" function, the register is incremented every machine cycle. Thus, one can think of it as counting machine cycles. Since a machine cycle consists of 12 oscillator periods, the count rate is $1/12$ of the oscillator frequency.

In the "Counter" function, the register is incremented in response to a 1-to-0 transition at its corresponding external input pin, T0, T1 or (in the 8052) T2. In this function, the external input is sampled during S5P2 of every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register during S3P1 of the cycle following the one in which the transition was detected. Since it takes 2 machine cycles (24 oscillator periods) to recognize a 1-to-0 transition, the maximum count rate is $1/24$ of the oscillator frequency. There are no restrictions on the duty cycle of the external input signal, but to ensure that a given level is sampled at least once before it changes, it should be held for at least one full machine cycle.

In addition to the "Timer" or "Counter" selection, Timer 0 and Timer 1 have four operating modes from which to select. Timer 2, in the 8052, has three modes of operation: "Capture," "Auto-Reload" and "baud rate generator."

Timer 0 and Timer 1

These Timer/Counters are present in both the 8051 and the 8052. The "Timer" or "Counter" function is selected by control bits $C/\overline{T}$ in the Special Function Register TMOD (Figure 6). These two Timer/Counters have four operating modes, which are selected by bit-pairs (M1, M0) in TMOD. Modes 0, 1, and 2 are the same for both Timer/Counters. Mode 3 is different. The four operating modes are described in the following text.

MODE 0

Either Timer in Mode 0 is an 8-bit Counter with a divide-by-32 prescaler. This 13-bit timer is MCS-48 compatible. Figure 7 shows the Mode 0 operation as it applies to Timer 1.

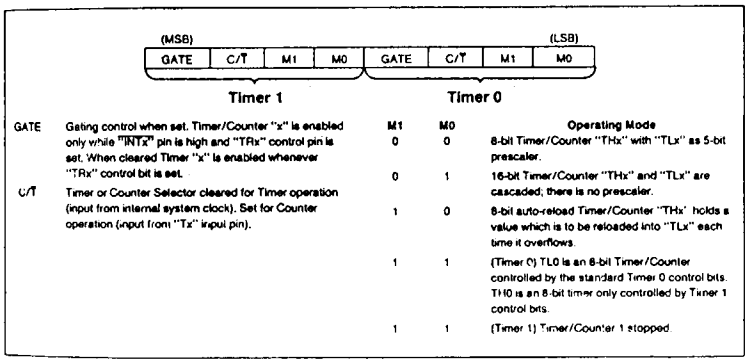
In this mode, the Timer register is configured as a 13-Bit register. As the count rolls over from all 1s to all 0s, it sets the Timer interrupt flag TF1. The counted input is enabled to the Timer when TR1 = 1 and either GATE = 0 or $\overline{INT1}$ = 1. (Setting GATE = 1 allows the Timer to be controlled by external input $\overline{INT1}$, to facilitate pulse width measurements.) TR1 is a control bit in the Special Function Register TCON (Figure 8). GATE is in TMOD.

The 13-Bit register consists of all 8 bits of TH1 and the lower 5 bits of TL1. The upper 3 bits of TL1 are indeterminate and should be ignored. Setting the run flag (TR1) does not clear the registers.

Mode 0 operation is the same for Timer 0 as for Timer 1. Substitute TR0, TF0 and $\overline{INT0}$ for the corresponding Timer 1 signals in Figure 7. There are two different GATE bits, one for Timer 1 (TMOD.7) and one for Timer 0 (TMOD.3).

MODE 1

Mode 1 is the same as Mode 0, except that the Timer register is being run with all 16 bits.



GATE — Gating control when set. Timer/Counter "x" is enabled only while "INTx" pin is high and "TRx" control pin is set. When cleared Timer "x" is enabled whenever "TRx" control bit is set.

C/T — Timer or Counter Selector cleared for Timer operation (input from internal system clock). Set for Counter operation (input from "Tx" input pin).

M1	M0	Operating Mode
0	0	8-bit Timer/Counter "THx" with "TLx" as 5-bit prescaler.
0	1	16-bit Timer/Counter "THx" and "TLx" are cascaded; there is no prescaler.
1	0	8-bit auto-reload Timer/Counter "THx" holds a value which is to be reloaded into "TLx" each time it overflows.
1	1	(Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. TH0 is an 8-bit timer only controlled by Timer 1 control bits.
1	1	(Timer 1) Timer/Counter 1 stopped.

Figure 6. TMOD: Timer/Counter Mode Control Register

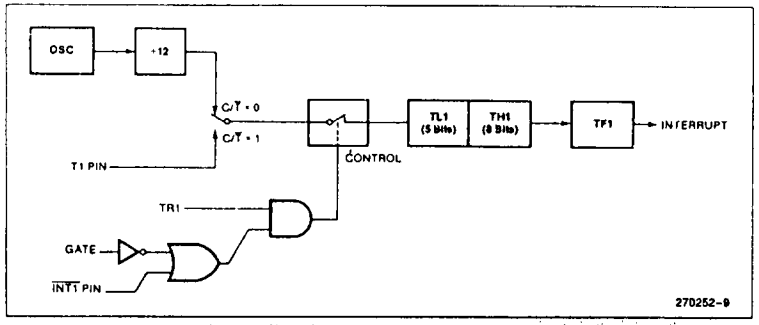
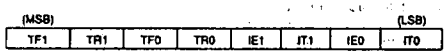


Figure 7. Timer/Counter 1 Mode 0: 13-Bit Counter



Symbol	Position	Name and Significance	Symbol	Position	Name and Significance
TF1	TCON.7	Timer 1 overflow Flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when processor vectors to interrupt routine.	IE1	TCON.3	Interrupt 1 Edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.
TR1	TCON.6	Timer 1 Run control bit. Set/cleared by software to turn Timer/Counter on/off.	IT1	TCON.2	Interrupt 1 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.
TF0	TCON.5	Timer 0 overflow Flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when processor vectors to interrupt routine.	IE0	TCON.1	Interrupt 0 Edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.
TR0	TCON.4	Timer 0 Run control bit. Set/cleared by software to turn Timer/Counter on/off.	IT0	TCON.0	Interrupt 0 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.

Figure 8.TCON: Timer/Counter Control Register

MODE 2

Mode 2 configures the Timer register as an 8-bit Counter (TL1) with automatic reload, as shown in Figure 9. Overflow from TL1 not only sets TF1, but also reloads TL1 with the contents of TH1, which is preset by software. The reload leaves TH1 unchanged.

Mode 2 operation is the same for Timer/Counter 0.

MODE 3

Timer 1 in Mode 3 simply holds its count. The effect is the same as setting TR1 = 0.

Timer 0 in Mode 3 establishes TL0 and TH0 as two separate counters. The logic for Mode 3 on Timer 0 is shown in Figure 10. TL0 uses the Timer 0 control bits: C/T, GATE, TR0, INT0, and TF0. TH0 is locked into a timer function (counting machine cycles) and takes over the use of TR1 and TF1 from Timer 1. Thus, TH0 now controls the "Timer 1" interrupt.

Mode 3 is provided for applications requiring an extra 8-bit timer or counter. With Timer 0 in Mode 3, an 8051 can look like it has three Timer/Counters, and an 8052, like it has four. When Timer 0 is in Mode 3, Timer 1 can be turned on and off by switching it out of and into its own Mode 3, or can still be used by the serial port as a baud rate generator, or in fact, in any application not requiring an interrupt.

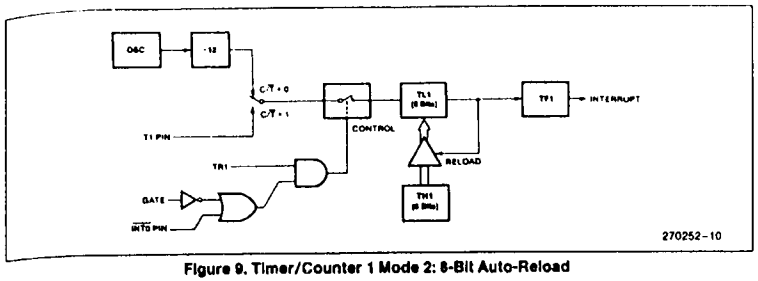


Figure 9. Timer/Counter 1 Mode 2: 8-Bit Auto-Reload

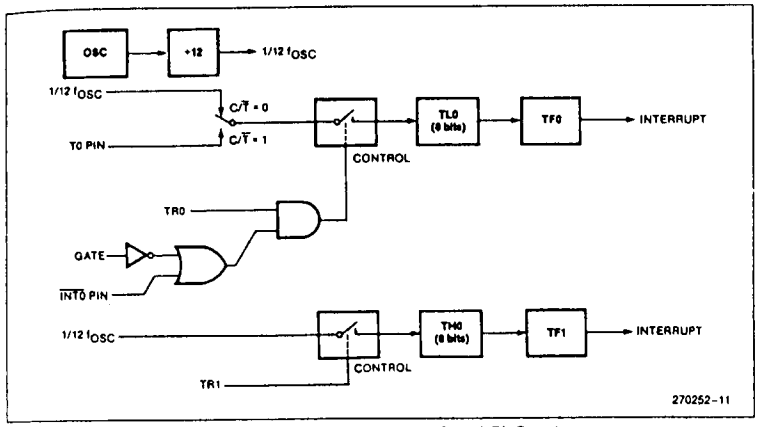


Figure 10. Timer/Counter 0 Mode 3: Two 8-Bit Counters

Timer 2

Timer 2 is a 16-bit Timer/Counter which is present only in the 8052. Like Timers 0 and 1, it can operate either as a timer or as an event counter. This is selected by bit C/T2 in the Special Function Register T2CON (Figure 11). It has three operating modes: "capture," "auto-load" and "baud rate generator," which are selected by bits in T2CON as shown in Table 2.

Table 2. Timer 2 Operating Modes

RCLK + TCLK	CP/RL2	TR2	Mode
0	0	1	16-bit Auto-Reload
0	1	1	16-bit Capture
1	X	1	Baud Rate Generator
X	X	0	(off)

7

(MSB)							(LSB)
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2	CP/RL2

Symbol	Position	Name and Significance
TF2	T2CON.7	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.
EXF2	T2CON.6	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software.
RCLK	T2CON.5	Receive clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in Modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.
TCLK	T2CON.4	Transmit clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.
EXEN2	T2CON.3	Timer 2 external enable flag. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.
TR2	T2CON.2	Start/stop control for Timer 2. A logic 1 starts the timer.
C/T2	T2CON.1	Timer or counter select. (Timer 2) 0 = Internal timer (OSC/12) 1 = External event counter (falling edge triggered).
CP/RL2	T2CON.0	Capture/Reload flag. When set, captures will occur on negative transitions at T2EX if EXEN2 = 1. When cleared, auto-reloads will occur either with Timer 2 overflows or negative transitions at T2EX when EXEN2 = 1. When either RCLK = 1 or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

Figure 11. T2CON: Timer/Counter 2 Control Register

In the Capture Mode there are two options which are selected by bit EXEN2 in T2CON. If EXEN2 = 0, then Timer 2 is a 16-bit timer or counter which upon overflowing sets bit TF2, the Timer 2 overflow bit, which can be used to generate an interrupt. If EXEN2 = 1, then Timer 2 still does the above, but with the added feature that a 1-to-0 transition at external input T2EX causes the current value in the Timer 2 registers, TL2 and TH2, to be captured into registers RCAP2L and RCAP2H, respectively. (RCAP2L and RCAP2H are new Special Function Registers in the 8052.) In addition, the transition at T2EX causes bit EXF2 in T2CON to be set, and EXF2, like TF2, can generate an interrupt.

The Capture Mode is illustrated in Figure 12.

In the auto-reload mode there are again two options, which are selected by bit EXEN2 in T2CON. If EXEN2 = 0, then when Timer 2 rolls over it not only sets TF2 but also causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2L and RCAP2H, which are preset by software. If EXEN2 = 1, then Timer 2 still does the above, but with the added feature that a 1-to-0 transition at external input T2EX will also trigger the 16-bit reload and set EXF2.

The auto-reload mode is illustrated in Figure 13.

The baud rate generator mode is selected by RCLK = 1 and/or TCLK = 1. It will be described in conjunction with the serial port.

SERIAL INTERFACE

The serial port is full duplex, meaning it can transmit and receive simultaneously. It is also receive-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the receive register. (However, if the first byte still hasn't been read by the time reception of the second byte is complete, one of the bytes will be lost). The serial port receive and transmit registers are both accessed at Special Function Register SBUF. Writing to SBUF loads the transmit register, and reading SBUF accesses a physically separate receive register.

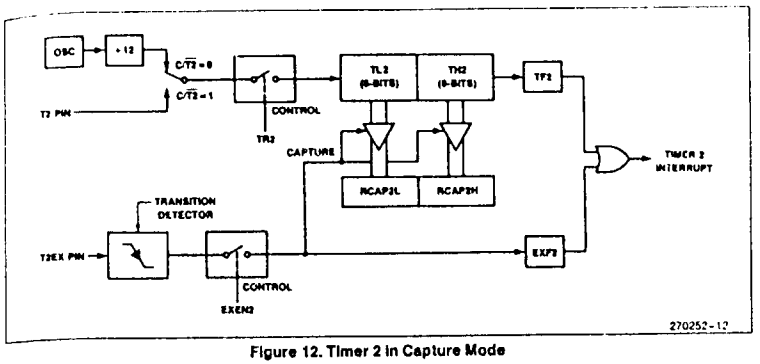


Figure 12. Timer 2 in Capture Mode

The serial port can operate in 4 modes:

Mode 0: Serial data enters and exits through RXD. TXD outputs the shift clock. 8 bits are transmitted/received: 8 data bits (LSB first). The baud rate is fixed at 1/12 the oscillator frequency.

Mode 1: 10 bits are transmitted (through TXD) or received (through RXD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB8 in Special Function Register SCON. The baud rate is variable.

Mode 2: 11 bits are transmitted (through TXD) or received (through RXD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On Transmit, the 9th data bit (TB8 in SCON) can be assigned the value of 0 or 1. Or, for example, the parity bit (P, in the PSW) could be moved into TB8. On receive, the 9th data bit goes into RB8 in Special Functon Register SCON, while the stop bit is ignored. The baud rate is programmable to either $1/_{32}$ or $1/_{64}$ the oscillator frequency.

Mode 3: 11 bits are transmitted (through TXD) or received (through RXD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit and a stop bit (1). In fact, Mode 3 is the same as Mode 2 in all respects except the baud rate. The baud rate in Mode 3 is variable.

In all four modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in Mode 0 by the condition RI = 0 and REN = 1. Reception is initiated in the other modes by the incoming start bit if REN = 1.

Multiprocessor Communications

Modes 2 and 3 have a special provision for multiprocessor communications. In these modes, 9 data bits are received. The 9th one goes into RB8. Then comes a stop bit. The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1. This feature is enabled by setting bit SM2 in SCON. A way to use this feature in multiprocessor systems is as follows.

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte which identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SM2 = 1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SM2 bit and prepare to receive the data bytes that will be coming. The slaves that weren't being addressed leave their SM2s set and go on about their business, ignoring the coming data bytes.

SM2 has no effect in Mode 0, and in Mode 1 can be used to check the validity of the stop bit. In a Mode 1 reception, if SM2 = 1, the receive interrupt will not be activated unless a valid stop bit is received.

Serial Port Control Register

The serial port control and status register is the Special Function Register SCON, shown in Figure 14. This register contains not only the mode selection bits, but also the 9th data bit for transmit and receive (TB8 and RB8), and the serial port interrupt bits (TI and RI).

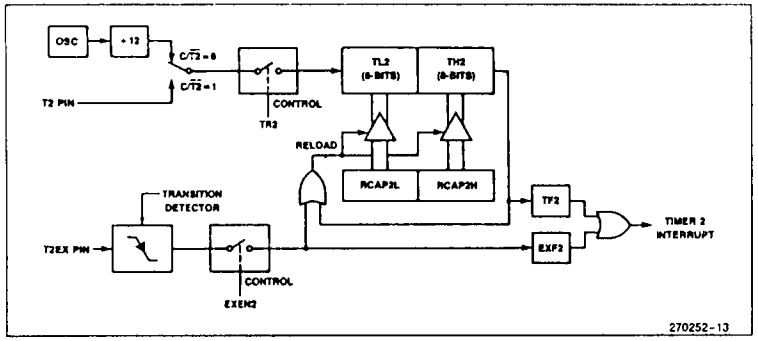
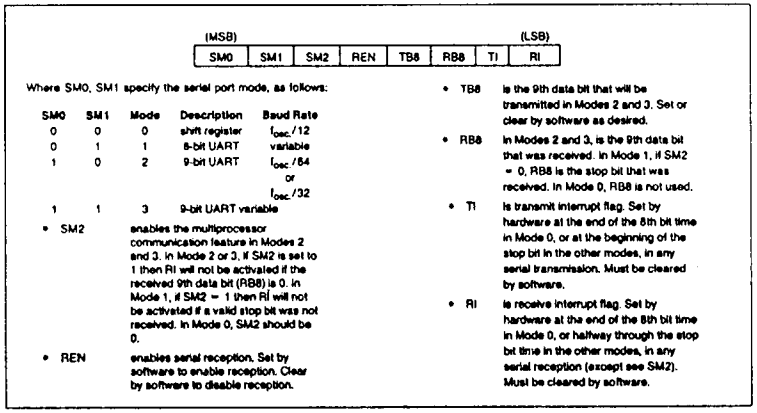


Figure 13. Timer 2 in Auto-Reload Mode



Where SM0, SM1 specify the serial port mode, as follows:

SM0	SM1	Mode	Description	Baud Rate
0	0	0	shift register	$f_{osc.}/12$
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	$f_{osc.}/64$ or $f_{osc.}/32$
1	1	3	9-bit UART	variable

- SM2 enables the multiprocessor communication feature in Modes 2 and 3. In Mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In Mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In Mode 0, SM2 should be 0.
- REN enables serial reception. Set by software to enable reception. Clear by software to disable reception.
- TB8 is the 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software as desired.
- RB8 In Modes 2 and 3, is the 9th data bit that was received. In Mode 1, if SM2 = 0, RB8 is the stop bit that was received. In Mode 0, RB8 is not used.
- TI is transmit interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. Must be cleared by software.
- RI is receive interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM2). Must be cleared by software.

Figure 14. SCON: Serial Port Control Register

Baud Rates

The baud rate in Mode 0 is fixed:

$$\text{Mode 0 Baud Rate} = \frac{\text{Oscillator Frequency}}{12}$$

The baud rate in Mode 2 depends on the value of bit SMOD in Special Function Register PCON. If SMOD = 0 (which is the value on reset), the baud rate is $1/64$ the oscillator frequency. If SMOD = 1, the baud rate is $1/32$ the oscillator frequency.

$$\text{Mode 2 Baud Rate} = \frac{2^{SMOD}}{64} \times (\text{Oscillator Frequency})$$

In the 8051, the baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate. In the 8052, these baud rates can be determined by Timer 1, or by Timer 2, or by both (one for transmit and the other for receive).

Using Timer 1 to Generate Baud Rates

When Timer 1 is used as the baud rate generator, the baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate and the value of SMOD as follows:

$$\text{Modes 1, 3 Baud Rate} = \frac{2^{SMOD}}{32} \times (\text{Timer 1 Overflow Rate})$$

The Timer 1 interrupt should be disabled in this application. The Timer itself can be configured for either "timer" or "counter" operation, and in any of its 3 running modes. In the most typical applications, it is configured for "timer" operation, in the auto-reload mode (high nibble of TMOD = 0010B). In that case, the baud rate is given by the formula

$$\text{Modes 1, 3 Baud Rate} = \frac{2^{SMOD}}{32} \times \frac{\text{Oscillator Frequency}}{12 \times [256 - (TH1)]}$$

One can achieve very low baud rates with Timer 1 by leaving the Timer 1 interrupt enabled, and configuring the Timer to run as a 16-bit timer (high nibble of TMOD = 0001B), and using the Timer 1 interrupt to do a 16-bit software reload.

Figure 15 lists various commonly used baud rates and how they can be obtained from Timer 1.

Baud Rate	f_{OSC}	SMOD	Timer 1		
			C/T	Mode	Reload Value
Mode 0 Max: 1 MHZ	12 MHZ	X	X	X	X
Mode 2 Max: 375K	12 MHZ	1	X	X	X
Modes 1, 3: 62.5K	12 MHZ	1	0	2	FFH
19.2K	11.059 MHZ	1	0	2	FDH
9.6K	11.059 MHZ	0	0	2	FDH
4.8K	11.059 MHZ	0	0	2	FAH
2.4K	11.059 MHZ	0	0	2	F4H
1.2K	11.059 MHZ	0	0	2	E8H
137.5	11.986 MHZ	0	0	2	1DH
110	6 MHZ	0	0	2	72H
110	12 MHZ	0	0	1	FEEBH

Figure 15. Timer 1 Generated Commonly Used Baud Rates

Using Timer 2 to Generate Baud Rates

In the 8052, Timer 2 is selected as the baud rate generator by setting TCLK and/or RCLK in T2CON (Figure 11). Note then the baud rates for transmit and receive can be simultaneously different. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode, as shown in Figure 16.

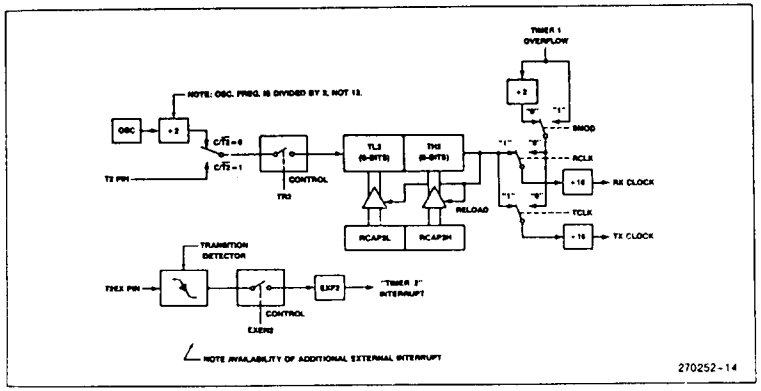


Figure 16. Timer 2 in Baud Rate Generator Mode

The baud rate generator mode is similar to the auto-reload mode, in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software.

Now, the baud rates in Modes 1 and 3 are determined by Timer 2's overflow rate as follows:

$$\text{Modes 1, 3 Baud Rate} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

The Timer can be configured for either "timer" or "counter" operation. In the most typical applications, it is configured for "timer" operation (C/T2 = 0). "Timer" operation is a little different for Timer 2 when it's being used as a baud rate generator. Normally, as a timer it would increment every machine cycle (thus at $1/12$ the oscillator frequency). As a baud rate generator, however, it increments every state time (thus at $1/2$ the oscillator frequency). In that case the baud rate is given by the formula

$$\text{Modes 1, 3 Baud Rate} = \frac{\text{Oscillator Frequency}}{32 \times [65536 - (\text{RCAP2H, RCAP2L})]}$$

where (RCAP2H, RCAP2L) is the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

Timer 2 as a baud rate generator is shown in Figure 16. This Figure is valid only if RCLK + TCLK = 1 in T2CON. Note that a rollover in TH2 does not set TF2, and will not generate an interrupt. Therefore, the Timer 2 interrupt does not have to be disabled when Timer 2 is in the baud rate generator mode. Note too, that if EXEN2 is set, a 1-to-0 transition in T2EX will set EXF2 but will not cause a reload from (RCAP2H, RCAP2L) to (TH2, TL2). Thus when Timer 2 is in use as a baud rate generator, T2EX can be used as an extra external interrupt, if desired.

It should be noted that when Timer 2 is running (TR2 = 1) in "timer" function in the baud rate generator mode, one should not try to read or write TH2 or TL2. Under these conditions the Timer is being incremented every state time, and the results of a read or write may not be accurate. The RCAP registers may be read, but shouldn't be written to, because a write might overlap a reload and cause write and/or reload errors. Turn the Timer off (clear TR2) before accessing the Timer 2 or RCAP registers, in this case.

More About Mode 0

Serial data enters and exits through RXD. TXD outputs the shift clock. 8 bits are transmitted/received: 8 data bits (LSB first). The baud rate is fixed at $1/12$ the oscillator frequency.

Figure 17 shows a simplified functional diagram of the serial port in Mode 0, and associated timing.

Transmission is initiated by any instruction that uses SBUF as a destination register. The "write to SBUF" signal at S6P2 also loads a 1 into the 9th position of the transmit shift register and tells the TX Control block to commence a transmission. The internal timing is such that one full machine cycle will elapse between "write to SBUF," and activation of SEND.

SEND enables the output of the shift register to the alternate output function line of P3.0, and also enables SHIFT CLOCK to the alternate output function line of P3.1. SHIFT CLOCK is low during S3, S4, and S5 of every machine cycle, and high during S6, S1 and S2. At S6P2 of every machine cycle in which SEND is active, the contents of the transmit shift register are shifted to the right one position.

As data bits shift out to the right, zeroes come in from the left. When the MSB of the data byte is at the output position of the shift register, then the 1 that was initially loaded into the 9th position, is just to the left of the MSB, and all positions to the left of that contain zeroes. This condition flags the TX Control block to do one last shift and then deactivate SEND and set TI. Both of these actions occur at S1P1 of the 10th machine cycle after "write to SBUF."

Reception is initiated by the condition REN = 1 and RI = 0. At S6P2 of the next machine cycle, the RX Control unit writes the bits 11111110 to the receive shift register, and in the next clock phase activates RECEIVE.

RECEIVE enables SHIFT CLOCK to the alternate output function line of P3.1. SHIFT CLOCK makes transitions at S3P1 and S6P1 of every machine cycle. At S6P2 of every machine cycle in which RECEIVE is active, the contents of the receive shift register are shifted to the left one position. The value that comes in from the right is the value that was sampled at the P3.0 pin at S5P2 of the same machine cycle.

As data bits come in from the right, 1s shift out to the left. When the 0 that was initially loaded into the rightmost position arrives at the leftmost position in the shift register, it flags the RX Control block to do one last shift and load SBUF. At S1P1 of the 10th machine cycle after the write to SCON that cleared RI, RECEIVE is cleared and RI is set.

More About Mode 1

Ten bits are transmitted (through TXD), or received (through RXD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB8 in SCON. In the 8051 the baud rate is determined by the Timer 1 overflow rate. In the 8052 it is determined either by the Timer 1 overflow rate, or the Timer 2 overflow rate, or both (one for transmit and the other for receive).

Figure 18 shows a simplified functional diagram of the serial port in Mode 1, and associated timings for transmit receive.

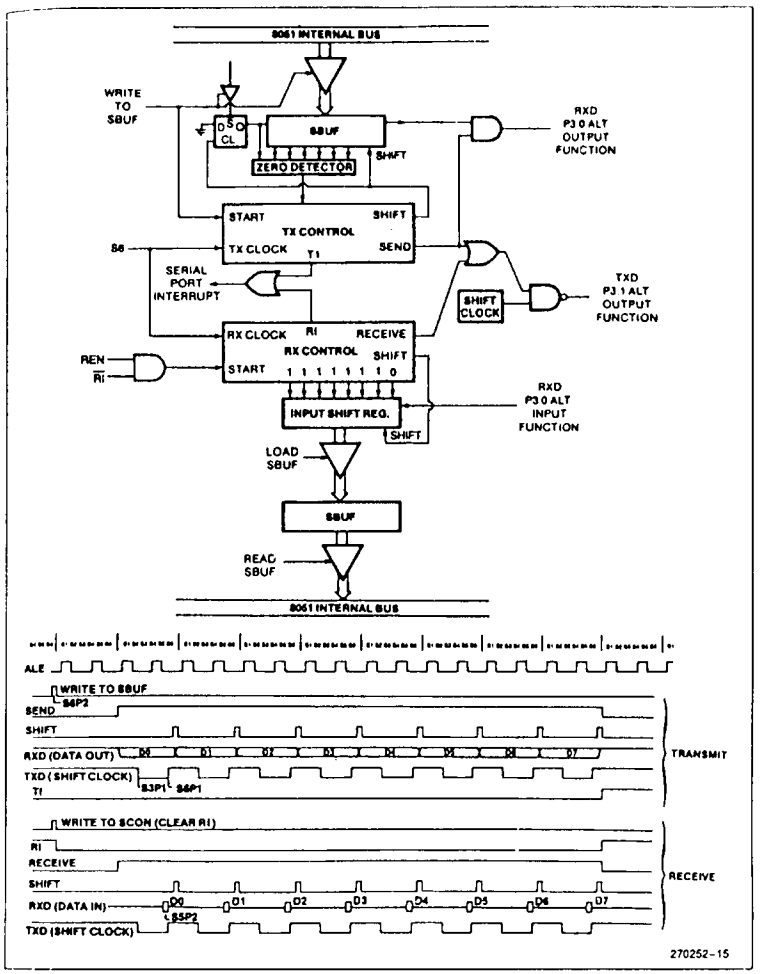


Figure 17. Serial Port Mode 0

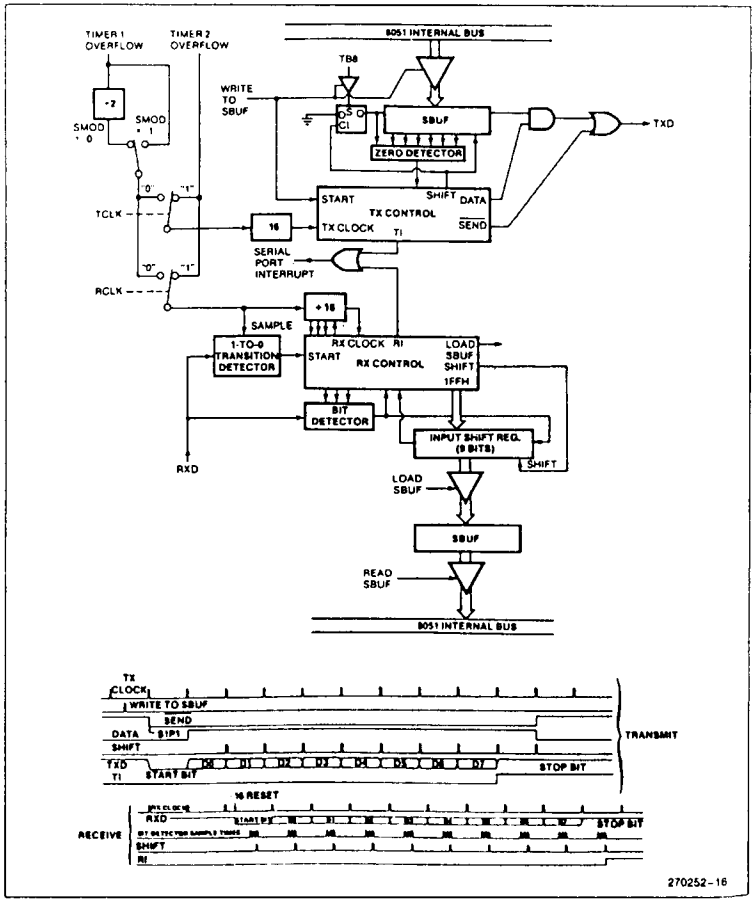


Figure 18. Serial Port Mode 1. TCLK, RCLK and Timer 2 are Present in the 8052/8032 Only.

Transmission is initiated by any instruction that uses SBUF as a destination register. The "write to SBUF" signal also loads a 1 into the 9th bit position of the transmit shift register and flags the TX Control unit that a transmission is requested. Transmission actually commences at S1P1 of the machine cycle following the next rollover in the divide-by-16 counter. (Thus, the bit times are synchronized to the divide-by-16 counter, not to the "write to SBUF" signal).

The transmission begins with activation of $\overline{\text{SEND}}$, which puts the start bit at TXD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TXD. The first shift pulse occurs one bit time after that.

As data bits shift out to the right, zeroes are clocked in from the left. When the MSB of the data byte is at the output position of the shift register, then the 1 that was initially loaded into the 9th position is just to the left of the MSB, and all positions to the left of that contain zeroes. This condition flags the TX Control unit to do one last shift and then deactivate $\overline{\text{SEND}}$ and set TI. This occurs at the 10th divide-by-16 rollover after "write to SBUF."

Reception is initiated by a detected 1-to-0 transition at RXD. For this purpose RXD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and 1FFH is written into the input shift register. Resetting the divide-by-16 counter aligns its rollovers with the boundaries of the incoming bit times.

The 16 states of the counter divide each bit time into 16ths. At the 7th, 8th, and 9th counter states of each bit time, the bit detector samples the value of RXD. The value accepted is the value that was seen in at least 2 of the 3 samples. This is done for noise rejection. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit goes back to looking for another 1-to-0 transition. This is to provide rejection of false start bits. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed.

As data bits come in from the right, 1s shift out to the left. When the start bit arrives at the leftmost position in the shift register, (which in mode 1 is a 9-bit register), it flags the RX Control block to do one last shift, load SBUF and RB8, and set RI. The signal to load SBUF and RB8, and to set RI, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated.

1) RI = 0, and
2) Either SM2 = 0, or the received stop bit = 1

If either of these two conditions is not met, the received frame is irretrievably lost. If both conditions are met, the stop bit goes into RB8, the 8 data bits go into SBUF, and RI is activated. At this time, whether the above conditions are met or not, the unit goes back to looking for a 1-to-0 transition in RXD.

More About Modes 2 and 3

Eleven bits are transmitted (through TXD), or received (through RXD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On transmit, the 9th data bit (TB8) can be assigned the value of 0 or 1. On receive, the 9th data bit goes into RB8 in SCON. The baud rate is programmable to either $1/32$ or $1/64$ the oscillator frequency in Mode 2. Mode 3 may have a variable baud rate generated from either Timer 1 or 2 depending on the state of TCLK and RCLK.

Figures 19 and 20 show a functional diagram of the serial port in Modes 2 and 3. The receive portion is exactly the same as in Mode 1. The transmit portion differs from Mode 1 only in the 9th bit of the transmit shift register.

Transmission is initiated by any instruction that uses SBUF as a destination register. The "write to SBUF" signal also loads TB8 into the 9th bit position of the transmit shift register and flags the TX Control unit that a transmission is requested. Transmission commences at S1P1 of the machine cycle following the next rollover in the divide-by-16 counter. (Thus, the bit times are synchronized to the divide-by-16 counter, not to the "write to SBUF" signal.)

The transmission begins with activation of SEND, which puts the start bit at TXD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TXD. The first shift pulse occurs one bit time after that. The first shift clocks a 1 (the stop bit) into the 9th bit position of the shift register. Thereafter, only zeroes are clocked in. Thus, as data bits shift out to the right, zeroes are clocked in from the left. When TB8 is at the output position of the shift register, then the stop bit is just to the left of TB8, and all positions to the left of that contain zeroes. This condition flags the TX Control unit to do one last shift and then deactivate SEND and set TI. This occurs at the 11th divide-by-16 rollover after "write to SBUF."

Reception is initiated by a detected 1-to-0 transition at RXD. For this purpose RXD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and 1FFH is written to the input shift register.

At the 7th, 8th and 9th counter states of each bit time, the bit detector samples the value of RXD. The value accepted is the value that was seen in at least 2 of the 3 samples. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit goes back to looking for another 1-to-0 transition. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed.

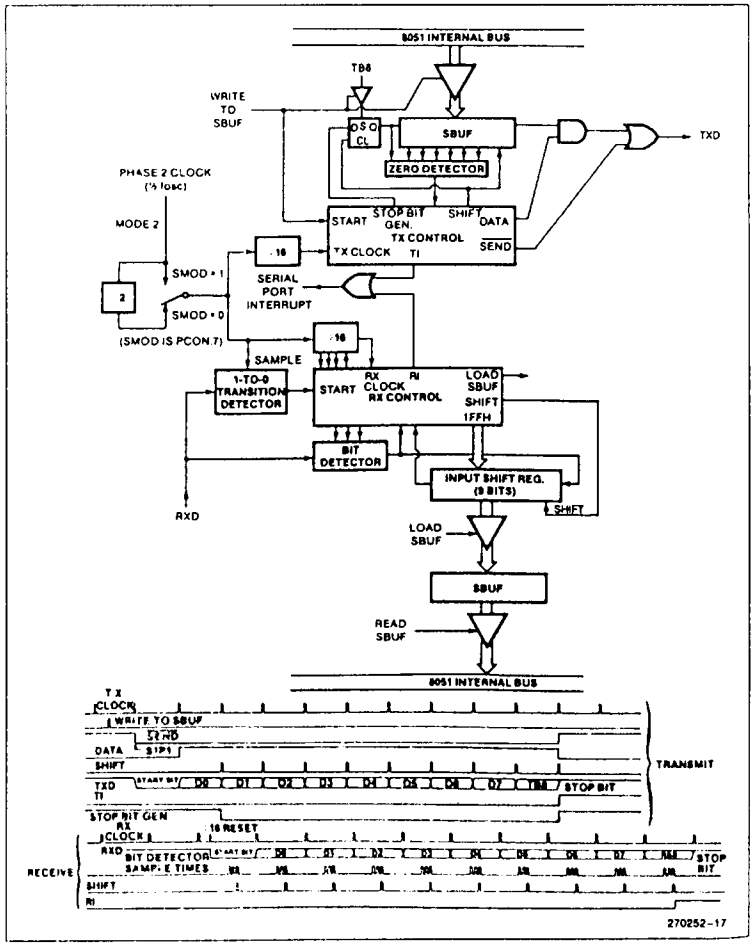


Figure 19. Serial Port Mode 2

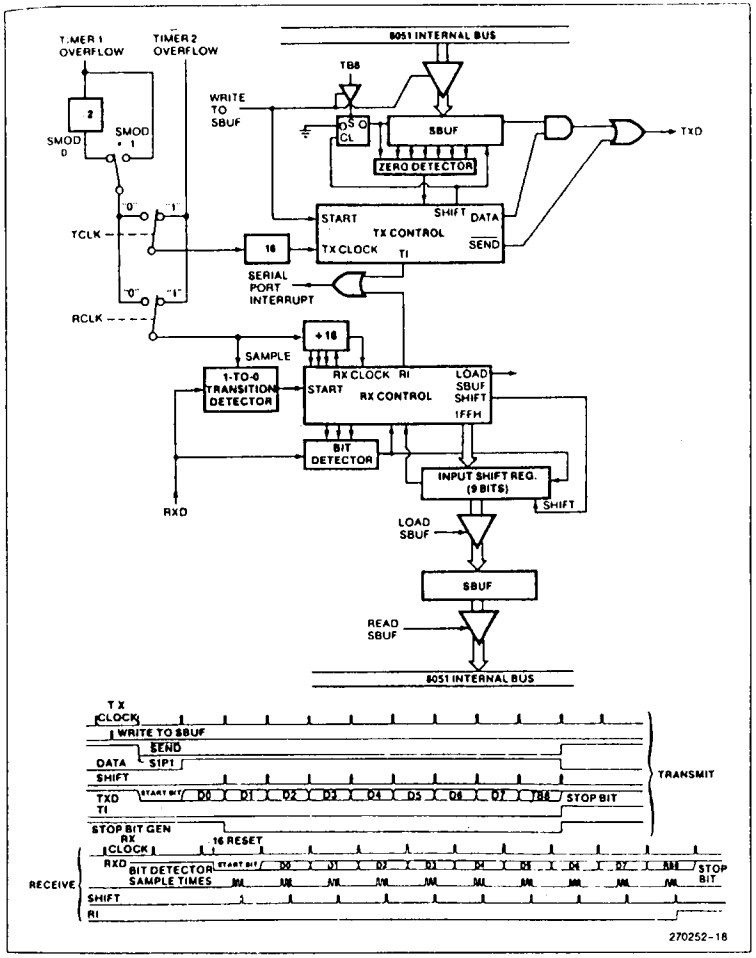


Figure 20. Serial Port Mode 3. TCLK, RCLK, and Timer 2 are Present in the 8052/8032 Only.

As data bits come in from the right, 1s shift out to the left. When the start bit arrives at the leftmost position in the shift register (which in Modes 2 and 3 is a 9-bit register), it flags the RX Control block to do one last shift, load SBUF and RB8, and set RI. The signal to load SBUF and RB8, and to set RI, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated:

1) RI = 0, and
2) Either SM2 = 0 or the received 9th data bit = 1

If either of these conditions is not met, the received frame is irretrievably lost, and RI is not set. If both conditions are met, the received 9th data bit goes into RB8, and the first 8 data bits go into SBUF. One bit time later, whether the above conditions were met or not, the unit goes back to looking for a 1-to-0 transition at the RXD input.

Note that the value of the received stop bit is irrelevant to SBUF, RB8, or RI.

INTERRUPTS

The 8051 provides 5 interrupt sources. The 8052 provides 6. These are shown in Figure 21.

The External Interrupts $\overline{INT0}$ and $\overline{INT1}$ can each be either level-activated or transition-activated, depending on bits IT0 and IT1 in Register TCON. The flags that actually generate these interrupts are bits IE0 and IE1 in TCON. When an external interrupt is generated, the flag that generated it is cleared by the hardware when the service routine is vectored to only if the interrupt was transition-activated. If the interrupt was level-activated, then the external requesting source is what controls the request flag, rather than the on-chip hardware.

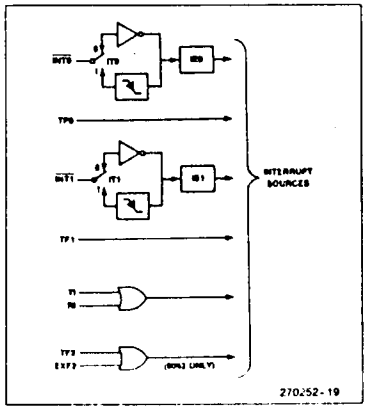


Figure 21. MCS®-51 Interrupt Sources

The Timer 0 and Timer 1 Interrupts are generated by TF0 and TF1, which are set by a rollover in their respective Timer/Counter registers (except see Timer 0 in Mode 3). When a timer interrupt is generated, the flag that generated it is cleared by the on-chip hardware when the service routine is vectored to.

The Serial Port Interrupt is generated by the logical OR of RI and TI. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine will normally have to determine whether it was RI or TI that generated the interrupt, and the bit will have to be cleared in software.

In the 8052, the Timer 2 Interrupt is generated by the logical OR of TF2 and EXF2. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine may have to determine whether it was TF2 or EXF2 that generated the interrupt, and the bit will have to be cleared in software.

All of the bits that generate interrupts can be set or cleared by software, with the same result as though it had been set or cleared by hardware. That is, interrupts can be generated or pending interrupts can be canceled in software.

(MSB)							(LSB)
EA	–	ET2	ES	ET1	EX1	ET0	EX0

Enable Bit = 1 enables the interrupt.
Enable Bit = 0 disables it.

Symbol	Position	Function
EA	IE.7	disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
–	IE.6	reserved.
ET2	IE.5	Timer 2 interrupt enable bit.
ES	IE.4	Serial Port interrupt enable bit.
ET1	IE.3	Timer 1 interrupt enable bit.
EX1	IE.2	External interrupt 1 enable bit.
ET0	IE.1	Timer 0 interrupt enable bit.
EX0	IE.0	External interrupt 0 enable bit.

User software should never write 1s to unimplemented bits, since they may be used in future MCS-51 products.

Figure 22. IE: Interrupt Enable Register

Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE (Figure 22). IE contains also a global disable bit, EA, which disables all interrupts at once.

Note in Figure 22 that bit position IE.6 is unimplemented. In the 8051s, bit position IE.5 is also unimplemented. User software should not write 1s to these bit positions, since they may be used in future MCS-51 products.

Priority Level Structure

Each interrupt source can also be individually programmed to one of two priority levels by setting or clearing a bit in Special Function Register IP (Figure 23). A low-priority interrupt can itself be interrupted by a high-priority interrupt, but not by another low-priority interrupt. A high-priority interrupt can't be interrupted by any other interrupt source.

(MSB)							(LSB)
–	–	PT2	PS	PT1	PX1	PT0	PX0

Priority bit = 1 assigns high priority.
Priority bit = 0 assigns low priority.

Symbol	Position	Function
–	IP.7	reserved
–	IP.6	reserved
PT2	IP.5	Timer 2 interrupt priority bit.
PS	IP.4	Serial Port interrupt priority bit.
PT1	IP.3	Timer 1 interrupt priority bit.
PX1	IP.2	External interrupt 1 priority bit.
PT0	IP.1	Timer 0 interrupt priority bit.
PX0	IP.0	External interrupt 0 priority bit.

User software should never write 1s to unimplemented bits, since they may be used in future MCS-51 products.

Figure 23. IP: Interrupt Priority Register

If two requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence, as follows:

	Source	Priority Within Level
1.	IE0	(highest)
2.	TF0	
3.	IE1	
4.	TF1	
5.	RI + TI	
6.	TF2 + EXF2	(lowest)

Note that the "priority within level" structure is only used to resolve *simultaneous requests of the same priority level.*

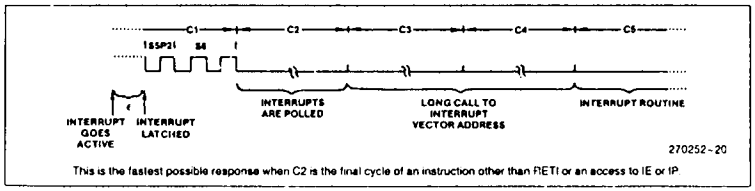
The IP register contains a number of unimplemented bits. IP.7 and IP.6 are vacant in the 8052s, and in the 8051s these and IP.5 are vacant. User software should not write 1s to these bit positions, since they may be used in future MCS-51 products.

How Interrupts Are Handled

The interrupt flags are sampled at S5P2 of every machine cycle. The samples are polled during the following machine cycle. The 8052's Timer 2 interrupt cycle is different, as described in the Response Time Section. If one of the flags was in a set condition at S5P2 of the preceding cycle, the polling cycle will find it and the interrupt system will generate an LCALL to the appropriate service routine, provided this hardware-generated LCALL is not blocked by any of the following conditions:

1. An interrupt of equal or higher priority level is already in progress.
2. The current (polling) cycle is not the final cycle in the execution of the instruction in progress.
3. The instruction in progress is RETI or any write to the IE or IP registers.

Any of these three conditions will block the generation of the LCALL to the interrupt service routine. Condition 2 ensures that the instruction in progress will be



This is the fastest possible response when C2 is the final cycle of an instruction other than RETI or an access to IE or IP.

Figure 24. Interrupt Response Timing Diagram

completed before vectoring to any service routine. Condition 3 ensures that if the instruction in progress is RETI or any access to IE or IP, then at least *one more* instruction will be executed before any interrupt is vectored to.

The polling cycle is repeated with each machine cycle, and the values polled are the values that were present at S5P2 of the previous machine cycle. Note then that if an interrupt flag is active but not being responded to for one of the above conditions, and is not *still* active when the blocking condition is removed, the denied interrupt will not be serviced. In other words, the fact that the interrupt flag was once active but not serviced is not remembered. Every polling cycle is new.

The polling cycle/LCALL sequence is illustrated in Figure 24.

Note that if an interrupt of higher priority level goes active prior to S5P2 of the machine cycle labeled C3 in Figure 24, then in accordance with the above rules it will be vectored to during C5 and C6, without any instruction of the lower priority routine having been executed.

Thus the processor acknowledges an interrupt request by executing a hardware-generated LCALL to the appropriate servicing routine. In some cases it also clears the flag that generated the interrupt, and in other cases it doesn't. It never clears the Serial Port or Timer 2 flags. This has to be done in the user's software. It clears an external interrupt flag (IE0 or IE1) only if it was transition-activated. The hardware-generated LCALL pushes the contents of the Program Counter onto the stack (but it does not save the PSW) and reloads the PC with an address that depends on the source of the interrupt being vectored to, as shown below.

Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI + TI	0023H
TF2 + EXF2	002BH

Execution proceeds from that location until the RETI instruction is encountered. The RETI instruction informs the processor that this interrupt routine is no longer in progress, then pops the top two bytes from the stack and reloads the Program Counter. Execution of the interrupted program continues from where it left off.

Note that a simple RET instruction would also have returned execution to the interrupted program, but it would have left the interrupt control system thinking an interrupt was still in progress.

External Interrupts

The external sources can be programmed to be level-activated or transition-activated by setting or clearing bit IT1 or IT0 in Register TCON. If ITx = 0, external interrupt x is triggered by a detected low at the $\overline{INTx}$ pin. If ITx = 1, external interrupt x is edge-triggered. In this mode if successive samples of the $\overline{INTx}$ pin show a high in one cycle and a low in the next cycle, interrupt request flag IEx in TCON is set. Flag bit IEx then requests the interrupt.

Since the external interrupt pins are sampled once each machine cycle, an input high or low should hold for at least 12 oscillator periods to ensure sampling. If the external interrupt is transition-activated, the external source has to hold the request pin high for at least one machine cycle, and then hold it low for at least one machine cycle to ensure that the transition is seen so that interrupt request flag IEx will be set. IEx will be automatically cleared by the CPU when the service routine is called.

If the external interrupt is level-activated, the external source has to hold the request active until the requested interrupt is actually generated. Then it has to deactivate the request before the interrupt service routine is completed, or else another interrupt will be generated.

Response Time

The $\overline{INT0}$ and $\overline{INT1}$ levels are inverted and latched into the interrupt flags IE0 and IE1 at S5P2 of every machine cycle. Similarly, the Timer 2 flag EXF2 and the Serial Port flags RI and TI are set at S5P2. The values are not actually polled by the circuitry until the next machine cycle.

The Timer 0 and Timer 1 flags, TF0 and TF1, are set at S5P2 of the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle. However, the Timer 2 flag TF2 is set at S2P2 and is polled in the same cycle in which the timer overflows.

If a request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The call itself takes two cycles. Thus, a minimum of three complete machine cycles elapse between activation of an external interrupt request and the beginning of execution of the first instruction of the service routine. Figure 24 shows interrupt response timings.

A longer response time would result if the request is blocked by one of the 3 previously listed conditions. If an interrupt of equal or higher priority level is already in progress, the additional wait time obviously depends on the nature of the other interrupt's service routine. If the instruction in progress is not in its final cycle, the additional wait time cannot be more than 3 cycles, since the longest instructions (MUL and DIV) are only 4

cycles long, and if the instruction in progress is RETI or an access to IE or IP, the additional wait time cannot be more than 5 cycles (a maximum of one more cycle to complete the instruction in progress, plus 4 cycles to complete the next instruction if the instruction is MUL or DIV).

Thus, in a single-interrupt system, the response time is always more than 3 cycles and less than 9 cycles.

SINGLE-STEP OPERATION

The 8051 interrupt structure allows single-step execution with very little software overhead. As previously noted, an interrupt request will not be responded to while an interrupt of equal priority level is still in progress, nor will it be responded to after RETI until at least one other instruction has been executed. Thus, once an interrupt routine has been entered, it cannot be re-entered until at least one instruction of the interrupted program is executed. One way to use this feature for single-stop operation is to program one of the external interrupts (say, $\overline{INT0}$) to be level-activated. The service routine for the interrupt will terminate with the following code:

```
JNB   P3.2,$  ;Wait Here Till INT0 Goes High
JB    P3.2,$  ;Now Wait Here Till It Goes Low
RETI          :Go Back and Execute One Instruction
```

Now if the $\overline{INT0}$ pin, which is also the P3.2 pin, is held normally low, the CPU will go right into the External Interrupt 0 routine and stay there until $\overline{INT0}$ is pulsed (from low to high to low). Then it will execute RETI, go back to the task program, execute one instruction, and immediately re-enter the External Interrupt 0 routine to await the next pulsing of P3.2. One step of the task program is executed each time P3.2 is pulsed.

RESET

The reset input is the RST pin, which is the input to a Schmitt Trigger.

A reset is accomplished by holding the RST pin high for at least two machine cycles (24 oscillator periods), *while the oscillator is running*. The CPU responds by generating an internal reset, with the timing shown in Figure 25.

The external reset signal is asynchronous to the internal clock. The RST pin is sampled during State 5 Phase 2 of every machine cycle. The port pins will maintain their current activities for 19 oscillator periods after a logic 1 has been sampled at the RST pin; that is, for 19 to 31 oscillator periods after the external reset signal has been applied to the RST pin.

While the RST pin is high, ALE and PSEN are weakly pulled high. After RST is pulled low, it will take 1 to 2 machine cycles for ALE and PSEN to start clocking. For this reason, other devices can not be synchronized to the internal timings of the 8051.

Driving the ALE and $\overline{PSEN}$ pins to 0 while reset is active could cause the device to go into an indeterminate state.

The internal reset algorithm writes 0s to all the SFRs except the port latches, the Stack Pointer, and SBUF. The port latches are initialized to FFH, the Stack Pointer to 07H, and SBUF is indeterminate. Table 3 lists the SFRs and their reset values.

The internal RAM is not affected by reset. On power up the RAM content is indeterminate.

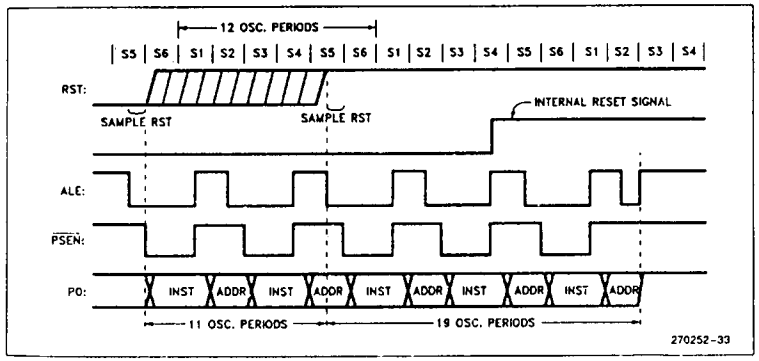


Figure 25. Reset Timing

Table 3. Reset Values of the SFRs

SFR Name	Reset Value
PC	0000H
ACC	00H
B	00H
PSW	00H
SP	07H
DPTR	0000H
P0–P3	FFH
IP (8051)	XXX00000B
IP (8052)	XX000000B
IE (8051)	0XX00000B
IE (8052)	0X000000B
TMOD	00H
TCON	00H
TH0	00H
TL0	00H
TH1	00H
TL1	00H
TH2 (8052)	00H
TL2 (8052)	00H
RCAP2H (8052)	00H
RCAP2L (8052)	00H
SCON	00H
SBUF	Indeterminate
PCON (HMOS)	0XXXXXXXB
PCON (CHMOS)	0XXX0000B

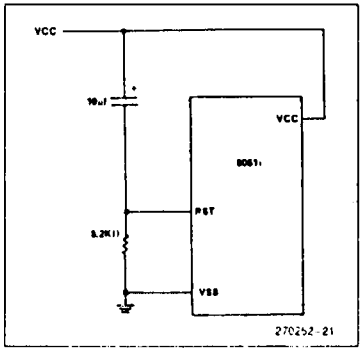


Figure 26. Power on Reset Circuit

POWER-ON RESET

For HMOS devices when V_{CC} is turned on an automatic reset can be obtained by connecting the RST pin to V_{CC} through a 10 μF capacitor and to V_{SS} through an 8.2 KΩ resistor (Figure 26). The CHMOS devices do not require this resistor although its presence does no harm. In fact, for CHMOS devices the external resistor can be removed because they have an internal pulldown on the RST pin. The capacitor value could then be reduced to 1 μF.

When power is turned on, the circuit holds the RST pin high for an amount of time that depends on the capacitor value and the rate at which it charges. To ensure a valid reset the RST pin must be held high long enough to allow the oscillator to start up plus two machine cycles.

On power up, V_{CC} should rise within approximately ten milliseconds. The oscillator start-up time will depend on the oscillator frequency. For a 10 MHz crystal, the start-up time is typically 1 ms. For a 1 MHz crystal, the start-up time is typically 10 ms.

With the given circuit, reducing V_{CC} quickly to 0 causes the RST pin voltage to momentarily fall below 0V. However, this voltage is internally limited and will not harm the device.

NOTE:
The port pins will be in a random state until the oscillator has started and the internal reset algorithm has written 1s to them.

Powering up the device without a valid reset could cause the CPU to start executing instructions from an indeterminate location. This is because the SFRs, specifically the Program Counter, may not get properly initialized.

POWER-SAVING MODES OF OPERATION

For applications where power consumption is critical the CHMOS version provides power reduced modes of operation as a standard feature. The power down mode in HMOS devices is no longer a standard feature and is being phased out.

CHMOS Power Reduction Modes

CHMOS versions have two power-reducing modes, Idle and Power Down. The input through which backup power is supplied during these operations is VCC. Figure 27 shows the internal circuitry which implements these features. In the Idle mode (IDL = 1), the oscillator continues to run and the Interrupt, Serial Port, and Timer blocks continue to be clocked, but the

clock signal is gated off to the CPU. In Power Down (PD = 1), the oscillator is frozen. The Idle and Power Down modes are activated by setting bits in Special Function Register PCON. The address of this register is 87H. Figure 26 details its contents.

In the HMOS devices the PCON register only contains SMOD. The other four bits are implemented only in the CHMOS devices. User software should never write 1s to unimplemented bits, since they may be used in future MCS-51 products.

IDLE MODE

An instruction that sets PCON.0 causes that to be the last instruction executed before going into the Idle mode. In the Idle mode, the internal clock signal is gated off to the CPU, but not to the Interrupt, Timer, and Serial Port functions. The CPU status is preserved in its entirety: the Stack Pointer, Program Counter, Program Status Word, Accumulator, and all other registers maintain their data during Idle. The port pins hold the logical states they had at the time Idle was activated. ALE and $\overline{PSEN}$ hold at logic high levels.

There are two ways to terminate the Idle. Activation of any enabled interrupt will cause PCON.0 to be cleared by hardware, terminating the Idle mode. The interrupt will be serviced, and following RETI the next instruction to be executed will be the one following the instruction that put the device into Idle.

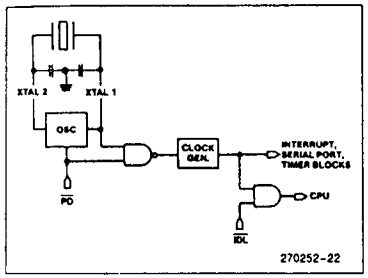


Figure 27. Idle and Power Down Hardware

(MSB)							(LSB)
SMOD	-	-	-	GF1	GF0	PD	IDL

Symbol	Position	Name and Function
SMOD	PCON.7	Double Baud rate bit. When set to a 1 and Timer 1 is used to generate baud rate, and the Serial Port is used in modes 1, 2, or 3.
—	PCON.6	(Reserved)
—	PCON.5	(Reserved)
—	PCON.4	(Reserved)
GF1	PCON.3	General-purpose flag bit.
GF0	PCON.2	General-purpose flag bit.
PD	PCON.1	Power Down bit. Setting this bit activates power down operation.
IDL	PCON.0	Idle mode bit. Setting this bit activates idle mode operation.

If 1s are written to PD and IDL at the same time, PD takes precedence. The reset value of PCON is (0XXX0000).
In the HMOS devices the PCON register only contains SMOD. The other four bits are implemented only in the CHMOS devices. User software should never write 1s to unimplemented bits, since they may be used in future MCS-51 products.

Figure 28. PCON: Power Control Register

The flag bits GF0 and GF1 can be used to give an indication if an interrupt occurred during normal operation or during an Idle. For example, an instruction that activates Idle can also set one or both flag bits. When Idle is terminated by an interrupt, the interrupt service routine can examine the flag bits.

The other way of terminating the Idle mode is with a hardware reset. Since the clock oscillator is still running, the hardware reset needs to be held active for only two machine cycles (24 oscillator periods) to complete the reset.

The signal at the RST pin clears the IDL bit directly and asynchronously. At this time the CPU resumes program execution from where it left off; that is, at the instruction following the one that invoked the Idle Mode. As shown in Figure 25, two or three machine cycles of program execution may take place before the internal reset algorithm takes control. On-chip hardware inhibits access to the internal RAM during this time, but access to the port pins is not inhibited. To eliminate the possibility of unexpected outputs at the port pins, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external Data RAM.

POWER DOWN MODE

An instruction that sets PCON.1 causes that to be the last instruction executed before going into the Power Down mode. In the Power Down mode, the on-chip oscillator is stopped. With the clock frozen, all func-

Table 4. EPROM Versions of the 8051 and 8052

Device Name	EPROM Version	EPROM Bytes	Ckt Type	VPP	Time Required to Program Entire Array
8051	(8751)	4K	HMOS	21.0V	4 minutes
8051AH	8751H	4K	HMOS	21.0V	4 minutes
80C51BH	87C51	4K	CHMOS	12.75V	13 seconds
8052AH	8752BH	8K	HMOS	12.75V	26 seconds

tions are stopped, but the on-chip RAM and Special Function Registers are held. The port pins output the values held by their respective SFRs. ALE and $\overline{PSEN}$ output lows.

The only exit from Power Down for the 80C51 is a hardware reset. Reset redefines all the SFRs, but does not change the on-chip RAM.

In the Power Down mode of operation, VCC can be reduced to as low as 2V. Care must be taken, however, to ensure that VCC is not reduced before the Power Down mode is invoked, and that VCC is restored to its normal operating level, before the Power Down mode is terminated. The reset that terminates Power Down also frees the oscillator. The reset should not be activated before VCC is restored to its normal operating level, and must be held active long enough to allow the oscillator to restart and stabilize (normally less than 10 msec).

EPROM VERSIONS

The EPROM versions of these devices are listed in Table 4. The 8751H programs at VPP = 21V using one 50 msec $\overline{PROG}$ pulse per byte programmed. This results in a total programming time (4K bytes) of approximately 4 minutes.

The 8752BH and 87C51 use the faster "Quick-Pulse" programming™ algorithm. These devices program at VPP = 12.75V using a series of twenty-five 100 µs $\overline{PROG}$ pulses per byte programmed. This results in a total programming time of approximately 26 seconds for the 8752BH (8K bytes) and 13 seconds for the 87C51 (4K bytes).

Detailed procedures for programming and verifying each device are given in the data sheets.

EXPOSURE TO LIGHT

It is good practice to cover the EPROM window with an opaque label when the device is in operation. This is not so much to protect the EPROM array from inadvertent erasure, but to protect the RAM and other on-chip logic. Allowing light to impinge on the silicon die while the device is operating can cause logical malfunction.

Program Memory Locks

In some microcontroller applications it is desirable that the Program Memory be secure from software piracy. Intel has responded to this need by implementing a Program Memory locking scheme in some of the MCS-51 devices. While it is impossible for anyone to guarantee absolute security against all levels of technological sophistication, the Program Memory locks in the MCS-51 devices will present a formidable barrier against illegal readout of protected software.

One Lock Bit Scheme on 8751H

The 8751H contains a lock bit which, once programmed, denies electrical access by any external means to the on-chip Program Memory. The effect of this lock bit is that while it is programmed the internal Program Memory can not be read out, the device can not be further programmed, and it *can not execute external Program Memory*. Erasing the EPROM array deactivates the lock bit and restores the device's full functionality. It can then be re-programmed.

The procedure for programming the lock bit is detailed in the 8751H data sheet.

Two-Level Program Memory Lock Scheme

The 87C51 and 8752BH contain two Program Memory locking schemes: Encrypted Verify and Lock Bits.

Encrypted Verify: These devices implement a 32-byte EPROM array that can be programmed by the customer, and which can then be used to encrypt the program code bytes during EPROM verification. The EPROM verification procedure is performed as usual, except that each code byte comes out X-NORed with one of the 32 key bytes. The key bytes are gone through in sequence. Therefore, to read the ROM code, one has to know the 32 key bytes in their proper sequence.

Unprogrammed bytes have the value FFH. Therefore, if the Encryption Array is left unprogrammed all the key bytes have the value FFH. Since any code byte X-NORed with FFH leaves the code byte unchanged, leaving the Encryption Array unprogrammed in effect bypasses the encryption feature.

Lock Bits: Also on the chip are two Lock Bits which can be left unprogrammed (U) or programmed (P) to obtain the following features:

Bit 2	Bit 1	Additional Features
U	U	None
U	P	• Externally fetched code can not access internal Program Memory. • Further programming disabled.
P	U	(Reserved for Future definition.)
P	P	• Externally fetched code can not access internal Program Memory. • Further programming disabled. • Program verification is disabled.

When Lock Bit 1 is programmed, the logic level at the $\overline{EA}$ pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value, and holds that value until reset is activated. It is necessary that the latched value of $\overline{EA}$ be in agreement with the current logic level at that pin in order for the device to function properly.

ROM Protection

The 8051AHP and 80C51BHP are ROM Protected versions of the 8051AH and 80C51BH, respectively. To incorporate this Protection Feature, program verification has been disabled and external memory accesses have been limited to 4K. Refer to the data sheets on these parts for more information.

ONCE Mode

The ONCE ("on-circuit emulation") mode facilitates testing and debugging of systems using the device without the device having to be removed from the circuit. The ONCE mode is invoked by:

1. Pull ALE low while the device is in reset and $\overline{PSEN}$ is high;
2. Hold ALE low as RST is deactivated.

While the device is in ONCE mode, the Port 0 pins go into a float state, and the other port pins and ALE and $\overline{PSEN}$ are weakly pulled high. The oscillator circuit remains active. While the device is in this mode, an emulator or test CPU can be used to drive the circuit. Normal operation is restored after a normal reset is applied.

THE ON-CHIP OSCILLATORS

HMOS Versions

The on-chip oscillator circuitry for the HMOS (HMOS-I and HMOS-II) members of the MCS-51 family is a single stage linear inverter (Figure 29), intended for use as a crystal-controlled, positive reactance oscillator (Figure 30). In this application the crystal is operated in its fundamental response mode as an inductive reactance in parallel resonance with capacitance external to the crystal.

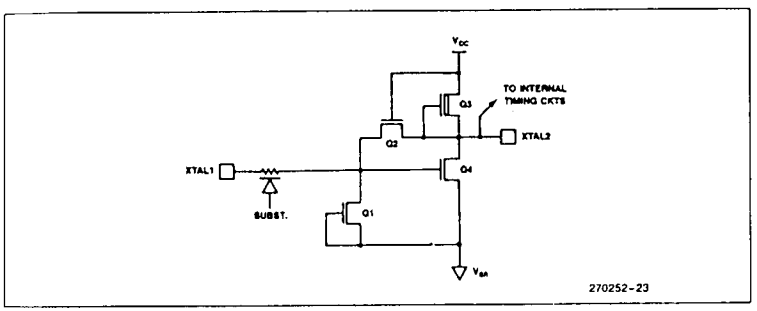


Figure 29. On-Chip Oscillator Circuitry in the HMOS Versions of the MCS®-51 Family

7

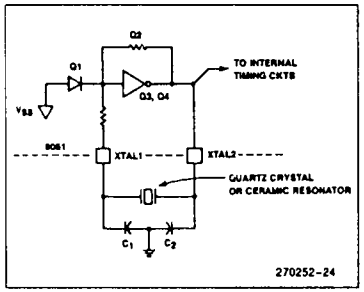


Figure 30. Using the HMOS On-Chip Oscillator

The crystal specifications and capacitance values (C1 and C2 in Figure 30) are not critical. 30 pF can be used in these positions at any frequency with good quality crystals. A ceramic resonator can be used in place of the crystal in cost-sensitive applications. When a ceramic resonator is used, C1 and C2 are normally selected to be of somewhat higher values, typically, 47 pF. The manufacturer of the ceramic resonator should be consulted for recommendations on the values of these capacitors.

In general, crystals used with these devices typically have the following specifications:

ESR (Equivalent Series Resistance)	see Figure 31
C_O (Shunt Capacitance)	7.0 pF max.
C_L (Load Capacitance)	30 pF ±3 pF
Drive Level	1 mW

Frequency, tolerance and temperature range are determined by the system requirements.

A more in-depth discussion of crystal specifications, ceramic resonators, and the selection of values for C1 and C2 can be found in Application Note AP-155, "Oscillators for Microcontrollers," which is included in the *Embedded Control Applications Handbook*.

To drive the HMOS parts with an external clock source, apply the external clock signal to XTAL2, and ground XTAL1, as shown in Figure 32. A pullup resistor may be used (to increase noise margin), but is optional if VOH of the driving gate exceeds the VIH MIN specification of XTAL2.

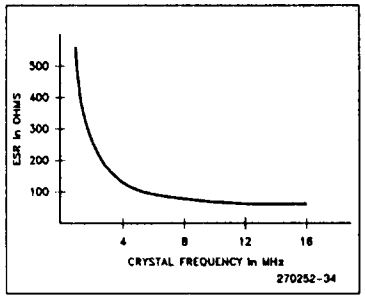


Figure 31. ESR vs Frequency

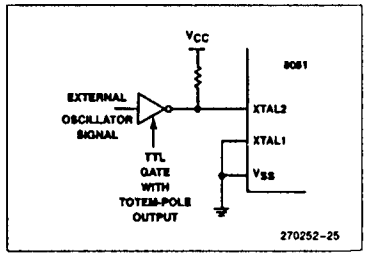


Figure 32. Driving the HMOS MCS®-51 Parts with an External Clock Source

CHMOS VERSIONS

The on-chip oscillator circuitry for the 80C51BH, shown in Figure 33, consists of a single stage linear inverter intended for use as a crystal-controlled, positive reactance oscillator in the same manner as the HMOS parts. However, there are some important differences.

One difference is that the 80C51BH is able to turn off its oscillator under software control (by writing a 1 to the PD bit in PCON). Another difference is that in the 80C51BH the internal clocking circuitry is driven by the signal at XTAL1, whereas in the HMOS versions it is by the signal at XTAL2.

The feedback resistor R_f in Figure 33 consists of paralleled n- and p- channel FETs controlled by the PD bit, such that R_f is opened when PD = 1. The diodes D1 and D2, which act as clamps to VCC and VSS, are parasitic to the R_f FETs.

The oscillator can be used with the same external components as the HMOS versions, as shown in Figure 34. Typically, C1 = C2 = 30 pF when the feedback element is a quartz crystal, and C1 = C2 = 47 pF when a ceramic resonator is used.

To drive the CHMOS parts with an external clock source, apply the external clock signal to XTAL1, and leave XTAL2 float, as shown in Figure 35.

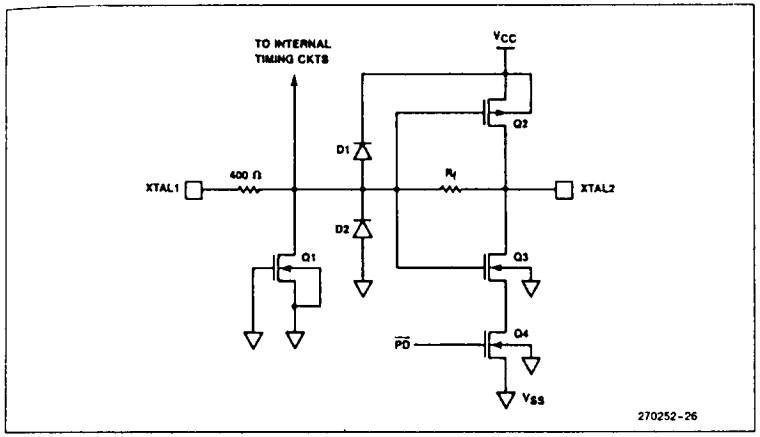


Figure 33. On-Chip Oscillator Circuitry in the CHMOS Versions of the MCS®-51 Family

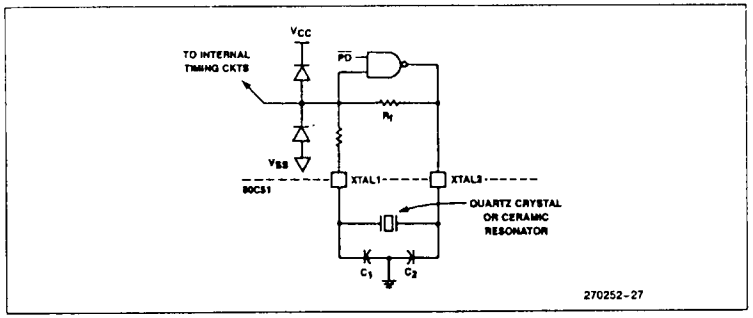


Figure 34. Using the CHMOS On-Chip Oscillator

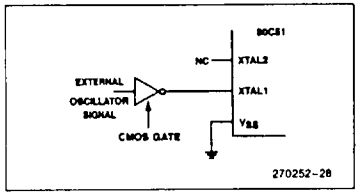


Figure 35. Driving the CHMOS MCS®-51 Parts with an External Clock Source

The reason for this change from the way the HMOS part is driven can be seen by comparing Figures 29 and 33. In the HMOS devices the internal timing circuits are driven by the signal at XTAL2. In the CHMOS devices the internal timing circuits are driven by the signal at XTAL1.

INTERNAL TIMING

Figures 36 through 39 show when the various strobe and port signals are clocked internally. The figures do not show rise and fall times of the signals, nor do they show propagation delays between the XTAL signal and events at other pins.

Rise and fall times are dependent on the external loading that each pin must drive. They are often taken to be something in the neighborhood of 10 nsec, measured between 0.8V and 2.0V.

Propagation delays are different for different pins. For a given pin they vary with pin loading, temperature, VCC, and manufacturing lot. If the XTAL waveform is taken as the timing reference, prop delays may vary from 25 to 125 nsec.

The AC Timings section of the data sheets do not reference any timing to the XTAL waveform. Rather, they relate the critical edges of control and input signals to each other. The timings published in the data sheets include the effects of propagation delays under the specified test conditions.

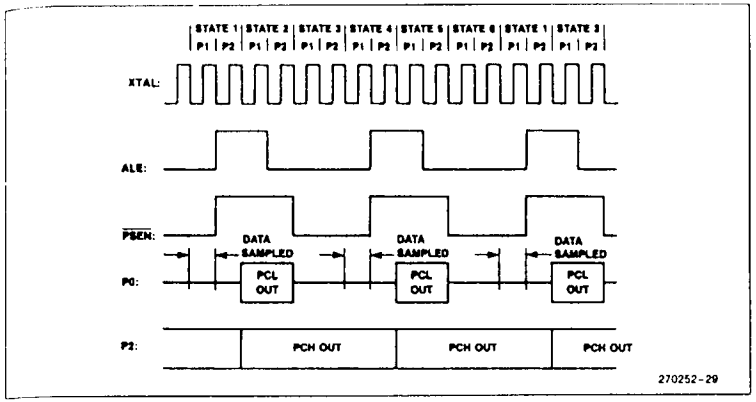


Figure 36. External Program Memory Fetches

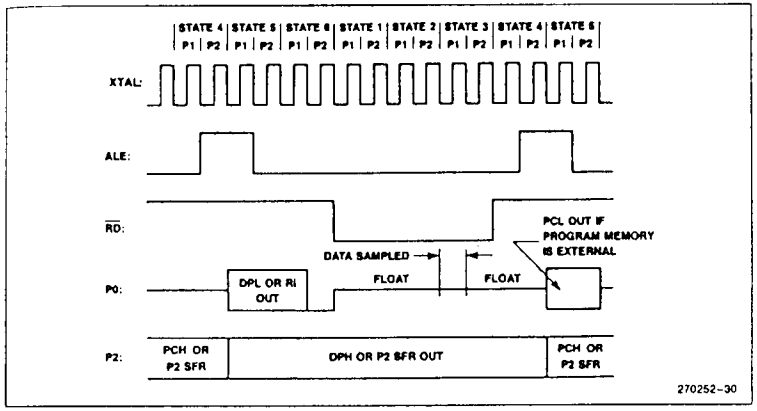


Figure 37. External Data Memory Read Cycle

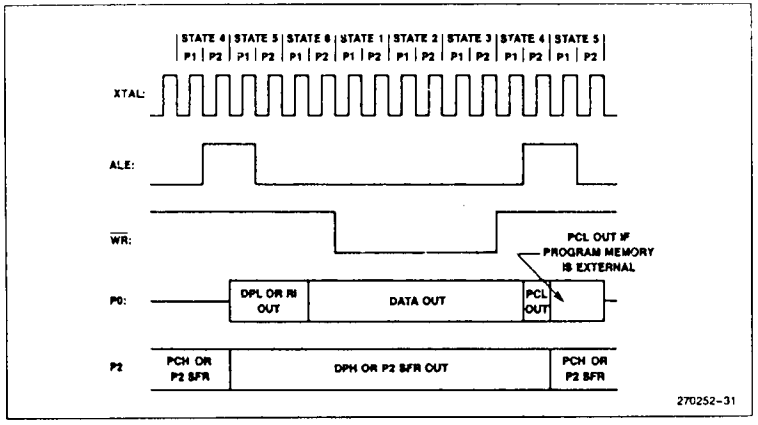


Figure 38. External Data Memory Write Cycle

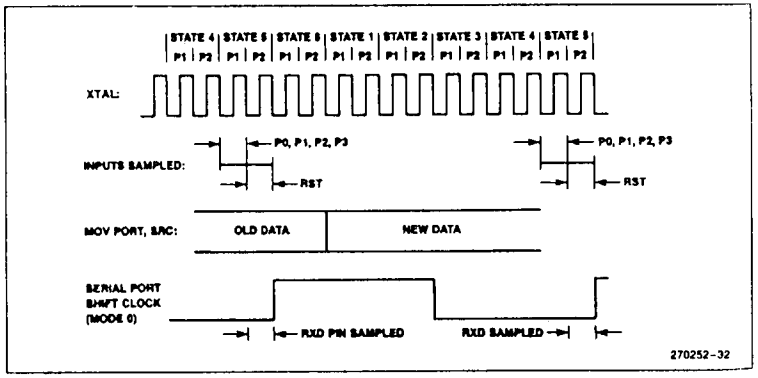


Figure 39. Port Operation

ADDITIONAL REFERENCES

The following application notes and articles are found in the *Embedded Control Applications* handbook. (Order Number: 270535)

1. AP-125 "Designing Microcontroller Systems for Electrically Noisy Environments".
2. AP-155 "Oscillators for Microcontrollers".
3. AP-252 "Designing with the 80C51BH".
4. AR-409 "Increased Functions in Chip Result in Lighter, Less Costly Portable Computer".
5. AR-517 "Using the 8051 Microcontroller with Resonant Transducers".

intel

8755A/8755A-2
16,384-BIT EPROM WITH I/O

- 2048 Words × 8 Bits
- Single +5V Power Supply (V_{CC})
- Directly Compatible with 8085A and 8088 Microprocessors
- U.V. Erasable and Electrically Reprogrammable
- Internal Address Latch
- 2 General Purpose 8-Bit I/O Ports
- Each I/O Port Line Individually Programmable as Input or Output
- Multiplexed Address and Data Bus
- 40-Pin DIP

The Intel® 8755A is an erasable and electrically reprogrammable ROM (EPROM) and I/O chip to be used in the 8085A and 8088 microprocessor systems. The EPROM portion is organized as 2048 words by 8 bits. It has a maximum access time of 450 ns to permit use with no wait states in an 8085A CPU.

The I/O portion consists of 2 general purpose I/O ports. Each I/O port has 8 port lines, and each I/O port line is individually programmable as input or output.

The 8755A-2 is a high speed selected version of the 8755A compatible with the 5 MHz 8085A-2 and the full speed 5 MHz 8088.

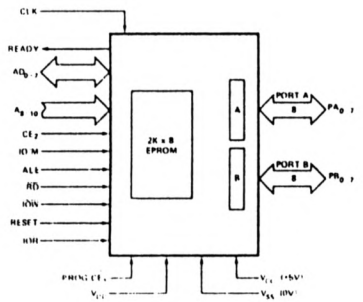


Figure 1. Block Diagram

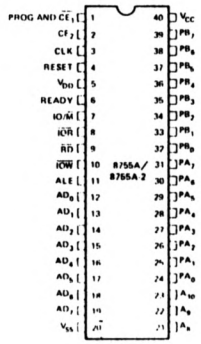


Figure 2. Pin Configuration

Table 1. Pin Description

Symbol	Type	Name and Function
ALE	I	**Address Latch Enable:** When Address Latch Enable goes *high*, AD_{0-7}, IO/M, A_{8-10}, CE_2, and $\overline{CE_1}$ enter the address latches. The signals (AD, IO/M, A_{8-10}, CE) are latched in at the trailing edge of ALE.
AD_{0-7}	I	**Bidirectional Address/Data Bus:** The lower 8-bits of the PROM or I/O address are applied to the bus lines when ALE is high. During an I/O cycle, Port A or B are selected based on the latched value of AD_0. If $\overline{RD}$ or $\overline{IOR}$ is low when the latched Chip Enables are active, the output buffers present data on the bus.
A_{8-10}	I	**Address:** These are the high order bits of the PROM address. They do not affect I/O operations.
PROG/$\overline{CE_1}$ CE_2	I	**Chip Enable Inputs:** $\overline{CE_1}$ is active low and CE_2 is active high. The 8755A can be accessed only when *both* Chip Enables are active at the time the ALE signal latches them up. If either Chip Enable input is not active, the AD_{0-7} and READY outputs will be in a high impedance state. $\overline{CE_1}$ is also used as a programming pin. (See section on programming.)
IO/$\overline{M}$	I	**I/O Memory:** If the latched IO/$\overline{M}$ is high when $\overline{RD}$ is low, the output data comes from an I/O port. If it is low the output data comes from the PROM.
$\overline{RD}$	I	**Read:** If the latched Chip Enables are active when $\overline{RD}$ goes low, the AD_{0-7} output buffers are enabled and output either the selected PROM location or I/O port. When both $\overline{RD}$ and $\overline{IOR}$ are high, the AD_{0-7} output buffers are 3-stated.
$\overline{IOW}$	I	**I/O Write:** If the latched Chip Enables are active, a low on $\overline{IOW}$ causes the output port pointed to by the latched value of AD_0 to be written with the data on AD_{0-7}. The state of IO/$\overline{M}$ is ignored.
CLK	I	**Clock:** The CLK is used to force the READY into its high impedance state after it has been forced low by $\overline{CE_1}$ low, CE_2 high, and ALE high.
READY	O	**Ready** is a 3-state output controlled by CE_2, $\overline{CE_1}$, ALE and CLK. READY is forced low when the Chip Enables are active during the time ALE is high, and remains low until the rising edge of the next CLK. (See Figure 6.)
PA_{0-7}	I/O	**Port A:** These are general purpose I/O pins. Their input/output direction is determined by the contents of Data Direction Register (DDR). Port A is selected for write operations when the Chip Enables are active and $\overline{IOW}$ is low and a 0 was previously latched from AD_0, AD_1. Read Operation is selected by either $\overline{IOR}$ low and active Chip Enables and AD_0 and AD_1 low, *or* IO/$\overline{M}$ high, $\overline{RD}$ low, active Chip Enables, and AD_0 and AD_1 low.
PB_{0-7}	I/O	**Port B:** This general purpose I/O port is identical to Port A except that it is selected by a 1 latched from AD_0 and a 0 from AD_1.
RESET	I	**Reset:** In normal operation, an input high on RESET causes all pins in Ports A and B to assume input mode (clear DDR register).
$\overline{IOR}$	I	**I/O Read:** When the Chip Enables are active, a low on $\overline{IOR}$ will output the selected I/O port onto the AD bus. $\overline{IOR}$ low performs the same function as the combination of IO/$\overline{M}$ high and $\overline{RD}$ low. When $\overline{IOR}$ is not used in a system, $\overline{IOR}$ should be tied to V_{CC} ("1").
V_{CC}		**Power:** +5 volt supply.
V_{SS}		**Ground:** Reference.
V_{DD}		**Power Supply:** V_{DD} is a programming voltage, and must be tied to V_{CC} when the 8755A is being read. For programming, a high voltage is supplied with V_{DD} = 25V, typical. (See section on programming.)

AFN-01941B

FUNCTIONAL DESCRIPTION

PROM Section

The 8755A contains an 8-bit address latch which allows it to interface directly to MCS-48, MCS-85 and iAPX 88/10 Microcomputers without additional hardware.

The PROM section of the chip is addressed by the 11-bit address and CE. The address, $\overline{CE_1}$ and CE_2 are latched into the address latches on the falling edge of ALE. If the latched Chip Enables are active and IO/$\overline{M}$ is low when $\overline{RD}$ goes low, the contents of the PROM location addressed by the latched address are put out on the AD_{0-7} lines (provided that V_{DD} is tied to V_{CC}).

I/O Section

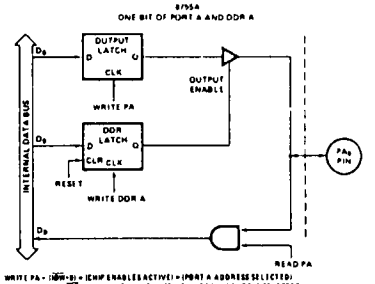
The I/O section of the chip is addressed by the latched value of AD_{0-1}. Two 8-bit Data Direction Registers (DDR) in 8755A determine the input/output status of each pin in the corresponding ports. A "0" in a particular bit position of a DDR signifies that the corresponding I/O port bit is in the input mode. A "1" in a particular bit position signifies that the corresponding I/O port bit is in the output mode. In this manner the I/O ports of the 8755A are bit-by-bit programmable as inputs or outputs. The table summarizes port and DDR designation. DDR's cannot be read.

AD_1	AD_0	Selection
0	0	Port A
0	1	Port B
1	0	Port A Data Direction Register (DDR A)
1	1	Port B Data Direction Register (DDR B)

When $\overline{IOW}$ goes low and the Chip Enables are active, the data on the AD is written into I/O port selected by the latched value of AD_{0-1}. During this operation all I/O bits of the selected port are affected, regardless of their I/O mode and the state of IO/$\overline{M}$. The actual output level does not change until $\overline{IOW}$ returns high. (glitch free output)

A port can be read out when the latched Chip Enables are active and either $\overline{RD}$ goes low with IO/$\overline{M}$ high, or $\overline{IOR}$ goes low. Both input and output mode bits of a selected port will appear on lines AD_{0-7}.

To clarify the function of the I/O Ports and Data Direction Registers, the following diagram shows the configuration of one bit of PORT A and DDR A. The same logic applies to PORT B and DDR B.



NOTE: WRITE PA IS NOT QUALIFIED BY IO/$\overline{M}$.

Note that hardware RESET or writing a zero to the DDR latch will cause the output latch's output buffer to be disabled, preventing the data in the Output Latch from being passed through to the pin. This is equivalent to putting the port in the input mode. Note also that the data can be written to the Output Latch even though the Output Buffer has been disabled. This enables a port to be initialized with a value prior to enabling the output.

The diagram also shows that the contents of PORT A and PORT B can be read even when the ports are configured as outputs.

TABLE 1. 8755A PROGRAMMING MODULE CROSS REFERENCE

MODULE NAME	USE WITH
UPP 955	UPP(4)
UPP UP2(2)	UPP 855
PROMPT 975	PROMPT 80/85(3)
PROMPT 475	PROMPT 48(1)

NOTES:
1. Described on p. 13-34 of 1978 Data Catalog.
2. Special adaptor socket.
3. Described on p. 13-39 of 1978 Data Catalog.
4. Described on p. 13-71 of 1978 Data Catalog.

ERASURE CHARACTERISTICS

The erasure characteristics of the 8755A are such that erasure begins to occur when exposed to light with wavelengths shorter than approximately 4000 Angstroms (Å). It should be noted that sunlight and certain types of fluorescent lamps have wavelengths in the 3000-4000Å range. Data show that constant exposure to room level fluorescent lighting could erase the typical 8755A in approximately 3 years while it would take approximately 1 week to cause erasure when exposed to direct sunlight. If the 8755A is to be exposed to these types of lighting conditions for extended periods of time, opaque labels are available from Intel which should be placed over the 8755 window to prevent unintentional erasure.

The recommended erasure procedure for the 8755A is exposure to shortwave ultraviolet light which has a wavelength of 2537 Angstroms (Å). The integrated dose (i.e., UV intensity X exposure time) for erasure should be a minimum of 15W-sec/cm². The erasure time with this dosage is approximately 15 to 20 minutes using an ultraviolet lamp with a 12000μW/cm² power rating. The 8755A should be placed within one inch from the lamp tubes during erasure. Some lamps have a filter on their tubes and this filter should be removed before erasure.

PROGRAMMING

Initially, and after each erasure, all bits of the EPROM portions of the 8755A are in the "1" state. Information is introduced by selectively programming "0" into the desired bit locations. A programmed "0" can only be changed to a "1" by UV erasure.

The 8755A can be programmed on the Intel® Universal PROM Programmer (UPP), and the PROMPT™ 80/85 and PROMPT-48™ design aids. The appropriate programming modules and adapters for use in programming both 8755A's and 8755's are shown in Table 1.

The program mode itself consists of programming a single address at a time, giving a single 50 msec pulse for every address. Generally, it is desirable to have a verify cycle after a program cycle for the same address as shown in the attached timing diagram. In the verify cycle (i.e., normal memory read cycle) V_{DD} should be at +5V.

Preliminary timing diagrams and parameter values pertaining to the 8755A programming operation are contained in Figure 7.

SYSTEM APPLICATIONS

System Interface with 8085A and 8088

A system using the 8755A can use either one of the two I/O interface techniques

- Standard I/O
- Memory Mapped I/O

If a standard I/O technique is used, the system can use the feature of both CE_3 and $\overline{CE_1}$. By using a combination of unused address lines A_{11-15} and the Chip Enable inputs, the 8085A system can use up to 5 each 8755A's without requiring a CE decoder. See Figure 2a and 2b.

If a memory mapped I/O approach is used the 8755A will be selected by the combination of both the Chip Enables and IO/$\overline{M}$ using the AD_{8-15} address lines. See Figure 1.

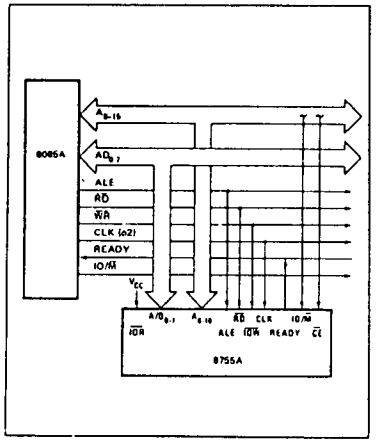


Figure 3. 8755A In 8085A System (Memory-Mapped I/O)

ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias	0°C to +70°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin With Respect to Ground	-0.5V to +7V
Power Dissipation	1.5W

NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

D.C. CHARACTERISTICS

(T_A = 0°C to 70°, V_{CC} = V_{DD} = 5V ± 5%; V_{CC} = V_{DD} = 5V ±10% for 8755A-2)

SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
V_{IL}	Input Low Voltage	-0.5	0.8	V	V_{CC} = 5.0V
V_{IH}	Input High Voltage	2.0	V_{CC}+0.5	V	V_{CC} = 5.0V
V_{OL}	Output Low Voltage		0.45	V	I_{OL} = 2mA
V_{OH}	Output High Voltage	2.4		V	I_{OH} = -400μA
I_{IL}	Input Leakage		10	μA	$V_{SS} \leq V_{IN} \leq V_{CC}$
I_{LO}	Output Leakage Current		±10	μA	$V_{SS} \leq 0.45V \leq V_{OUT} \leq V_{CC}$
I_{CC}	V_{CC} Supply Current		180	mA	
I_{DD}	V_{DD} Supply Current		30	mA	V_{DD} = V_{CC}
C_{IN}	Capacitance of Input Buffer		10	pF	f_C = 1μHz
$C_{I/O}$	Capacitance of I/O Buffer		15	pF	f_C = 1μHz

D.C. CHARACTERISTICS — PROGRAMMING

(T_A = 0°C to 70°, V_{CC} = 5V ± 5%, V_{SS} = 0V, V_{DD} = 25V ±1V; V_{CC} = V_{DD} = 5V ±10% for 8755A-2)

Symbol	Parameter	Min.	Typ.	Max.	Unit
V_{DD}	Programming Voltage (during Write to EPROM)	24	25	26	V
I_{DD}	Prog Supply Current		15	30	mA

A.C. CHARACTERISTICS

(T_A = 0°C to 70°, V_{CC} = 5V ± 5%, V_{CC} = V_{DD} = 5V ±10% for 8755A-2)

Symbol	Parameter	8755A		8755A-2 (Preliminary)		Units
		Min.	Max.	Min.	Max.	
t_{CYC}	Clock Cycle Time	320		200		ns
T_1	CLK Pulse Width	80		40		ns
T_2	CLK Pulse Width	120		70		ns
t_{r},t_{f}	CLK Rise and Fall Time		30		30	ns
t_{AL}	Address to Latch Set Up Time	50		30		ns
t_{LA}	Address Hold Time after Latch	80		45		ns
t_{LC}	Latch to READ/WRITE Control	100		40		ns
t_{RD}	Valid Data Out Delay from READ Control		170*		140*	ns
t_{AD}	Address Stable to Data Out Valid		450		330	ns
t_{LL}	Latch Enable Width	100		70		ns
t_{RDF}	Data Bus Float after READ	0	100	0	85	ns
t_{CL}	READ/WRITE Control to Latch Enable	20		10		ns
t_{CC}	READ/WRITE Control Width	250		200		ns
t_{DW}	Data In to Write Set Up Time	150		150		ns
t_{WD}	Data In Hold Time After WRITE	30		10		ns
t_{WP}	WRITE to Port Output		400		300	ns
t_{PR}	Port Input Set Up Time	50		50		ns
t_{RP}	Port Input Hold Time to Control	50		50		ns
t_{RYH}	READY HOLD Time to Control	0	160	0	160	ns
t_{ARY}	ADDRESS (CE) to READY		160		160	ns
t_{RV}	Recovery Time Between Controls	300		200		ns
t_{RDE}	READ Control to Data Bus Enable	10		10		ns
t_{LD}	ALE to Data Out Valid		350		270	ns

NOTE:

C_{LOAD} = 150pF.

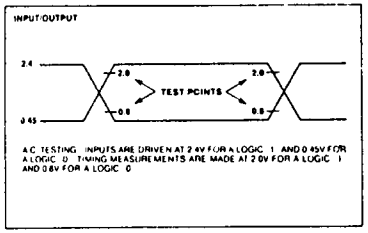
*Or T_{AD} = (T_{AL} + T_{LC}), whichever is greater.

A.C. CHARACTERISTICS — PROGRAMMING

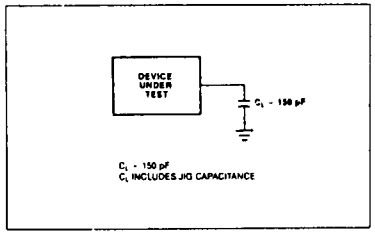
(T_A = 0°C to 70°, V_{CC} = 5V ± 5%, V_{SS} = 0V, V_{DD} = 25V ± 1V; V_{CC} = V_{DD} = 5V ±10% for 8755A-2)

Symbol	Parameter	Min.	Typ.	Max.	Unit
t_{PS}	Data Setup Time	10			ns
t_{PD}	Data Hold Time	0			ns
t_S	Prog Pulse Setup Time	2			μs
t_H	Prog Pulse Hold Time	2			μs
t_{PR}	Prog Pulse Rise Time	0.01	2		μs
t_{PF}	Prog Pulse Fall Time	0.01	2		μs
t_{PRG}	Prog Pulse Width	45	50		msec

A.C. TESTING INPUT, OUTPUT WAVEFORM

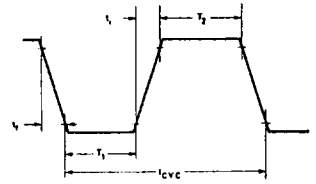


A.C. TESTING LOAD CIRCUIT

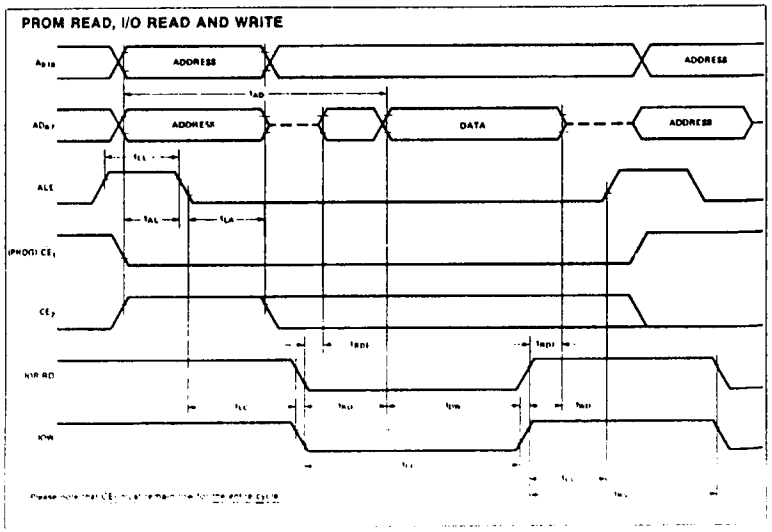


WAVEFORMS

CLOCK SPECIFICATION FOR 8755A

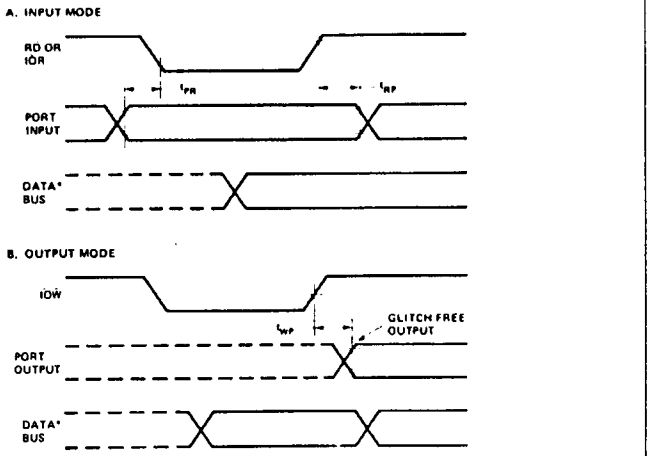


PROM READ, I/O READ AND WRITE

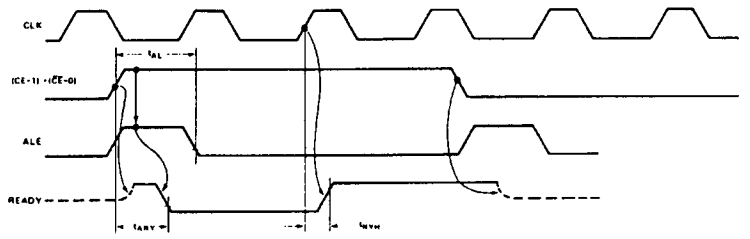


WAVEFORMS (Continued)

I/O PORT

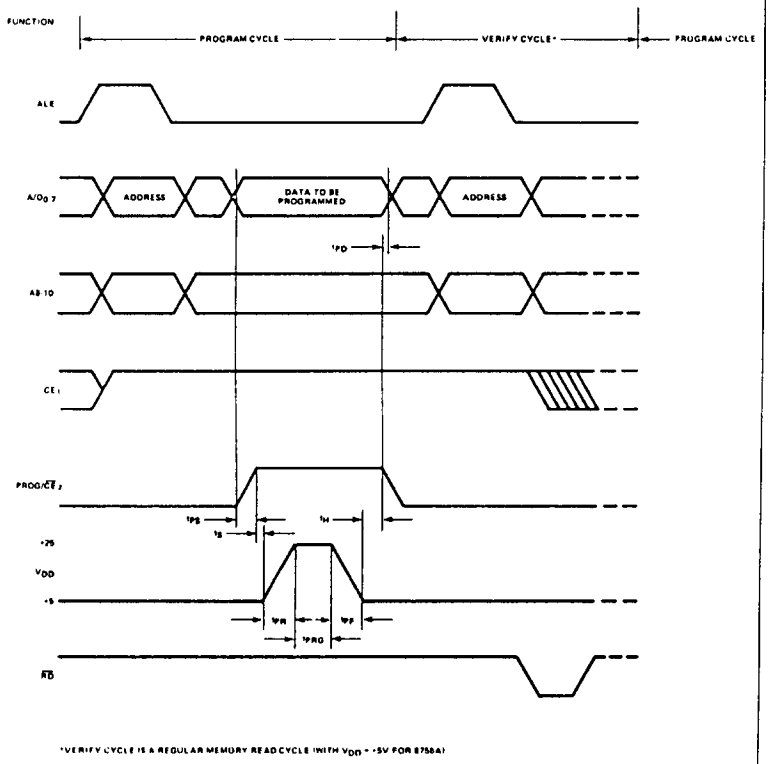


WAIT STATE (READY = 0)



WAVEFORMS (Continued)

8755A PROGRAM MODE



*VERIFY CYCLE IS A REGULAR MEMORY READ CYCLE (WITH V_{DD} = +5V FOR 8755A)

intel

8155/8156/8155-2/8156-2
2048 BIT STATIC MOS RAM WITH I/O PORTS AND TIMER

- 256 Word x 8 Bits
- Single +5V Power Supply
- Completely Static Operation
- Internal Address Latch
- 2 Programmable 8 Bit I/O Ports
- 1 Programmable 6-Bit I/O Port
- Programmable 14-Bit Binary Counter/ Timer
- Compatible with 8085A and 8088 CPU
- Multiplexed Address and Data Bus
- 40 Pin DIP

The 8155 and 8156 are RAM and I/O chips to be used in the 8085A and 8088 microprocessor systems. The RAM portion is designed with 2048 static cells organized as 256 x 8. They have a maximum access time of 400 ns to permit use with no wait states in 8085A CPU. The 8155-2 and 8156-2 have maximum access times of 330 ns for use with the 8085A-2 and the full speed 5 MHz 8088 CPU.

The I/O portion consists of three general purpose I/O ports. One of the three ports can be programmed to be status pins, thus allowing the other two ports to operate in handshake mode.

A 14-bit programmable counter/timer is also included on chip to provide either a square wave or terminal count pulse for the CPU system depending on timer mode.

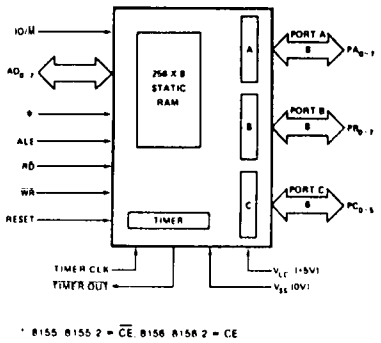


Figure 1. Block Diagram

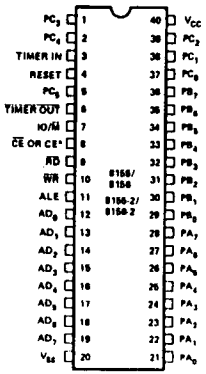


Figure 2. Pin Configuration

Table 1. Pin Description

Symbol	Type	Name and Function
RESET	I	**Reset:** Pulse provided by the 8085A to initialize the system (connect to 8085A RESET OUT). Input high on this line resets the chip and initializes the three I/O ports to input mode. The width of RESET pulse should typically be two 8085A clock cycle times.
AD_{0-7}	I/O	**Address/Data:** 3-state Address/Data lines that interface with the CPU lower 8-bit Address/Data Bus. The 8-bit address is latched into the address latch inside the 8155/86 on the falling edge of ALE. The address can be either for the memory section or the I/O section depending on the IO/$\overline{M}$ input. The 8-bit data is either written into the chip or read from the chip, depending on the $\overline{WR}$ or $\overline{RD}$ input signal.
CE or $\overline{CE}$	I	**Chip Enable:** On the 8155, this pin is $\overline{CE}$ and is ACTIVE LOW. On the 8156, this pin is CE and is ACTIVE HIGH.
$\overline{RD}$	I	**Read Control:** Input low on this line with the Chip Enable active enables and AD_{0-7} buffers. If IO/$\overline{M}$ pin is low, the RAM content will be read out to the AD bus. Otherwise the content of the selected I/O port or command/status registers will be read to the AD bus.
$\overline{WR}$	I	**Write Control:** Input low on this line with the Chip Enable active causes the data on the Address/Data bus to be written to the RAM or I/O ports and command/status register, depending on IO/$\overline{M}$.
ALE	I	**Address Latch Enable:** This control signal latches both the address on the AD_{0-7} lines and the state of the Chip Enable and IO/$\overline{M}$ into the chip at the falling edge of ALE.
IO/$\overline{M}$	I	**I/O Memory:** Selects memory if low and I/O and command/status registers if high.
PA_{0-7}(8)	I/O	**Port A:** These 8 pins are general purpose I/O pins. The in/out direction is selected by programming the command register.
PB_{0-7}(8)	I/O	**Port B:** These 8 pins are general purpose I/O pins. The in/out direction is selected by programming the command register.
PC_{0-5}(6)	I/O	**Port C:** These 6 pins can function as either input port, output port, or as control signals for PA and PB. Programming is done through the command register. When PC_{0-5} are used as control signals, they will provide the following: PC_0 — A INTR (Port A Interrupt) PC_1 — ABF (Port A Buffer Full) PC_2 — $\overline{A\ STB}$ (Port A Strobe) PC_3 — B INTR (Port B Interrupt) PC_4 — $\overline{B}$ BF (Port B Buffer Full) PC_5 — $\overline{B\ STB}$ (Port B Strobe)
TIMER IN	I	**Timer Input:** Input to the counter-timer.
$\overline{TIMER\ OUT}$	O	**Timer Output:** This output can be either a square wave or a pulse, depending on the timer mode.
V_{CC}		**Voltage:** +5 volt supply.
V_{SS}		**Ground:** Ground reference.

FUNCTIONAL DESCRIPTION

The 8155/8156 contains the following:

- 2k Bit Static RAM organized as 256 x 8
- Two 8-bit I/O ports (PA & PB) and one 6-bit I/O port (PC)
- 14-bit timer-counter

The IO/$\overline{M}$ (IO/Memory Select) pin selects either the five registers (Command, Status, PA_{0-7}, PB_{0-7}, PC_{0-5}) or the memory (RAM) portion.

The 8-bit address on the Address/Data lines, Chip Enable input CE or $\overline{CE}$, and IO/$\overline{M}$ are all latched on-chip at the falling edge of ALE.

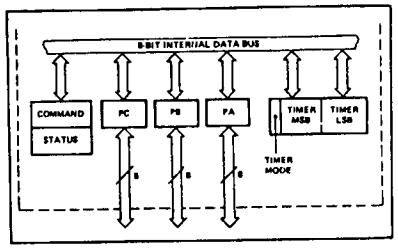


Figure 3. 8155/8156 Internal Registers

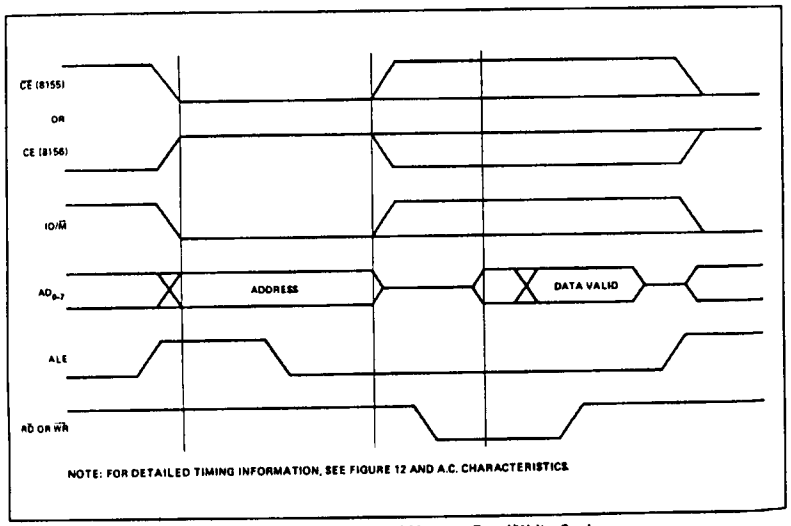


Figure 4. 8155/8156 On-Board Memory Read/Write Cycle

PROGRAMMING OF THE COMMAND REGISTER

The command register consists of eight latches. Four bits (0-3) define the mode of the ports, two bits (4-5) enable or disable the interrupt from port C when it acts as control port, and the last two bits (6-7) are for the timer.

The command register contents can be altered at any time by using the I/O address XXXXX000 during a WRITE operation with the Chip Enable active and IO/$\overline{M}$ = 1. The meaning of each bit of the command byte is defined in Figure 5. The contents of the command register may never be read.

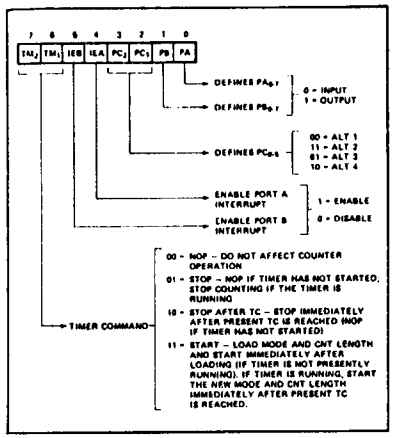


Figure 5. Command Register Bit Assignment

READING THE STATUS REGISTER

The status register consists of seven latches, one for each bit, six (0-5) for the status of the ports and one (6) for the status of the timer.

The status of the timer and the I/O section can be polled by reading the Status Register (Address XXXXX000). Status word format is shown in Figure 6. Note that you may never write to the status register since the command register shares the same I/O address and the command register is selected when a write to that address is issued.

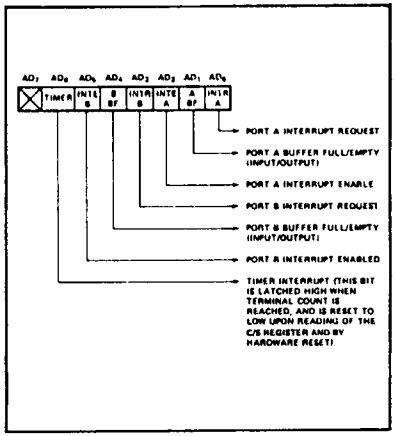


Figure 6. Status Register Bit Assignment

INPUT/OUTPUT SECTION

The I/O section of the 8155/8156 consists of five registers: (See Figure 7.)

- **Command/Status Register (C/S)** — Both registers are assigned the address XXXXX000. The C/S address serves the dual purpose.

  When the C/S registers are selected during WRITE operation, a command is written into the command register. The contents of this register are *not* accessible through the pins.

  When the C/S (XXXXX000) is selected during a READ operation, the status information of the I/O ports and the timer becomes available on the AD_{0-7} lines.

- **PA Register** — This register can be programmed to be either input or output ports depending on the status of the contents of the C/S Register. Also depending on the command, this port can operate in either the basic mode or the strobed mode (See timing diagram). The I/O pins assigned in relation to this register are PA_{0-7}. The address of this register is XXXXX001.

- **PB Register** — This register functions the same as PA Register. The I/O pins assigned are PB_{0-7}. The address of this register is XXXXX010.

- **PC Register** — This register has the address XXXXX011 and contains only 6 bits. The 6 bits can be programmed to be either input ports, output ports or as control signals for PA and PB by properly programming the AD_2 and AD_3 bits of the C/S register.

  When PC_{0-5} is used as a control port, 3 bits are assigned for Port A and 3 for Port B. The first bit is an interrupt that the 8155 sends out. The second is an output signal indicating whether the buffer is full or empty, and the third is an input pin to accept a strobe for the strobed input mode. (See Table 2.)

When the 'C' port is programmed to either ALT3 or ALT4, the control signals for PA and PB are initialized as follows:

CONTROL	INPUT MODE	OUTPUT MODE
BF	Low	Low
INTR	Low	High
STB	Input Control	Input Control

I/O ADDRESS[1]								SELECTION
A7	A6	A5	A4	A3	A2	A1	A0	
X	X	X	X	X	0	0	0	Interval Command/Status Register
X	X	X	X	X	0	0	1	General Purpose I/O Port A
X	X	X	X	X	0	1	0	General Purpose I/O Port B
X	X	X	X	X	0	1	1	Port C — General Purpose I/O or Control
X	X	X	X	X	1	0	0	Low-Order 8 bits of Timer Count
X	X	X	X	X	1	0	1	High 6 bits of Timer Count and 2 bits of Timer Mode

X: Don't Care.
1. I/O Address must be qualified by $\overline{CE}$ = 1 (8156) or $\overline{CE}$ = 0 (8155) and IO/$\overline{M}$ = 1 in order to select the appropriate register.

Figure 7. I/O Port and Timer Addressing Scheme

Figure 8 shows how I/O PORTS A and B are structured within the 8155 and 8156:

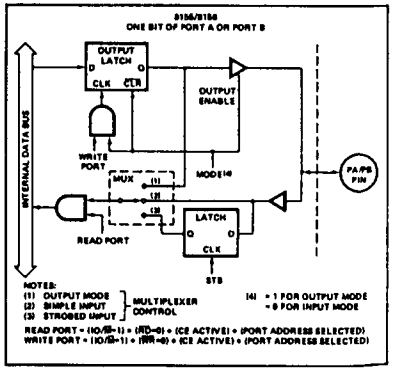


Figure 8. 8155/8156 Port Functions

Table 2. Port Control Assignment

Pin	ALT 1	ALT 2	ALT 3	ALT 4
PC0	Input Port	Output Port	A INTR (Port A Interrupt)	A INTR (Port A Interrupt)
PC1	Input Port	Output Port	A BF (Port A Buffer Full	A BF (Port A Buffer Full)
PC2	Input Port	Output Port	A $\overline{STB}$ (Port A Strobe)	A $\overline{STB}$ (Port A Strobe)
PC3	Input Port	Output Port	Output Port	B INTR (Port B Interrupt)
PC4	Input Port	Output Port	Output Port	B BF (Port B Buffer Full)
PC5	Input Port	Output Port	Output Port	B $\overline{STB}$ (Port B Strobe)

Note in the diagram that when the I/O ports are programmed to be output ports, the contents of the output ports can still be read by a READ operation when appropriately addressed.

The outputs of the 8155/8156 are "glitch-free" meaning that you can write a "1" to a bit position that was previously "1" and the level at the output pin will not change.

Note also that the output latch is cleared when the port enters the input mode. The output latch cannot be loaded by writing to the port if the port is in the input mode. The result is that each time a port mode is changed from input to output, the output pins will go low. When the 8155/56 is RESET, the output latches are all cleared and all 3 ports enter the input mode.

When in the ALT 1 or ALT 2 modes, the bits of PORT C are structured like the diagram above in the simple input or output mode, respectively.

Reading from an input port with nothing connected to the pins will provide unpredictable results.

Figure 9 shows how the 8155/8156 I/O ports might be configured in a typical MCS-85 system.

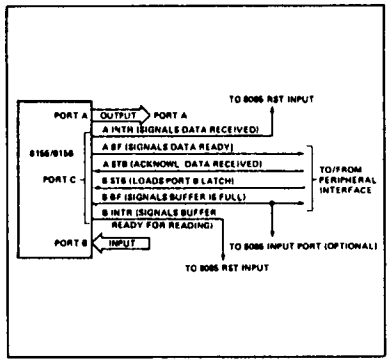


Figure 9. Example: Command Register = 00111001

TIMER SECTION

The timer is a 14-bit down-counter that counts the TIMER IN pulses and provides either a square wave or pulse when terminal count (TC) is reached.

The timer has the I/O address XXXXX100 for the low order byte of the register and the I/O address XXXXX101 for the high order byte of the register. (See Figure 7.)

To program the timer, the COUNT LENGTH REG is loaded first, one byte at a time, by selecting the timer addresses. Bits 0-13 of the high order count register will specify the length of the next count and bits 14-15 of the high order register will specify the timer output mode (see Figure 10). The value loaded into the count length register can have any value from 2H through 3FFH in Bits 0–13.

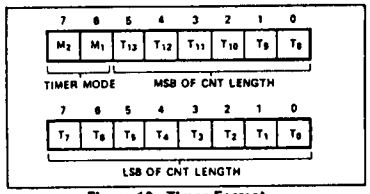


Figure 10. Timer Format

There are four modes to choose from: M2 and M1 define the timer mode, as shown in Figure 11.

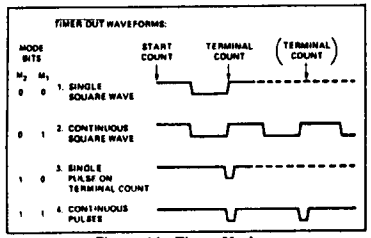


Figure 11. Timer Modes

Bits 6-7 (TM_2 and TM_1) of command register contents are used to start and stop the counter. There are four commands to choose from:

TM_2	TM_1	
0	0	NOP — Do not affect counter operation
0	1	STOP — NOP if timer has not started, stop counting if the timer is running
1	0	STOP AFTER TC — Stop immediately after present TC is reached. NOP if timer has not started
1	1	START — Load mode and CNT length and start immediately after loading, if timer is not presently running. If timer is running, start the new mode and CNT length immediately after present TC is reached.

Note that while the counter is counting, you may load a new count and mode into the count length registers. Before the new count and mode will be used by the counter, you must issue a START command to the counter. This applies even though you may only want to change the count and use the previous mode.

In case of an odd-numbered count, the first half-cycle of the squarewave output, which is high, is one count longer than the second (low) half-cycle, as shown in Figure 12.

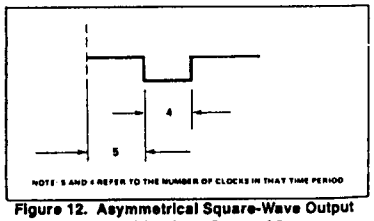


Figure 12. Asymmetrical Square-Wave Output Resulting from Count of 9

The counter in the 8155 is not initialized to any particular mode or count when hardware RESET occurs, but RESET does *stop* the counting. Therefore, counting cannot begin following RESET until a START command is issued via the C/S register.

Please note that the timer circuit on the 8155/8156 chip is designed to be a square-wave timer, not an event counter. To achieve this, it counts down by twos twice in completing one cycle. Thus, its registers do not contain values directly representing the number of TIMER IN pulses received. You cannot load an initial value of 1 into the count register and cause the timer to operate, as its terminal count value is 10 (binary) or 2 (decimal). (For the detection of single pulses, it is suggested that one of the hardware interrupt pins on the 8085A be used.) After the timer has started counting down, the values residing in the count registers can be used to calculate the actual number of TIMER IN pulses required to complete the timer cycle if desired. To obtain the remaining count, perform the following operations in order:

1. Stop the count
2. Read in the 16-bit value from the count length registers
3. Reset the upper two mode bits
4. Reset the carry and rotate right one position all 16 bits through carry
5. If carry is set, add 1/2 of the full original count (1/2 full count — 1 if full count is odd).

Note: If you started with an odd count and you read the count length register before the third count pulse occurs, you will not be able to discern whether one or two counts has occurred. Regardless of this, the 8155/56 always counts out the right number of pulses in generating the TIMER OUT waveforms.

ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias	0°C to +70°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin With Respect to Ground	-0.5V to +7V
Power Dissipation	1.5W

**NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

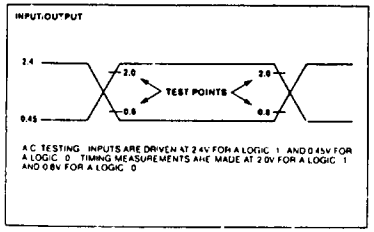
D.C. CHARACTERISTICS (T_A = 0°C to 70°C; V_{CC} = 5V ± 5%)

SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
V_{IL}	Input Low Voltage	-0.5	0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC}+0.5	V	
V_{OL}	Output Low Voltage		0.45	V	I_{OL} = 2mA
V_{OH}	Output High Voltage	2.4		V	I_{OH} = -400μA
I_{IL}	Input Leakage		±10	μA	$0V \leq V_{IN} \leq V_{CC}$
I_{LO}	Output Leakage Current		± 10	μA	$0.45V \leq V_{OUT} \leq V_{CC}$
I_{CC}	V_{CC} Supply Current		180	mA	
I_{IL}(CE)	Chip Enable Leakage 8155 8156		 +100 -100	 μA μA	$0V \leq V_{IN} \leq V_{CC}$

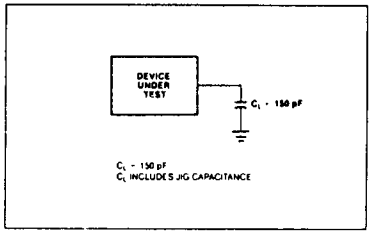
A.C. CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = 5V ± 5%)

		8155/8156		8155-2/8156-2		
SYMBOL	PARAMETER	MIN.	MAX.	MIN.	MAX.	UNITS
t_{AL}	Address to Latch Set Up Time	50		30		ns
t_{LA}	Address Hold Time after Latch	80		30		ns
t_{LC}	Latch to READ/WRITE Control	100		40		ns
t_{RD}	Valid Data Out Delay from READ Control		170		140	ns
t_{AD}	Address Stable to Data Out Valid		400		330	ns
t_{LL}	Latch Enable Width	100		70		ns
t_{RDF}	Data Bus Float After READ	0	100	0	80	ns
t_{CL}	READ/WRITE Control to Latch Enable	20		10		ns
t_{CC}	READ/WRITE Control Width	250		200		ns
t_{DW}	Data In to WRITE Set Up Time	150		100		ns
t_{WD}	Data In Hold Time After WRITE	0		0		ns
t_{RV}	Recovery Time Between Controls	300		200		ns
t_{WP}	WRITE to Port Output		400		300	ns
t_{PR}	Port Input Setup Time	70		50		ns
t_{RP}	Port Input Hold Time	50		10		ns
t_{SBF}	Strobe to Buffer Full		400		300	ns
t_{SS}	Strobe Width	200		150		ns
t_{RBE}	READ to Buffer Empty		400		300	ns
t_{SI}	Strobe to INTR On		400		300	ns
t_{RDI}	READ to INTR Off		400		300	ns
t_{PSS}	Port Setup Time to Strobe Strobe	50		0		ns
t_{PHS}	Port Hold Time After Strobe	120		100		ns
t_{SBE}	Strobe to Buffer Empty		400		300	ns
t_{WBF}	WRITE to Buffer Full		400		300	ns
t_{WI}	WRITE to INTR Off		400		300	ns
t_{TL}	TIMER-IN to $\overline{\text{TIMER-OUT}}$ Low		400		300	ns
t_{TH}	TIMER-IN to $\overline{\text{TIMER-OUT}}$ High		400		300	ns
t_{RDE}	Data Bus Enable from READ Control	10		10		ns
t_1	TIMER-IN Low Time	80		40		ns
t_2	TIMER-IN High Time	120		70		ns

A.C. TESTING INPUT, OUTPUT WAVEFORM



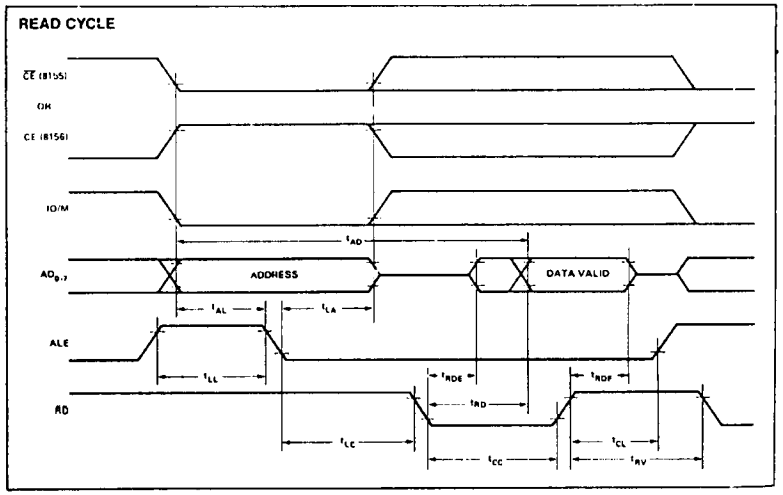
A.C. TESTING LOAD CIRCUIT



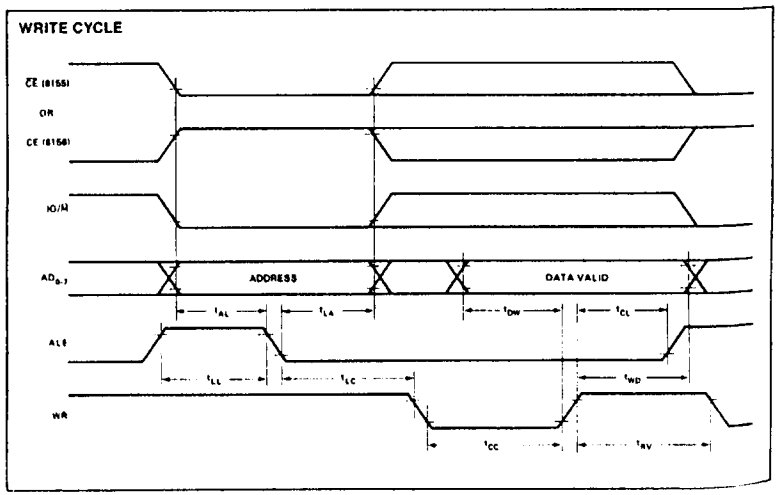
AFN-00201C

WAVEFORMS

READ CYCLE



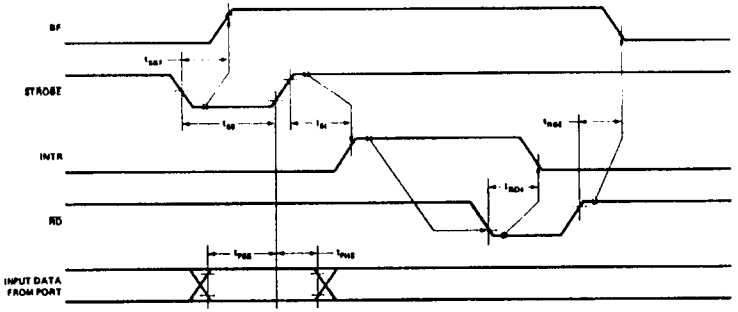
WRITE CYCLE



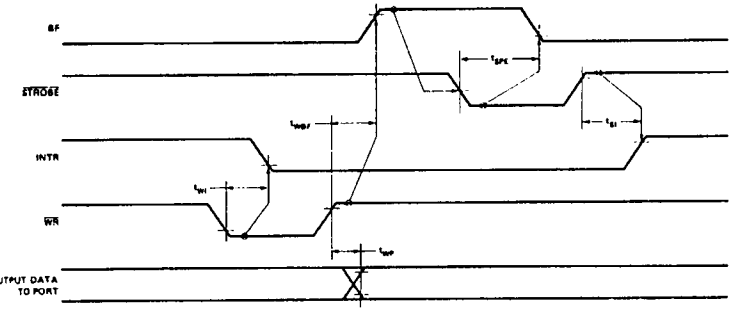
AFN-00201C

WAVEFORMS (Continued)

STROBED INPUT MODE

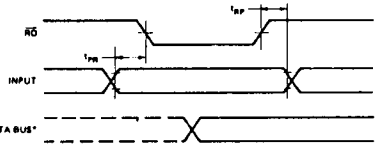


STROBED OUTPUT MODE

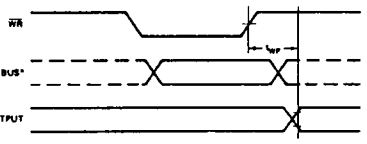


WAVEFORMS (Continued)

BASIC INPUT MODE

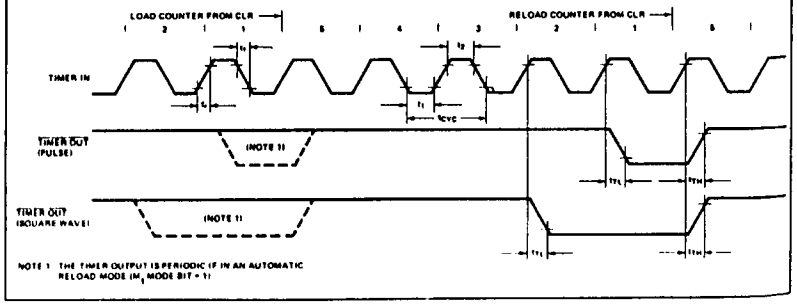


BASIC OUTPUT MODE



*DATA BUS TIMING IS SHOWN IN FIGURE 7.

TIMER OUTPUT COUNTDOWN FROM 5 TO 1



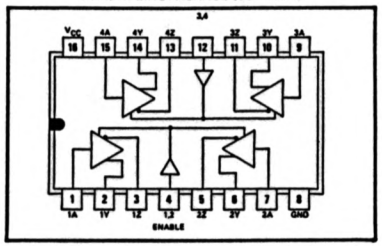
NOTE 1: THE TIMER OUTPUT IS PERIODIC IF IN AN AUTOMATIC RELOAD MODE (M_2 MODE BIT = 1)

TYPE SN75174
QUADRUPLE DIFFERENTIAL LINE DRIVER

BULLETIN NO. DL-S 12771, OCTOBER 1980

- Meets EIA Standard RS-422A and CCITT Recommendations V.11 and X.27
- Meets EIA Subcommittee TR30.1 Draft Standard PN1360 (as of April 1980)
- Designed for Multipoint Transmission on Long Bus Lines in Noisy Environments
- 3-State Outputs
- Common-Mode Output Voltage Range ... −7 V to 12 V
- Active-High Enables
- Thermal Shutdown Protection
- Positive and Negative Current Limiting
- Operates from Single 5-Volt Supply
- Low Power Requirements
- Functionally Interchangeable with MC3487

J OR N
DUAL-IN-LINE PACKAGE (TOP VIEW)



FUNCTION TABLE (EACH DRIVER)

INPUT	ENABLE	OUTPUTS	
		Y	Z
H	H	H	L
L	H	L	H
X	L	Z	Z

H = TTL high level, L = TTL low level, X = irrelevant, Z = high impedance (off)

description

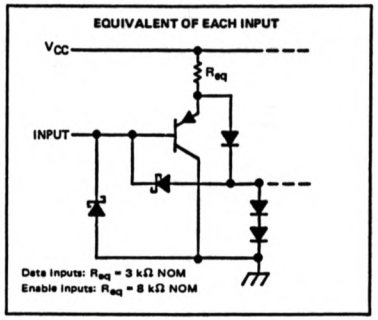
The SN75174 is a monolithic guadruple differential line driver with three-state outputs. It is designed to meet the requirements of EIA Standard RS-422A and CCITT Recommendations V.11 and X.27. The device is optimized for balanced multipoint bus transmission at rates up to 4 megabits per second. Each driver features wide positive and negative common-mode output voltage ranges making it suitable for party-line applications in noisy environments.

The SN75174 provides positive- and negative-current limiting and thermal shutdown for protection from line fault conditions on the transmission bus line. Shutdown occurs at a junction temperature of approximately 150°C. This device offers optimum performance when used with the SN75173 or SN75175 quadruple differential line receivers.

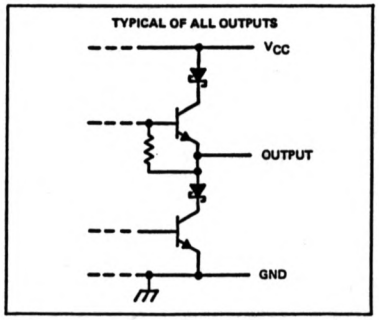
The SN75174 is characterized for operation from 0°C to 70°C.

schematics of inputs and outputs

EQUIVALENT OF EACH INPUT



TYPICAL OF ALL OUTPUTS



absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1) ..... 7 V
Input voltage ..... 5.5 V
Continuous total dissipation at (or below) 25°C free-air temperature (see Note 2): J package ..... 1375 mW
N package ..... 1150 mW
Operating free-air temperature range ..... 0°C to 70°C
Storage temperature range ..... −65°C to 150°C
Lead temperature 1/16 inch (1,6 mm) from case for 60 seconds: J package ..... 300°C
Lead temperature 1/16 inch (1,6 mm) from case for 10 seconds: N package ..... 260°C

NOTES: 1. All voltage values are with respect to the network ground terminal.
2. For operation above 25°C free-air temperature, derate the J package to 880 mW at 70°C at the rate of 11 mW/°C and the N package to 736 mW at 70°C at the rate of 9.2 mW/°C. In the J package, SN75174 chips are alloy-mounted.

recommended operating conditions

	MIN	NOM	MAX	UNIT
Supply voltage, V_{CC}	4.75	5	5.25	V
Common-mode output voltage, V_{OC}	−7†		12	V
High-level output current, I_{OH}			−60	mA
Low-level output current, I_{OL}			60	mA
Operating free-air temperature, T_A	0		70	°C

†The algebraic convention, where the less-positive (more-negative) limit is designated minimum, is used in this data sheet with common-mode output voltage only.

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature (unless other noted)

PARAMETER		TEST CONDITIONS		MIN	TYP‡	MAX	UNIT
V_{IH}	High-level input voltage			2			V
V_{IL}	Low-level input voltage					0.8	V
V_{IK}	Input clamp voltage	I_I = −18 mA				−1.5	V
V_{OH}	High-level output voltage	V_{IH} = 2 V, V_{IL} = 0.8 V, I_{OH} = −33 mA			3.7		V
V_{OL}	Low-level output voltage	V_{IH} = 2 V, V_{IL} = 0.8 V, I_{OL} = 33 mA			1.1		V
$\lvert V_{OD1}\rvert$	Differential output voltage	I_O = 0				$2V_{OD2}$	V
$\lvert V_{OD2}\rvert$	Differential output voltage	R_L = 100 Ω,	See Figure 1	2			V
		R_L = 60 Ω,	See Figure 1	1.5			
$\Delta\lvert V_{OD}\rvert$	Change in magnitude of differential output voltage§	R_L = 60 Ω or 100 Ω, See Figure 1				±0.2	V
V_{OC}	Common-mode output voltage¶					3	V
$\Delta\lvert V_{OC}\rvert$	Change in magnitude of common-mode output voltage§					±0.2	V
I_O	Output current with power off	V_{CC} = 0, V_O = −7 V to 12				±100	μA
I_{OZ}	High-impedance-state output current	V_O = −7 V to 12 V				±100	μA
I_{IH}	High-level input current	V_I = 2.7 V				20	μA
I_{IL}	Low-level input current	V_I = 0.5 V				−360	μA
I_{OS}	Short-circuit output current	V_O = −7 V				−180	mA
		V_O = V_{CC}				180	
		V_O = 12 V				500	
I_{CC}	Supply current (all drivers)	No load	Outputs enabled		38	60	mA
			Outputs disabled		18	40	

‡All typical values are at V_{CC} = 5 V and T_A = 25°C.
§$\Delta\lvert V_{OD}\rvert$ and $\Delta\lvert V_{OC}\rvert$ are the changes in magnitude of V_{OD} and V_{OC}, respectively, that occur when the input is changed from a high level to a low level.
¶In EIA Standard RS-422A, V_{OC}, which is the average of the two output voltages with respect to ground, is called output offset voltage, V_{OS}.

switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER		TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{DD}	Differential-output delay time	R_L = 60 Ω,	See Figure 2		35	50	ns
t_{TD}	Differential-output transition time				50	75	ns
t_{PLH}	Propagation delay time, low-to-high-level output	R_L = 27 Ω,	See Figure 3		16	25	ns
t_{PHL}	Propagation delay time, high-to-low-level output				44	65	ns
t_{PZH}	Output enable time to high level	R_L = 110 Ω,	See Figure 4		60	80	ns
t_{PZL}	Output enable time to low level	R_L = 110 Ω,	See Figure 5		30	45	ns
t_{PHZ}	Output disable time from high level	R_L = 110 Ω,	See Figure 4		51	75	ns
t_{PLZ}	Output disable time from low level	R_L = 110 Ω,	See Figure 5		18	30	ns

PARAMETER MEASUREMENT INFORMATION

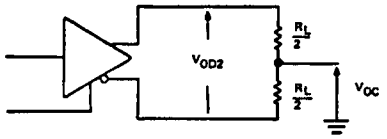
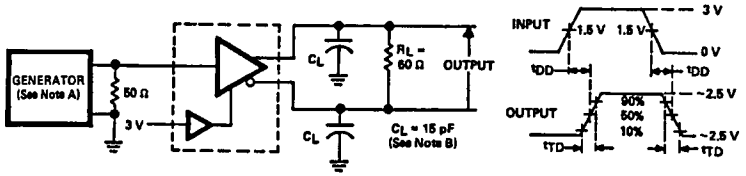


FIGURE 1–DIFFERENTIAL AND COMMON-MODE OUTPUT VOLTAGES



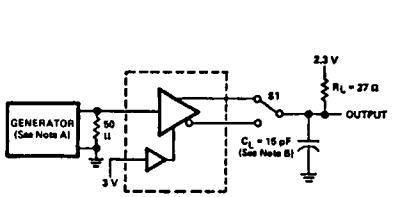
TEST CIRCUIT VOLTAGE WAVEFORMS

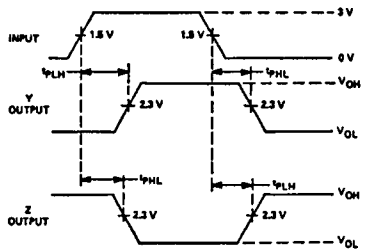
NOTES: A. The input pulse is supplied by a generator having the following characteristics: $t_r \leq 5$ ns, $t_f \leq 5$ ns, PRR = 1 MHz, duty cycle = 50%, Z_o = 50 Ω.
B. C_L includes probe and stray capacitance.
C. All diodes are IN916 or IN3064.

FIGURE 2–DIFFERENTIAL-OUTPUT DELAY AND TRANSITION TIMES

TYPE SN75174
QUADRUPLE DIFFERENTIAL LINE DRIVER

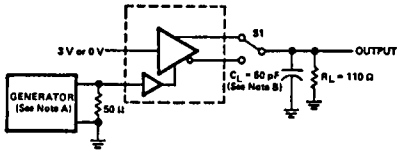
PARAMETER MEASUREMENT INFORMATION

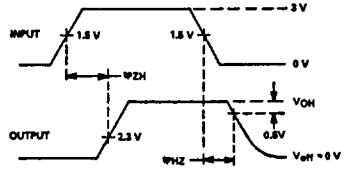




TEST CIRCUIT VOLTAGE WAVEFORMS

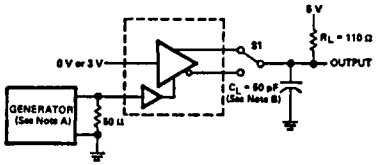
FIGURE 3–PROPAGATION DELAY TIMES

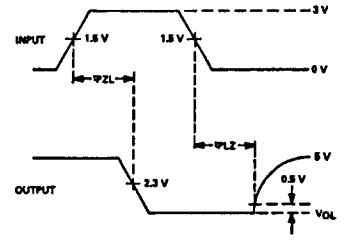




TEST CIRCUIT VOLTAGE WAVEFORMS

FIGURE 4–t_{PZH} AND t_{PHZ}





TEST CIRCUIT VOLTAGE WAVEFORMS

FIGURE 5–t_{PZL} AND t_{PLZ}

NOTES: A. The input pulse is supplied by a generator having the following characteristics: PRR = 1 MHz, duty cycle = 50%, $t_r \leq 5$ ns, $t_f \leq 5$ ns, Z_o = 50 Ω.
B. C_L includes probe and stray capacitance.

TYPICAL CHARACTERISTICS

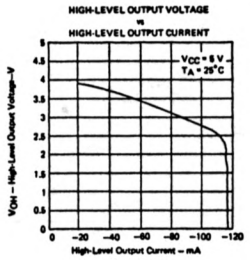


FIGURE 6

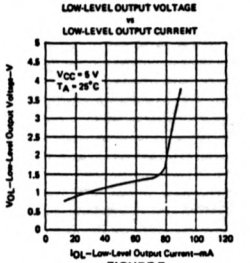


FIGURE 7

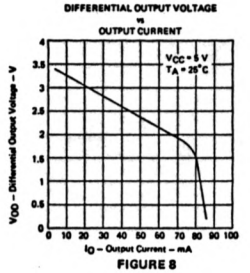


FIGURE 8

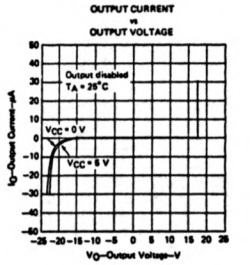


FIGURE 9

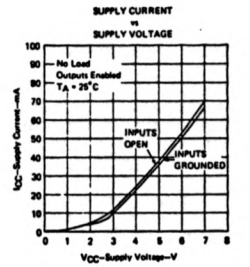


FIGURE 10

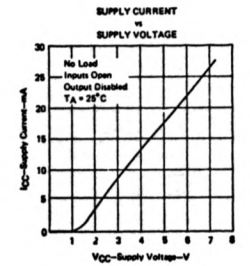
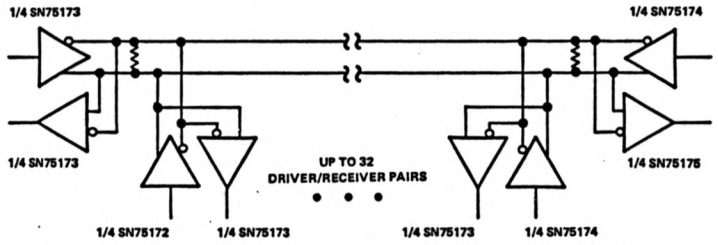


FIGURE 11

TYPICAL APPLICATION



NOTE: The line length should be terminated at both ends in its characteristic impedance. Stub lengths off the main line should be kept as short as possible.

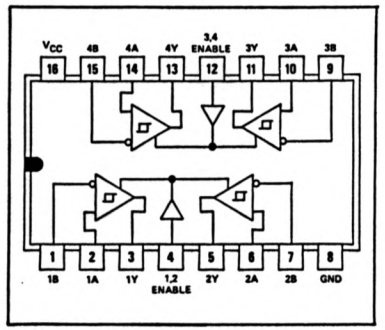
FIGURE 12

TYPE SN75175
QUADRUPLE DIFFERENTIAL LINE RECEIVER

BULLETIN NO. DL-S 12772, OCTOBER 1980

- Meets EIA Standards RS-422A and RS-423A
- Meets CCITT Recommendations V.10, V.11, X.26, and X.27
- Meets EIA Subcommittee TR30.1 Draft Standard PN1360 (as of April 1980)
- Designed for Multipoint Bus Transmission on Long Bus Lines in Noisy Environments
- 3-State Outputs
- Common-Mode Input Voltage Range . . . –12 V to 12 V
- Input Sensitivity . . . ±200 mV
- Input Hysteresis . . . 50 mV Typ
- High Input Impedance . . . 12 kΩ Min
- Operates from Single 5-Volt Supply
- Low Power Requirements
- Plug-In Replacement for MC3486

J OR N
DUAL-IN-LINE PACKAGE (TOP VIEW)



description

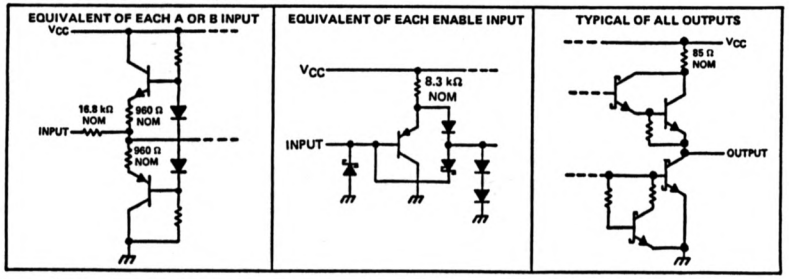
The SN75175 is a monolithic quadrupled differential line receiver with three-state outputs. It is designed to meet the requirements of EIA Standards RS-422A and RS-423A and several CCITT recommendations. The device is optimized for balanced multipoint bus transmission at rates up to 10 megabits per second. Each receiver features two active-high enables, each common to two receivers. It also features high input impedance, input hysteresis for increased noise immunity, and input sensitivity of ±200 millivolts over a common-mode input voltage range of –12 volts to 12 volts. The SN75175 is designed for optimum performance when used with the SN75172 or SN75174 quadruple differential line drivers.

FUNCTION TABLE (EACH RECEIVER)

DIFFERENTIAL INPUTS A – B	ENABLE	OUTPUT Y
V_{ID} >0.2 V	H	H
–0.2 V < V_{ID} < 0.2 V	H	?
V_{ID} < –0.2 V	H	L
X	L	Z

H = high level, L = low level, ? = indeterminate,
X = irrelevant, Z = high impedance (off)

schematics of inputs and outputs



absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1)	7 V
Input voltage, A or B inputs	±25 V
Differential input voltage (see Note 2)	±25 V
Enable input voltage	7 V
Low-level output current	50 mA
Continuous total dissipation at (or below) 25°C free-air temperature (see Note 3): J Package	1025 mW
N Package	1150 mW
Operating free-air temperature range	0°C to 70°C
Storage temperature range	−65°C to 150°C
Lead temperature 1/16 inch (1,6 mm) from case for 60 seconds: J Package	300°C
Lead temperature 1/16 inch (1,6 mm) from case for 10 seconds: N Package	260°C

NOTES: 1. All voltage values, except differential input voltage, are with respect to network ground terminal.
2. Differential-input voltage is measured at the noninverting input with respect to the corresponding inverting input.
3. For operation above 25°C free-air temperature, derate the J package to 656 mW at 70°C at a rate of 8.2 mW/°C and the N package to 736 mW at 70°C at a rate of 9.2 mW/°C. In the J package, SN75175 chips are glass-mounted.

recommended operating conditions

	MIN	NOM	MAX	UNIT
Supply voltage, V_{CC}	4.75	5	5.25	V
Common-mode input voltage, V_{IC}			±12	V
Differential input voltage, V_{ID}			±12	V
High-level output current, I_{OH}			−400	μA
Low-level output current, I_{OL}			16	mA
Operating free-air temperature, T_A	0		70	°C

electrical characteristics over recommended ranges of common-mode input voltage, supply voltage, and operating free-air temperature (unless otherwise noted)

PARAMETER		TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{TH}	Differential-input high-threshold voltage	V_O = 2.7 V, I_O = −0.4 mA				0.2	V
V_{TL}	Differential-input low-threshold voltage	V_O = 0.5 V, I_O = 16 mA		−0.2‡			V
$V_{T+} - V_{T-}$	Hysteresis§				50		mV
V_{IH}	High-level enable input voltage			2			V
V_{IL}	Low-level enable input voltage					0.8	V
V_{IK}	Enable-input clamp voltage	I_I = −18 mA				−1.5	V
V_{OH}	High-level output voltage	V_{ID} = 200 mV, I_{OH} = −400 μA, See Figure 1		2.7			V
V_{OL}	Low-level output voltage	V_{ID} = −200 mV, See Figure 1	I_{OL} = 8 mA			0.45	V
			I_{OL} = 16 mA			0.5	
I_{OZ}	High-impedance-state output current	V_O = 0.4 V to 2.4 V				±20	μA
I_I	Line input current	Other input at 0 V, See Note 4	V_I = 12 V			1	mA
			V_I = −7 V			−0.8	
I_{IH}	High-level enable-input current	V_{IH} = 2.7 V				20	μA
I_{IL}	Low-level enable-input current	V_{IL} = 0.4 V				−100	μA
r_I	Input resistance			12			kΩ
I_{OS}	Short-circuit output current¶			−15		−85	mA
I_{CC}	Supply current	Outputs disabled				70	mA

†All typical values are at V_{CC} = 5 V, T_A = 25°C
‡The algebraic convention, where the less-positive (more-negative) limit is designated as minimum, is used in this data sheet for threshold voltage levels only.
§Hysteresis is the difference between the positive-going input threshold voltage, V_{T+}, and the negative-going input threshold voltage, V_{T-}. See Figure 4.
¶Not more than one output should be shorted at a time and the duration of the short-circuit should not exceed one second.
NOTE 4: Refer to EIA standards RS-422-A and RS-423-A for exact conditions.

TYPE SN75175
QUADRUPLE DIFFERENTIAL LINE RECEIVER

switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER		TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{PLH}	Propagation delay time, low-to-high-level output	C_L = 15 pF,	See Figure 2		22	35	ns
t_{PHL}	Propagation delay time, high-to-low-level output				26	35	ns
t_{PZH}	Output enable time to high level	C_L = 15 pF,	See Figure 3		13	30	ns
t_{PZL}	Output enable time to low level				19	30	ns
t_{PHZ}	Output disable time from high level	C_L = 5 pF,	See Figure 3		26	35	ns
t_{PLZ}	Output disable time from low level				25	35	ns

PARAMETER MEASUREMENT INFORMATION

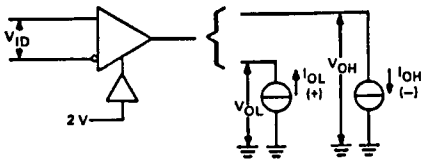


FIGURE 1–V_{OH}, V_{OL}

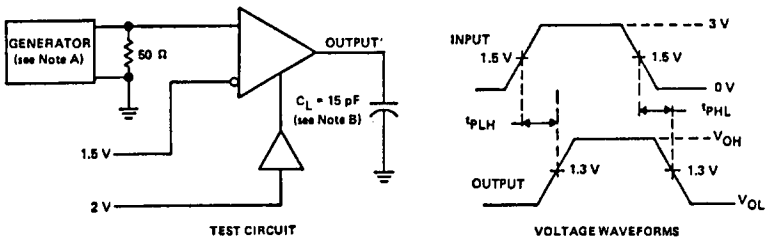
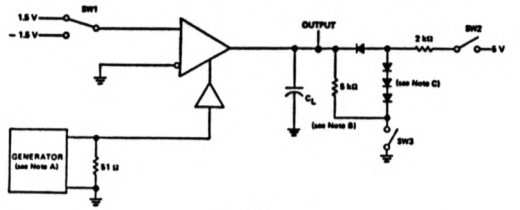


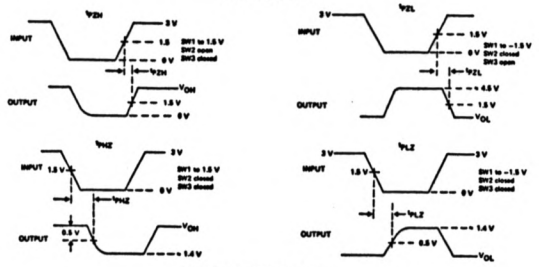
FIGURE 2–PROPAGATION DELAY TIMES

NOTES: A. The input pulse is supplied by a generator having the following characteristics: PRR = 1 MHz, duty cycle = 50%, t_r = t_f = 6 ns, Z_{out} = 50 Ω.
B. C_L includes probe and stray capacitance.

PARAMETER MEASUREMENT INFORMATION



TEST CIRCUIT



VOLTAGE WAVEFORMS

FIGURE 3—ENABLE AND DISABLE TIMES

NOTES: A. The input pulse is supplied by a generator having the following characteristics: PRR = 1 MHz, duty cycle = 50%, $t_r = t_f = 6$ ns, $Z_{out} = 50\ \Omega$.
B. C_L includes probe and stray capacitance
C. All diodes are 1N916 or equivalent.

TYPICAL CHARACTERISTICS

OUTPUT VOLTAGE vs DIFFERENTIAL INPUT VOLTAGE

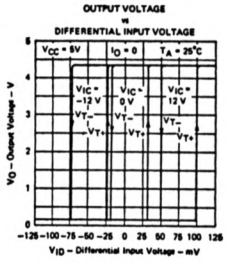


FIGURE 4

HIGH-LEVEL OUTPUT VOLTAGE vs HIGH-LEVEL OUTPUT CURRENT

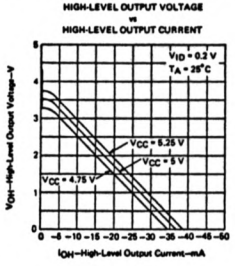


FIGURE 5

HIGH-LEVEL OUTPUT VOLTAGE vs FREE-AIR TEMPERATURE

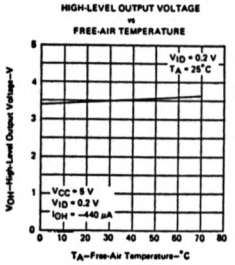


FIGURE 6

TYPICAL CHARACTERISTICS

LOW-LEVEL OUTPUT VOLTAGE vs LOW-LEVEL OUTPUT CURRENT

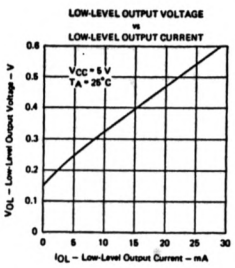


FIGURE 7

LOW-LEVEL OUTPUT VOLTAGE vs FREE-AIR TEMPERATURE

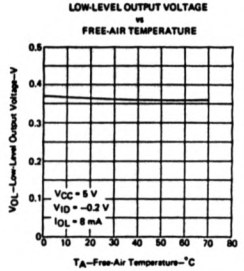


FIGURE 8

OUTPUT VOLTAGE vs ENABLE VOLTAGE

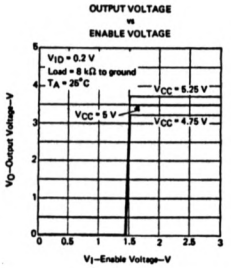


FIGURE 9

OUTPUT VOLTAGE vs ENABLE VOLTAGE

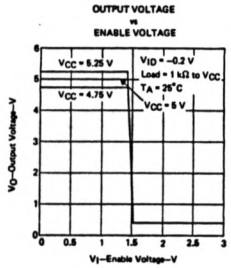


FIGURE 10

TYPICAL APPLICATION

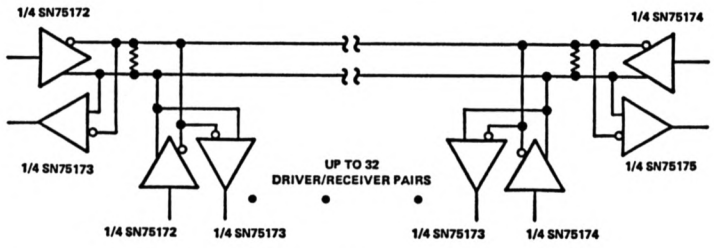


FIGURE 11

NOTE: The line should be terminated at both ends in its characteristic impedance. Stub lengths off the main line should be kept as short as possible.

BURR-BROWN®

SDM854

FOR A COMPLETE DATA SHEET, SEE PDS-423D

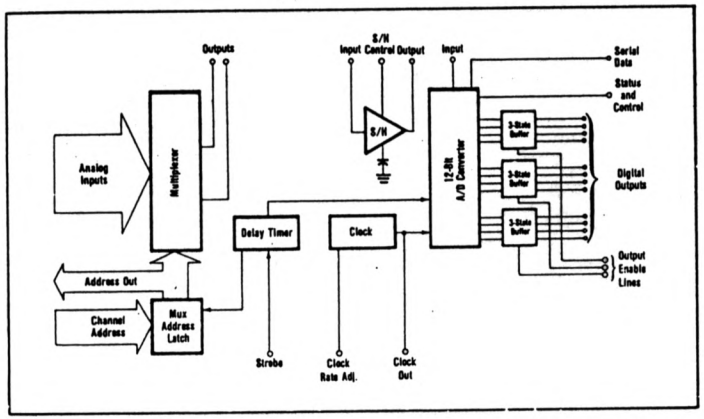
HYBRID DATA ACQUISITION SYSTEM

FEATURES

- 12-BIT, ±0.012% LINEARITY ERROR
- INPUTS UP TO ±10 VOLTS
- WIDE TEMPERATURE RANGE
- SELECTABLE 16 SINGLE, 8 DIFFERENTIAL INPUTS
- THREE-STATE OUTPUT BUFFERS

DESCRIPTION

The SDM854 is a complete data acquisition system contained in a miniature 2.2" x 1.7" x 0.22" (55.9mm x 43.2mm x 5.6mm) ceramic package. This system offers all the functions available in large modular data acquisition systems. Inputs up to ±10V can be accepted and low-level inputs can be accommodated by connecting an external instrumentation amplifier to the output of the multiplexer and to the input of the sample/hold amplifier. Digital resolution is 12 bits with accuracy of ±0.024% at a throughput rate of 27kHz.



International Airport Industrial Park · P.O. Box 11400 · Tucson, Arizona 85734 · Tel. (602) 746-1111 · Twx: 910-952-1111 · Cable: BBRCORP · Telex: 66-6491

SYSTEM DESCRIPTION

The SDM854 contains all components necessary to multiplex and convert analog signals up to ±10V into equivalent digital outputs. Throughput sampling rates are from 27kHz (12-bit resolution) to 70kHz (8-bit resolution) in the overlap mode of operation. The SDM854 can be configured to accept either 8-channel differential or 16-channel single-ended signals and can be expanded almost without limit with external multiplexers. Three-state outputs are provided for easy interface to microprocessor and other bus-structure systems. The system components are illustrated in Figure 1 and described in the following paragraphs.

ANALOG MULTIPLEXER

The analog multiplexer consists of two CMOS integrated circuits. Pin interconnects are used to select 16-channel single-ended or 8-channel differential operation. In single-ended operation the multiplexer can be used in a pseudo-differential mode by connecting an external amplifier's inverting input to common remote signal ground. Channel selection is made by an internally latched 3- or 4-bit binary word, for differential or single-ended operation respectively.

SAMPLE/HOLD

A complete stand-alone circuit, the sample/hold amplifier features buffered output, 10μsec acquisition time, and 100nsec aperture time.

Input, output, and mode control lines are brought out to separate pins. This allows maximum system flexibility for performing functions, such as automatic gain ranging, with no loss of aperture time.

ANALOG-TO-DIGITAL CONVERTER

The ADC is a 12-bit, 25μsec converter with 0.01% linearity error. Its features include positive and negative reference voltage outputs, external gain and offset adjustments, straight binary or two's complement output, serial data and clock outputs, status output, a short cycle feature, and a clock rate control for higher throughput rates at lower resolution or accuracy.

THREE-STATE OUTPUT BUFFERS

Digital outputs of the ADC are internally buffered by LSTTL three-state buffers. Three separate enable lines are brought out for easy interfacing to 4-, 8- or 16-bit data buses. $\overline{\text{MSB}}$ and BUSY are also buffered by separate three-state devices, each with its own enable line.

ADDRESS LATCH

Outputs of the 4-bit LSTTL register latch are connected to the address inputs of the multiplexer. This latch serves as an address storage register for the selected analog input. It may be loaded through 4 address inputs. Other inputs are LOAD and CLEAR. The 3 least significant bits are used for 8-channel differential mode addressing.

DELAY TIMER

A delay timer allows settling time for the multiplexer and sample/hold circuits before conversion begins. The delay is adjustable over a wide range by use of an external resistor or capacitor. This allows for longer settling time if an external instrumentation amplifier is used and is operating at high gains, or shorter settling time for lower resolution operation.

CHANNEL EXPANSION

The number of analog input channels of the SDM854 can be easily increased by using Burr-Brown's MPC8D (8-channel differential) and MPC16S (16-channel single-ended) multiplexers. These are latch-free devices which contain internal binary decoding at TTL or MOS levels and may be integrated into a system with minimal external logic.

SYSTEM PERFORMANCE

The SDM854 is configured for random channel selection. With the addition of an external counter they can be configured to continuously sequence through all analog channels or sequence through all analog channels on command from an external trigger.

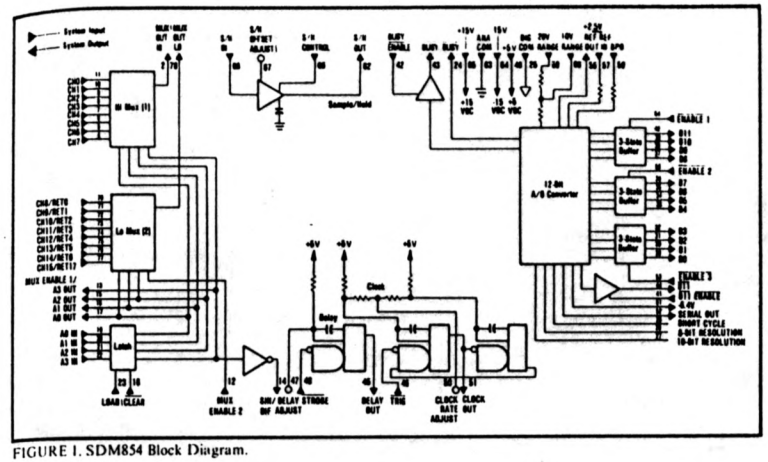


FIGURE 1. SDM854 Block Diagram.

With the appropriate 4-bit (single-ended) or 3-bit (differential) channel address on the latch inputs, and DELAY OUT (pin 45) tied to the LOAD input (pin 23), a negative going edge is applied to the $\overline{\text{STROBE}}$ input (pin 48). This starts the delay timer, latches the multiplexer address, and allows the input signal to pass through the multiplexer, and sample/hold before starting the A/D conversion. The DELAY OUT signal (pin 45) is also connected to the $\overline{\text{TRIG}}$ input (pin 46) and the A/D conversion is initiated on the negative-going edge. The S/H CONTROL input (pin 66) is connected to $\overline{\text{BUSY}}$ (pin 24) so that the sample/hold is in the HOLD mode during the A/D conversion.

By using overlap programming the settling time effects of the analog multiplexer and external instrumentation amplifier (if used) can be reduced, extending throughput sampling rates up to 27kHz for 12-bit and 70kHz for 8-bit resolution (ADC short-cycled). This mode of operation is most useful when converting low-level inputs to accommodate the increased settling time of the external instrumentation amplifier. Overlap programming is accomplished by connecting $\overline{\text{BUSY}}$ to $\overline{\text{STROBE}}$ and S/H CONTROL to DELAY OUT to LOAD and $\overline{\text{TRIG}}$. In this mode of operation the address of the next channel to be converted is latched and the output of the external instrumentation amplifier allowed to settle to a new value during the present conversion.

DIGITAL INPUT SPECIFICATIONS

Address Inputs (A0 - A3)	One standard LSTTL load, positive true
Address Coding	4-bit binary
LOAD	One standard LSTTL load, positive true, address loaded on positive edge.
$\overline{\text{CLEAR}}$	One standard LSTTL load, negative true, low level clears address latch.
$\overline{\text{STROBE}}$	One standard TTL load, high-to-low transition triggers the delay timer.
$\overline{\text{TRIG}}$	One standard TTL load, a negative going edge initiates the A/D conversion.
SHORT CYCLE	One standard LSTTL load, logic 1 for 12-bit resolution. Connect to "8-bit" or "10-bit" for 8- or 10-bit resolution.
$\overline{\text{ENABLE 1}}$, $\overline{\text{ENABLE 2}}$, $\overline{\text{ENABLE 3}}$, $\overline{\text{DTI}}$ ENABLE	One standard LSTTL load, a low level enables the 3-state output.
BUSY ENABLE	
S/H CONTROL	TTL compatible, 10μA maximum input current. Logic 0 = Hold mode, Logic 1 = Sample (track) mode.
MUX ENABLE 2	TTL compatible, 2μA input current, logic 0 enables multiplexer 2 (channels 8-15).

DIGITAL OUTPUT SPECIFICATIONS

Parallel Data Outputs	5 standard TTL loads, positive true 3-state.
Serial Output	2 standard TTL loads, positive true, NRZ, time serial data output beginning with DI1 (see Timing Diagram).
$\overline{\text{DTI}}$	5 standard TTL loads, positive true, 3-state.
$\overline{\text{BUSY}}$	5 standard TTL loads, low during A/D conversion.
BUSY	5 standard TTL loads, high during A/D conversion, 3-state
CLOCK OUT	5 standard TTL loads, for synchronizing serial out data (see Timing Diagram).
Address Outputs (A0 - A3)	5 LSTTL or 2 standard TTL loads, positive true
DELAY OUT	5 standard TTL loads, high during delay period, triggered by Strobe input.
SIN/DIF	5 LSTTL or 2 standard TTL loads, high while addressing channels 0-7, low while addressing channels 8-15.

SPECIFICATIONS

ELECTRICAL

Typical at T_A = +25°C and rated power supplies unless otherwise noted.

PARAMETER	MIN	TYP	MAX	UNITS
TRANSFER CHARACTERISTICS				
Resolution	12			Bits
Number of Analog Channels		16SIN/8DIF		
Throughput Rate Normal mode				
SDM854AG	33	35		kHz
SDM854BG	25	27		kHz
Throughput Rate Overlap mode				
SDM854AG	38	40		kHz
SDM854BG	27	29		kHz
ANALOG INPUTS				
ADC Input Voltage Ranges		0 to +10, ±5, ±10		V
Mux Input Voltage Range				
Absolute max without damage			±35	V
For linear operation			±15	V
Mux Input Impedance, OFF Channel			10[1]	kΩ
Mux Input Impedance, ON Channel		1.5	1.8	kΩ
Input Leakage, OFF Channel		0.02		nA
Output Leakage, All Channels Disabled		0.2		nA
Output Leakage with Input Overvoltage of				
+35V		1		nA
-35V		1		μA
TEMPERATURE STABILITY				
System Accuracy				
Unipolar		±15	±25	ppm/°C
Bipolar		±10	±20	ppm/°C
Linearity Drift			±2	ppm/°C of FSR
REFERENCE VOLTAGES				
Positive Output	+2.490	+2.500	+2.510	V
Positive Output Drift		±5	±10	ppm/°C
Negative Output	-6.0	-6.4	-6.8	V
Negative Output Drift		±15	±10	ppm/°C
ACCURACY				
Throughput Accuracy				
0 to +10V, ±5V, ±10V, AG			±0.048	% of FSR[1]
0 to +10V, ±5V, ±10V, BG			±0.024	% of FSR
Linearity				
AG			±0.024	% of FSR
BG			±0.012	% of FSR
Differential Linearity				
AG		±0.024	±0.048	% of FSR
BG		±0.012	±0.024	% of FSR
Quantizing Error			±0.012	% of FSR
System Gain Error[2]		±0.1	±0.3	%
System Offset Error[2]		±0.1	±0.3	% of FSR
Power Supply Sensitivity +15V		±0.0007		%/%ΔV
Power Supply Sensitivity -15V		±0.0007		%/%ΔV
Power Supply Sensitivity +5V		±0.001		%/%ΔV
DYNAMIC ACCURACY				
Sample/Hold Characteristics				
Aperture Time		100		nsec
Acquisition Time		10		μsec
Feedthrough 10V step		±1.4		mV
OUTPUTS				
Digital Output Coding		Binary, Offset Binary, Two's Complement		
Serial Output Coding		Nonreturn to zero NRZ		
ADC Conversion Time[3]		25	30	μsec
Clock Frequency[3]		520		kHz
Delay[4]		15		μsec

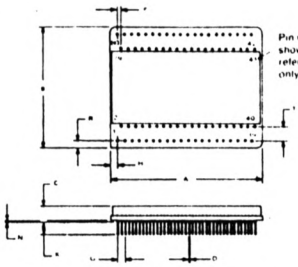
PARAMETER	MIN	TYP	MAX	UNITS
POWER REQUIREMENTS				
Rated Voltage for Specified Accuracy	±14.5	±15	±15.5	V
	+4.75	+5	+5.25	V
Quiescent Current				
+15VDC		+10	+20	mA
-15VDC		-35	-50	mA
+5VDC		+170	+220	mA
Power Dissipation		1300	1750	mW
ENVIRONMENTAL				
Specification Temperature Range	-25		+85	°C
Operating Temperature Range	-40		+85	°C
Storage Temperature Range	-55		+125	°C

NOTES:
1. FSR means Full Scale Range (FSR is 20V for ±10V range).
2. Adjustable to zero.
3. Conversion time and clock frequency can be externally adjusted from 13μsec (f$_{clock}$ = 1.0MHz) to 110μsec (f$_{clock}$ = 118kHz). Conv. times are for 12-bit resolution.
4. Can be externally adjusted from 3μsec to 300μsec.

PIN DESIGNATIONS

NC	1	80	NC
MUX OUT HI	2	79	NC
NC	3	78	MUX OUT LO
CH7	4	77	CH15/RET7
CH6	5	76	CH14/RET6
CH5	6	75	CH13/RET5
CH4	7	74	CH12/RET4
CH3	8	73	CH11/RET3
CH2	9	72	CH10/RET2
CH1	10	71	CH9/RET1
CH0	11	70	CH8/RET0
MUX ENABLE 2	12	69	NC
MUX ENABLE 1/A3 OUT	13	68	S/H IN
SIN/DIF	14	67	S/H OFFSET ADJUST
A2 OUT	15	66	S/H CONTROL
A1 OUT	16	65	+15VDC
A0 OUT	17	64	-15VDC
$\overline{\text{CLEAR}}$	18	63	ANA COM
A0 IN	19	62	S/H OUT
A1 IN	20	61	-6.4V REF OUT
A2 IN	21	60	10V RANGE
A3 IN	22	59	BIPOLAR OFFSET
LOAD	23	58	20V RANGE
$\overline{\text{BUSY}}$	24	57	+2.5V REF IN
DIG COM	25	56	$\overline{\text{ENABLE 2}}$
SHORT CYCLE	26	55	+2.5V REF OUT
10-BIT RESOLUTION	27	54	$\overline{\text{ENABLE 1}}$
8-BIT RESOLUTION	28	53	$\overline{\text{ENABLE 3}}$
D0 LSB	29	52	SERIAL OUT
D1	30	51	CLOCK OUT
D2	31	50	CLOCK RATE ADJUST
D3	32	49	+5VDC
D4	33	48	$\overline{\text{STROBE}}$
D5	34	47	DELAY ADJUST
D6	35	46	$\overline{\text{TRIG}}$
D7	36	45	DELAY OUT
D8	37	44	$\overline{\text{DTI}}$
D9	38	43	BUSY
D10	39	42	BUSY ENABLE
D11 MSB	40	41	$\overline{\text{DTI}}$ ENABLE

MECHANICAL



Pin numbers shown for reference only

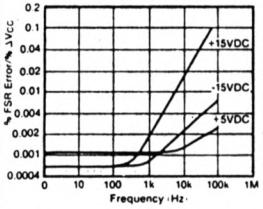
NOTE
Leads in true position within 0.015 0.38mm R at MMC at seating plane

DIM	INCHES		MILLIMETERS	
	MIN.	MAX.	MIN.	MAX.
A	2.120	2.180	53.85	55.37
B	1.610	1.720	40.89	43.69
C	.110	.270	2.79	6.86
D	.018	.021	0.46	0.53
F	.035	.050	0.89	1.27
G	.100 BASIC		2.54 BASIC	
H	.100 BASIC		2.54 BASIC	
K	.150	.250	3.81	6.35
L	1.500 BASIC		38.1 BASIC	
M	.007	.010	0.25	0.31
P	.050 BASIC		1.27 BASIC	
R	.100 BASIC		2.54 BASIC	
T	.200 BASIC		5.08 BASIC	
U	1.100 BASIC		27.94 BASIC	

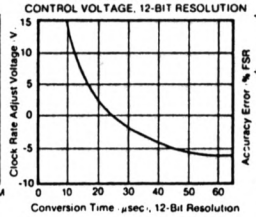
MATERIAL Ceramic
WEIGHT 32 grams 1.2 oz
MATING
CONNECTOR
2350MC set of four
20-pin strips or
0422MC assembled unit

TYPICAL PERFORMANCE CURVES

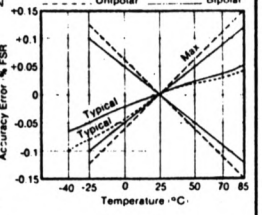
POWER SUPPLY REJECTION VS POWER SUPPLY RIPPLE FREQUENCY



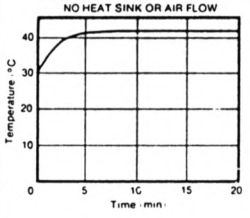
A/D CONVERSION TIME VS CLOCK RATE CONTROL VOLTAGE, 12-BIT RESOLUTION



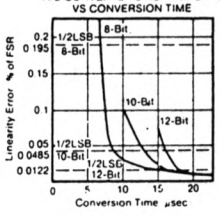
SYSTEM ACCURACY DRIFT



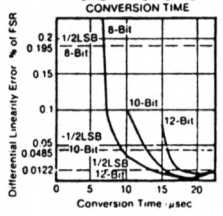
CASE TEMPERATURE VS TIME WITH NO HEAT SINK OR AIR FLOW



A/D CONVERTER LINEARITY ERROR VS CONVERSION TIME



A/D CONVERTER DIFFERENTIAL LINEARITY ERROR VS CONVERSION TIME



BURR-BROWN®

INA101

Very-High Accuracy INSTRUMENTATION AMPLIFIER

FEATURES

- ULTRA-LOW VOLTAGE DRIFT - 0.25μV/°C
- LOW OFFSET VOLTAGE - 25μV
- LOW NONLINEARITY - 0.002%
- LOW NOISE - 13nV/√Hz at f_0 = 1kHz
- HIGH CMR - 106dB at 60Hz
- HIGH INPUT IMPEDANCE - $10^{10}\Omega$
- LOW COST, TO-100, CERAMIC DIP AND PLASTIC PACKAGE

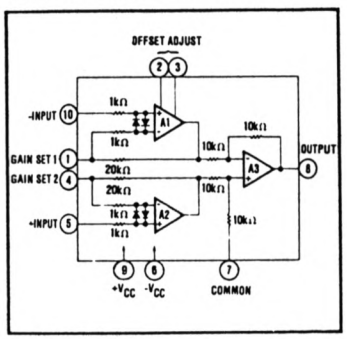
APPLICATIONS

- AMPLIFICATION OF SIGNALS FROM SOURCES SUCH AS:
  - Strain Gages
  - Thermocouples
  - RTDs
- REMOTE TRANSDUCERS
- LOW LEVEL SIGNALS
- MEDICAL INSTRUMENTATION

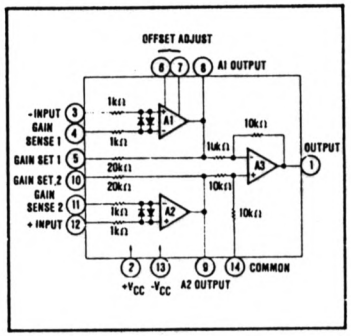
DESCRIPTION

The INA101 is a high accuracy, multistage, integrated-circuit instrumentation amplifier designed for signal conditioning requirements where very-high performance is desired. All circuits, including the interconnected laser-trimmed thin-film resistors, are integrated on a single monolithic substrate.

A multiamplifier design is used to provide the highest performance and maximum versatility with monolithic construction for low cost. The input stage uses Burr-Brown's ultra-low drift, low noise technology to provide exceptional input characteristics.



M Package



G and P Packages

International Airport Industrial Park - P.O. Box 11400 - Tucson, Arizona 85734 Tel. (602) 746-1111 - Twx: 910-952-1111 - Cable: BBRCORP - Telex: 66-6491

SPECIFICATIONS

ELECTRICAL

At +25°C with ±15VDC power supply and in circuit of Figure 2 unless otherwise noted.

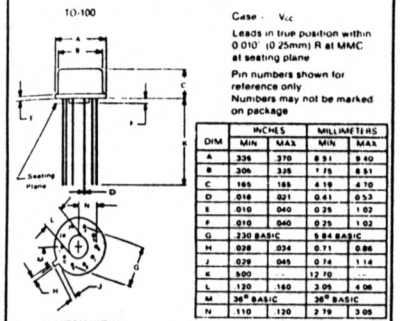
MODEL	INA101AM/AG			INA101SM/SG			INA101CM/CG			INA101HP			
	MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	UNITS
GAIN													
Range of Gain	1		1000	*		*	*		*	*		*	V/V
Gain Equation		G=1+(40k/R_G)			*			*			*		V/V
Error From Equation, DC[1]		±(0.04+0.00016 −0.02/G)			*	*		*	*		±(0.1+ 0.00015G) −0.05/G	±(0.3+ 0.0002G) −0.10/G	%
Gain Temp. Coefficient[2]													
G = 1		2	5		*	*		*	*		*	*	ppm/°C
G = 10		20	100		*	*		10	*		*	*	ppm/°C
G = 100		22	110		*	*		11	*		*	*	ppm/°C
G = 1000		22	110		*	*		11	*		*	*	ppm/°C
Nonlinearity, DC[3]		±(0.002+10^{-6}G)	±(0.005+2×10^{-6}G)		±(0.001 +10^{-6}G)	±(0.002 +10^{-6}G)		±(0.001 +10^{-6}G)	±(0.002 +10^{-6}G)		*	*	% of p-p FS
RATED OUTPUT													
Voltage	±10	±12.5		*	*		*	*		*	*		V
Current	±5	±10		*	*		*	*		*	*		mA
Output Impedance		0.2			*			*			*		Ω
Capacitive Load		1000			*			*			*		pF
INPUT OFFSET VOLTAGE													
Initial Offset at +25°C		±(25+200/G)	±(50+400/G)		±(10+ 100/G)	±(25 +200/G)		±(10+ 100/G)	±(25+ 200/G)		±(125+ 450/G)	±(250+ 900/G)	μV
vs Temperature			±(2+20/G)			±(0.75 +10/G)			±(0.25+ 10/G)		±(2+20/G)		μV/°C
vs Supply		±(1+20/G)			*			*			*		μV/V
vs Time		±(1+20/G)			*			*			*		μV/mo
INPUT BIAS CURRENT													
Initial Bias Current (each input)		±15	±30		±10	*		±5	±20		*	*	nA
vs Temperature		±0.2			*			*			*		nA/°C
vs Supply		±0.1			*			*			*		nA/V
Initial Offset Current		±15	±30		±10	*		±5	±20		*	*	nA
vs Temperature		±0.5			*			*			*		nA/°C
INPUT IMPEDANCE													
Differential		10^{10}‖3			*			*			*		Ω‖pF
Common-mode		10^{10}‖3			*			*			*		Ω‖pF
INPUT VOLTAGE RANGE													
Range, Linear Response	±10	±12		*	*		*	*		*	*		V
CMR with 1kΩ Source Imbal													
DC to 60Hz, G=1	80	90		*	*		*	*		65	85		dB
DC to 60Hz, G=10	96	106		*	*		*	*		90	95		dB
DC to 60Hz, G=100 to 1000	106	110		*	*		*	*		100	105		dB
INPUT NOISE													
Input Voltage Noise													
f_0=0.01Hz to 10Hz		0.8			*			*			*		μV, p-p
Density, G=1000													
f_0=10Hz		18			*			*			*		nV/√Hz
f_0=100Hz		15			*			*			*		nV/√Hz
f_0=1kHz		13			*			*			*		nV/√Hz
Input Current Noise													
f_0=0.01Hz to 10Hz		50			*			*			*		pA p-p
Density													
f_0=10Hz		0.8			*			*			*		pA/√Hz
f_0=100Hz		0.46			*			*			*		pA/√Hz
f_0=1kHz		0.35			*			*			*		pA/√Hz
DYNAMIC RESPONSE													
Small Signal, ±3dB Flatness													
G = 1		300			*			*			*		kHz
G = 10		140			*			*			*		kHz
G = 100		25			*			*			*		kHz
G = 1000		2.5			*			*			*		kHz
Small Signal, ±1% Flatness													
G = 1		70			*			*			*		kHz
G = 10		10			*			*			*		kHz
G = 100		1			*			*			*		kHz
G = 1000		200			*			*			*		Hz
Full Power, G=1 to 100		6.4			*			*			*		kHz
Slew Rate, G=1 to 100	0.2	0.4		*	*		*	*		*	*		V/μs
Settling Time (0.1%)													
G = 1		30	40		*	*		*	*		*	*	μs
G = 100		40	55		*	*		*	*		*	*	μs
G = 1000		350	470		*	*		*	*		*	*	μs
Settling Time (0.01%)													
G = 1		30	45		*	*		*	*		*	*	μs
G = 100		50	70		*	*		*	*		*	*	μs
G = 1000		500	650		*	*		*	*		*	*	μs
POWER SUPPLY													
Rated Voltage		±15			*			*			*		V
Voltage Range	±5		±20	*		*	*		*	*		*	V
Current, Quiescent[4]		±6.7	±8.5		*	*		*	*		*	*	mA
TEMPERATURE RANGE[5]													
Specification	25		+85	55		+125	*		*		0	+70	°C
Operation	55		+125	*		*	*		*		25	+85	°C
Storage	65		+150	*		*	*		*		40	+85	°C

* Specification is same as for INA101AM-AG.

NOTES: (1) Typically the tolerance of R_G will be the major source of gain error. (2) Nonlinearity is the maximum peak deviation from the best straight line as a percentage of peak to peak full scale output. (3) Not including the TCR of R_G. (4) Adjustable to zero at any one gain. (5) θ_{JA} output stage 113°C/W, θ_{JA} quiescent circuits θ_{JA} 83°C/W.

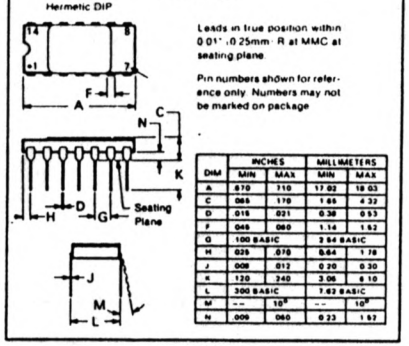
MECHANICAL

M Package



DIM	INCHES MIN	INCHES MAX	MILLIMETERS MIN	MILLIMETERS MAX
A	.335	.370	8.51	9.40
B	.305	.335	7.75	8.51
C	.165	.185	4.19	4.70
D	.016	.021	0.41	0.53
E	.010	.040	0.25	1.02
F	.010	.040	0.25	1.02
G	.230 BASIC		5.84 BASIC	
H	.028	.034	0.71	0.86
J	.029	.045	0.74	1.14
K	.500	--	12.70	--
L	.120	.160	3.05	4.06
M	36° BASIC		36° BASIC	
N	.110	.120	2.79	3.05

G Package



DIM	INCHES MIN	INCHES MAX	MILLIMETERS MIN	MILLIMETERS MAX
A	.670	.710	17.02	18.03
C	.065	.170	1.65	4.32
D	.016	.021	0.38	0.53
F	.045	.080	1.14	1.52
G	.100 BASIC		2.54 BASIC	
H	.025	.070	0.64	1.78
J	.008	.012	0.20	0.30
K	.120	.240	3.06	6.10
L	.300 BASIC		7.62 BASIC	
M	--	15°	--	15°
N	.009	.060	0.23	1.52

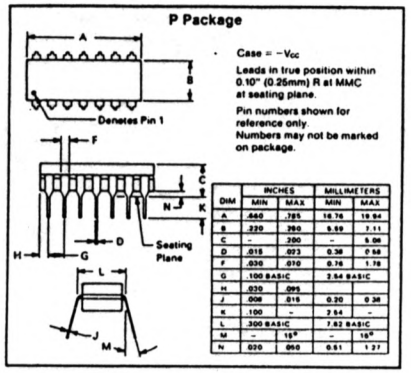
ORDERING INFORMATION

INA101 C G

- Basic Model Number
- Performance Grade Code
  - S: −55°C to +125°C
  - A, C: −25°C to +85°C
  - H: 0°C to +70°C
- Package Code
  - M: TO-100
  - G: 14-Pin Hermetic DIP
  - P: 14-Pin Plastic DIP

TO-100 (M Suffix)	Hermetic DIP (G Suffix)	Plastic DIP (P Suffix)
INA101AM	INA101AG	INA101HP
INA101CM	INA101CG	
INA101SM	INA101SG	

P Package

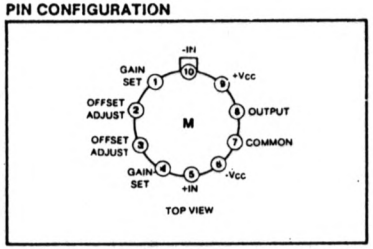


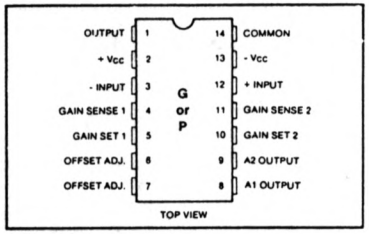
DIM	INCHES MIN	INCHES MAX	MILLIMETERS MIN	MILLIMETERS MAX
A	.660	.785	16.76	19.94
B	.220	.280	5.59	7.11
C	-	.200	-	5.08
D	.015	.023	0.38	0.58
F	.030	.070	0.76	1.78
G	.100 BASIC		2.54 BASIC	
H	.030	.095		
J	.008	.015	0.20	0.38
K	.100	-	2.54	-
L	.300 BASIC		7.62 BASIC	
M	-	15°	-	15°
N	.020	.060	0.51	1.27

ABSOLUTE MAXIMUM RATINGS

Supply	±20V
Internal Power Dissipation	600mW
Input Voltage Range	±V_{CC}
Operating Temperature Range	−55°C to +125°C
Storage Temperature Range:	
M, G	−65°C to +150°C
P	−40°C to +85°C
Lead Temperature (soldering 10 seconds)	+300°C
Output Short-Circuit Duration	Continuous to ground

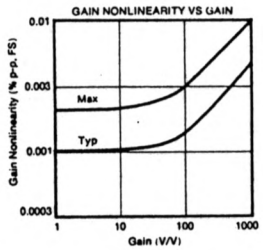
PIN CONFIGURATION

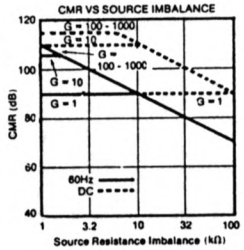


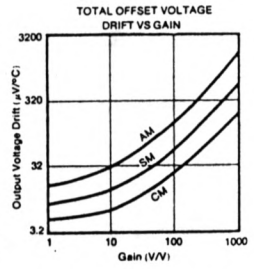


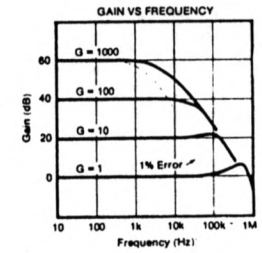
TYPICAL PERFORMANCE CURVES

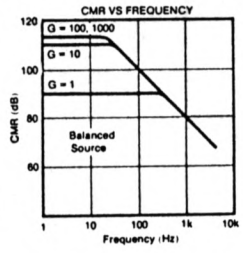
At +25°C and in circuit of Figure 2 unless otherwise noted.

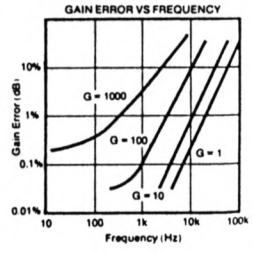


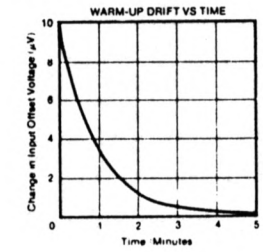


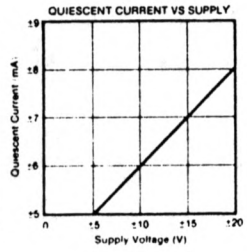


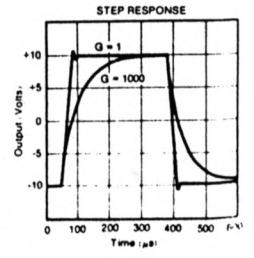


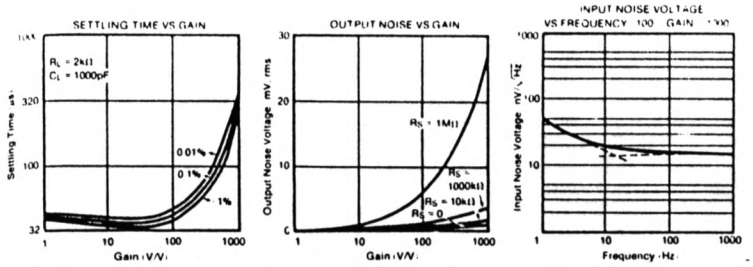












DISCUSSION OF PERFORMANCE

INSTRUMENTATION AMPLIFIERS

Instrumentation amplifiers are differential input closed-loop gain blocks whose committed circuit accurately amplifies the voltage applied to their inputs. They respond only to the difference between the two input signals and exhibit extremely-high input impedance, both differentially and common-mode. Feedback networks are packaged within the amplifier module. Only one external gain setting resistor must be added. An operational amplifier, on the other hand, is an open-loop, uncommitted device that requires external networks to close the loop. While op amps can be used to achieve the same basic function as instrumentation amplifiers, it is very difficult to reach the same level of performance. Using op amps often leads to design trade-offs when it is necessary to amplify low level signals in the presence of common-mode voltages while maintaining high input impedances. Figure 1 shows a simplified model of an instrumentation amplifier that eliminates most of the problems.

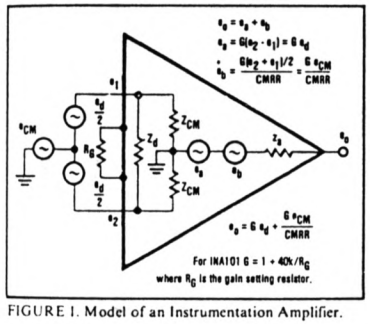


FIGURE 1. Model of an Instrumentation Amplifier.

THE INA101

Simplified schematics of the INA101 are shown on the first page. It is a three-amplifier device which provides all the desirable characteristics of a premium performance instrumentation amplifier. In addition, it has features not normally found in integrated circuit instrumentation amplifiers.

The input section (A1 and A2) incorporates high performance, low drift amplifier circuitry. The amplifiers are connected in the noninverting configuration to provide the high input impedance ($10^{10}\Omega$) desirable in the instrumentation amplifier function. The offset voltage and offset voltage versus temperature is low due to the monolithic design and improved even further by the state-of-the-art laser-trimming techniques.

The output section (A3) is connected in a unity-gain difference amplifier configuration. A critical part of this stage is the matching of the four 10kΩ resistors which provide the difference function. These resistors must be initially well matched and the matching must be maintained over temperature and time in order to retain excellent common-mode rejection. (The 106dB minimum at 60Hz for gains greater than 100V/V is a significant improvement compared to most other integrated circuit instrumentation amplifiers.)

All of the internal resistors are compatible thin-film nichrome formed with the integrated circuit. The critical resistors are laser-trimmed to provide the desired high gain accuracy and common-mode rejection. Nichrome ensures long-term stability of trimmed resistors and simultaneous achievement of excellent TCR and TCR tracking. This provides gain accuracy and common-mode rejection when the INA101 is operated over wide temperature ranges.

USING THE INA101

Figure 2 shows the simplest configuration of the INA101. The gain is set by the external resistor, R_G, with a gain equation of $G = 1 + (40K/R_G)$. The reference and TCR of R_G contribute directly to the gain accuracy and drift.

For gains greater than unity, resistor R_G is connected externally between pins 1 and 4. At high gains where the value of R_G becomes small, additional resistance (i.e., relays, sockets) in the R_G circuit will contribute to a gain error. Care should be taken to minimize this effect.

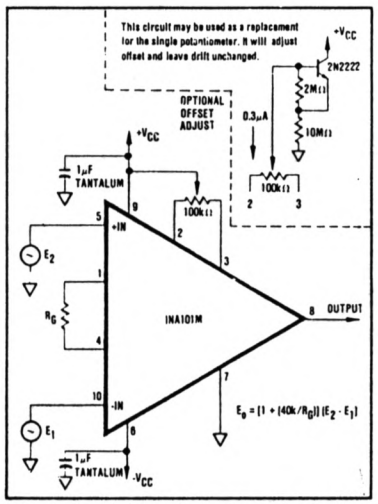


FIGURE 2. Basic Circuit Connection for the INA101 Including Optional Input Offset Null Potentiometer.

The optional offset null capability is shown in Figure 2. The adjustment affects only the input stage component of the offset voltage. Thus, the null condition will be disturbed when the gain is changed. Also, the input drift will be affected by approximately 0.31µV/°C per 100µV of input offset voltage that is trimmed. Therefore, care should be taken when considering use of the control for removal of other sources of offset. Output offsetting can be accomplished in Figure 3 by applying a voltage to Common (pin 7) through a buffer amplifier. This limits the resistance in series with pin 7 to minimize CMR error. Resistance above 0.1Ω will cause the common-mode rejection to fall below 106dB. Be certain to keep this resistance low.

It is important to not exceed the input amplifiers' dynamic range. The amplified differential input signal and its associated common-mode voltage should not cause the output of A_1 or A_2 to exceed approximately ±10V or nonlinear operation will result.

BASIC CIRCUIT CONNECTION

The basic circuit connection for the INA101 is shown in Figure 2. The output voltage is a function of the differential input voltage times the gain.

OPTIONAL OFFSET ADJUSTMENT PROCEDURE

It is frequently desirable to null the input component of offset (Figure 2) and occasionally that of the output (Figure 3). The quality of the potentiometer will affect the results, therefore, choose one with good temperature and mechanical-resistance stability. The procedure is as follows:

1. Set $E_1 = E_2 = 0V$ (be sure a good ground return path exists to the input).
2. Set the gain to the desired value by choosing R_G.
3. Adjust to 100kΩ potentiometer in Figure 2 until the output reads 0V ±1mV or desired setting. Note that the offset will change when the gain is changed. If the output component of offset is to be removed or if it is desired to establish an intentional offset, adjust the 100kΩ potentiometer in Figure 3 until the output reads 0V ±1mV or desired setting. Note that the offset will not change with gain, but be sure to use a stable external amplifier with good DC characteristics. The range of adjustment is ±15mV as shown. For larger ranges change the ratio of R_1 to R_2.

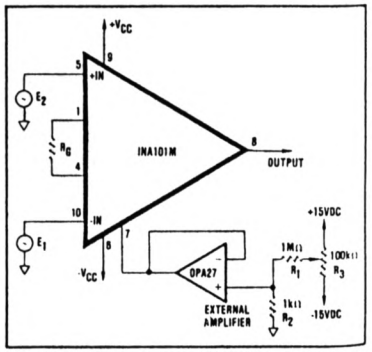


FIGURE 3. Optional Output Offset Nulling or Offsetting Using External Amplifier (Low Impedance to Pin 7).

THERMAL EFFECTS ON OFFSET

To maintain specified offset performance, especially in high gain, prevent air currents from circulating around the input pins. This can be done by using a skirted heat sink on the INA101M package. Rapid changes in die temperature and thermocouple effects on the pins will then be minimized. Surrounding the package with low power components will also help to reduce air flow across the package and pins.

TYPICAL APPLICATIONS

Many applications of instrumentation amplifiers involve the amplification of low level differential signals from

bridges and transducers such as strain gages, thermocouples, and RTD's. Some of the important parameters include common-mode rejection (differential cancellation of common-mode offset and noise, see Figure 1), input impedance, offset voltage and drift, gain accuracy, linearity, and noise. The INA101 accomplishes all of these with high precision.

Figures 4 through 16 show some typical applications circuits.

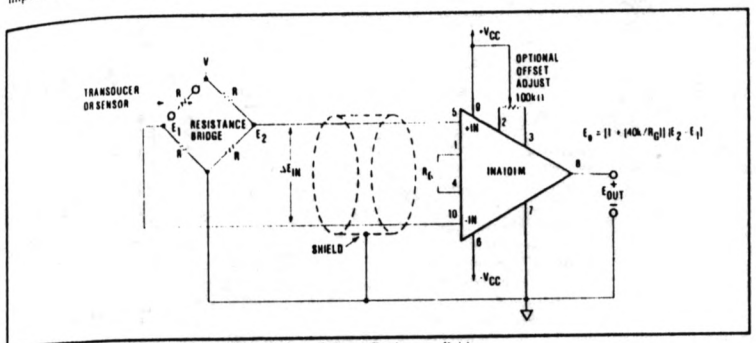


FIGURE 4. Amplification of a Differential Voltage from a Resistance Bridge.

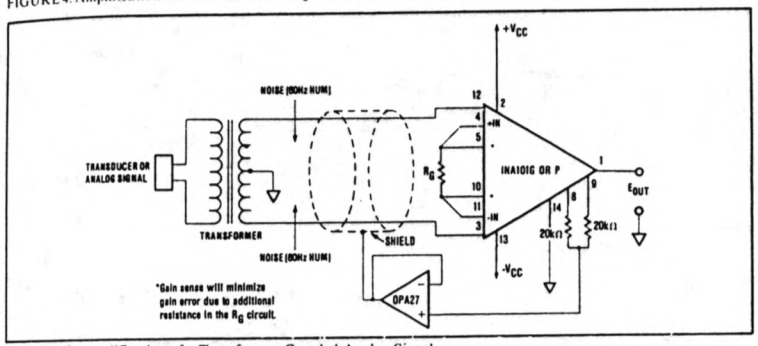


FIGURE 5. Amplification of a Transformer-Coupled Analog Signal.

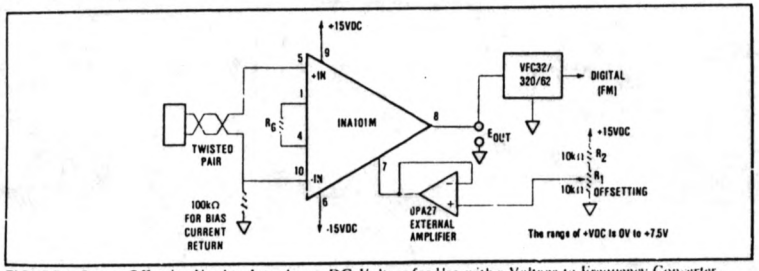


FIGURE 6. Output Offsetting Used to Introduce a DC Voltage for Use with a Voltage-to-Frequency Converter.

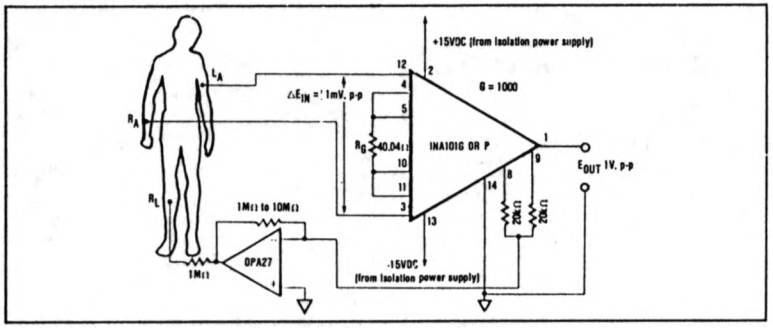


FIGURE 7. ECG Amplifier or Recorder Preamp for Biological Signals.

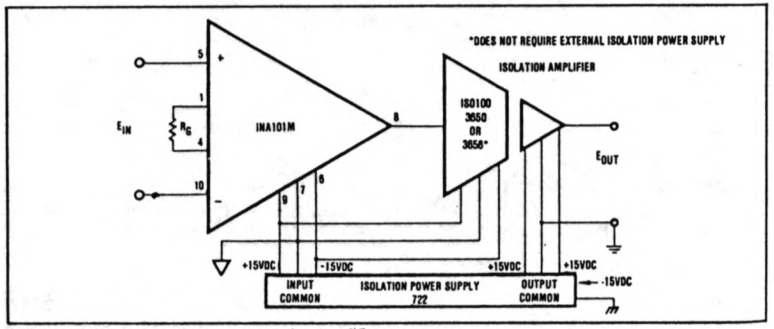


FIGURE 8. Precision Isolated Instrumentation Amplifier.

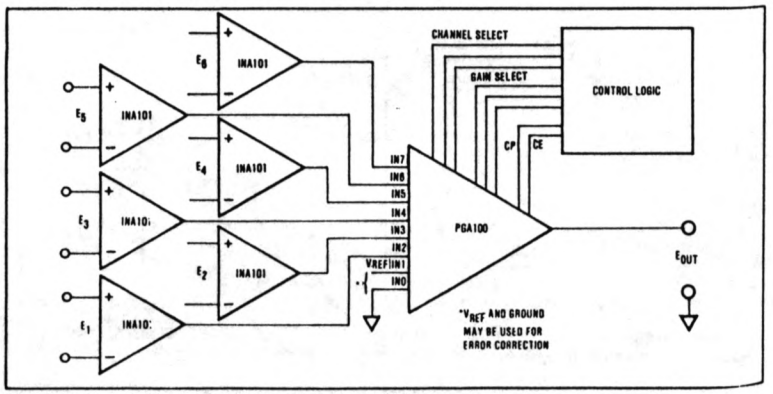


FIGURE 9. Multiple Channel Precision Instrumentation Amplifier.

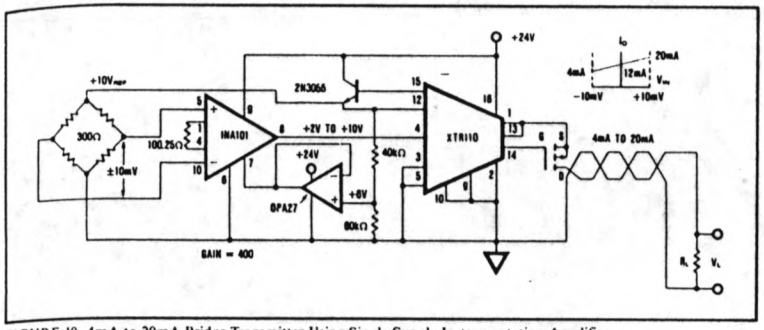


FIGURE 10. 4mA to 20mA Bridge Transmitter Using Single Supply Instrumentation Amplifier.

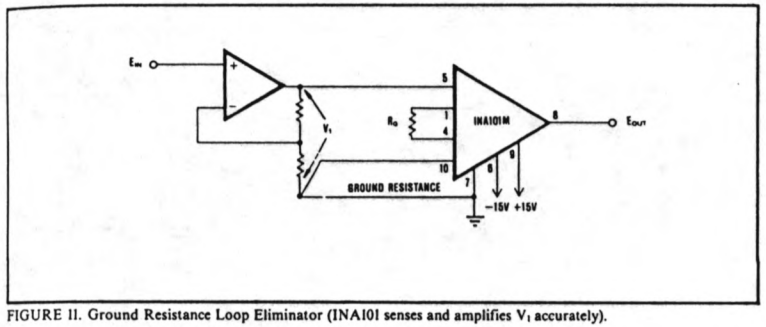


FIGURE 11. Ground Resistance Loop Eliminator (INA101 senses and amplifies V_1 accurately).

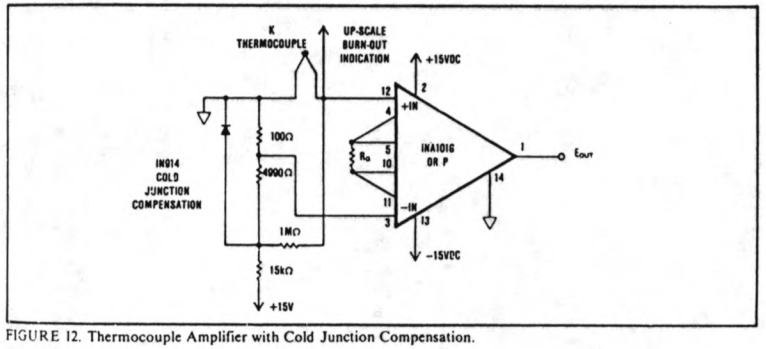


FIGURE 12. Thermocouple Amplifier with Cold Junction Compensation.

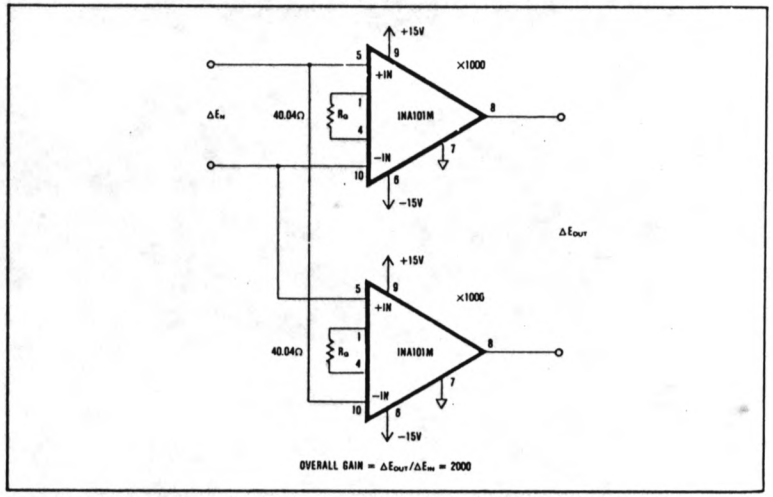


FIGURE 13. Differential Input/Differential Output Amplifier (twice the gain of one INA).

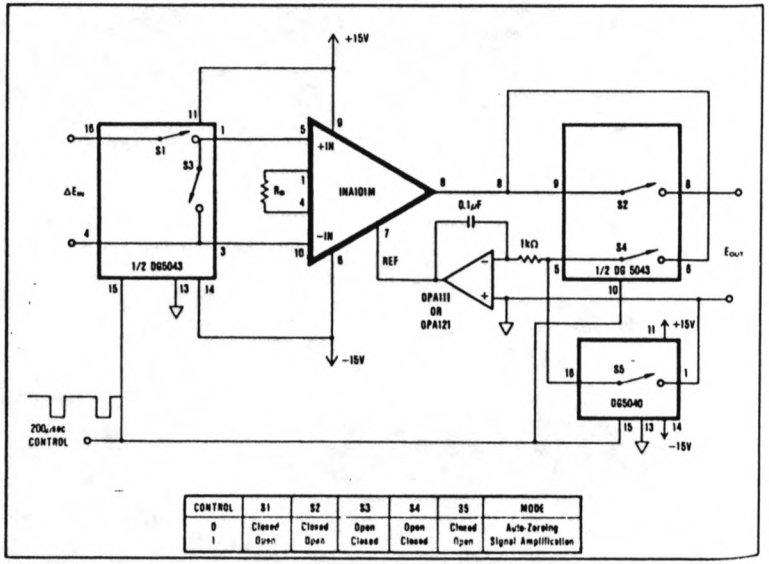


FIGURE 14. Auto-Zeroing Instrumentation Amplifier Circuit.

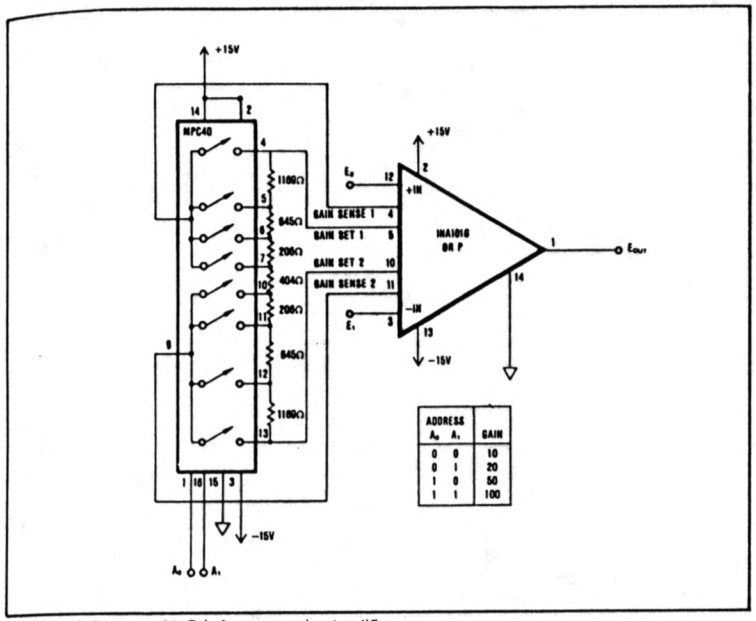


FIGURE 15. Programmable Gain Instrumentation Amplifier.

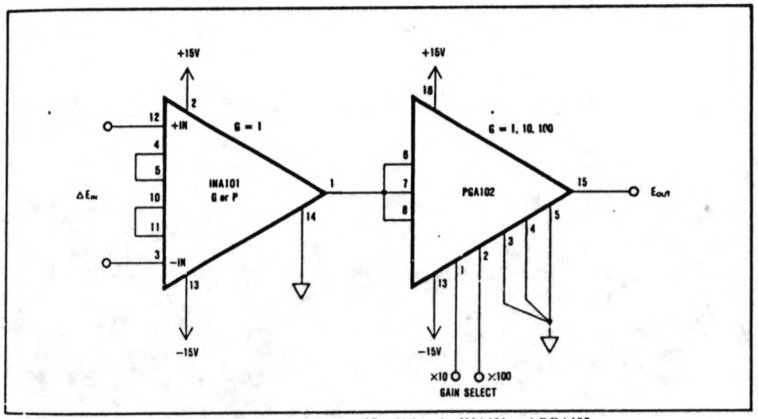


FIGURE 16. Programmable-Gain Instrumentation Amplifier Using the INA101 and PGA102.

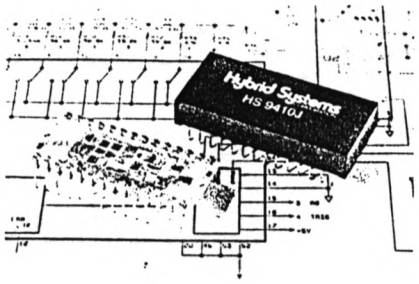
Hybrid Systems CORPORATION

HS 9410 Series

8 Channel, 12-Bit Data Acquisition System with μP Interface

FEATURES

- **Complete 8 Channel, 12-bit Data Acquisition System with MUX, S/H, REF, Clock and three-state outputs**
- **Full 8- or 16-Bit Microprocessor Bus Interface**
- **Guaranteed Linearity Over Temperature**
- **High Throughput Rate: 28kHz**
- **Hermetic 28-Pin Epoxy DIP**
- **Low Power: 1W**



DESCRIPTION

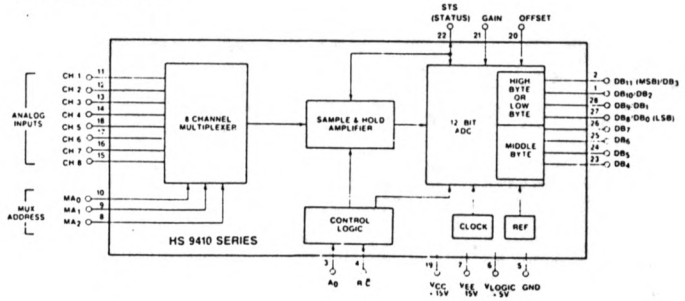
The HS 9410 Series is a complete 8 channel, microprocessor compatible, 12-bit data acquisition system with all the interface logic to connect directly to 8- or 16-bit microprocessor buses. It is contained in a 28-pin DIP and includes an 8 channel multiplexer, a sample-and-hold amplifier, and a 12-bit A/D converter along with the control logic needed to perform a complete data acquisition function. System throughput rate is 28kHz for full rated accuracy.

The Analog-to-Digital converter section contains the now standard HS 574 which is a 2 chip 12-bit ADC. This ADC is implemented with advanced bipolar and CMOS LSI chips resulting in maximum performance at lowest cost. The SAR, 12-bit decoded D/A, control logic, switches and buffers are fabricated using CMOS processing for lowest power. A unique comparator, reference and required amplifiers are fabricated using linear bipolar processes for maximum speed and reduced offset and drift over temperature.

Incorporating a unique precision comparator design, the ADC offers several advantages over more conventional circuits. A proprietary decoded 12-bit D/A provides increased accuracy, lower drift and reduced output noise over the A/D operating range. Precision low TCR laser trimmed resistors are used in the converter for setting critical performance parameters including gain, offset, input ranges, and accuracy.

The HS 9410 Series operates from ±15V and +5V with a total power consumption of 1W. To take advantage of the 28-pin package the user must specify input range of 0 to +10V, ±5V or ±10V when ordering. All units are rigorously tested including full power burn in at +85°C. Hybrid Systems guarantees Acceptable Quality Level (AQL) of 0.4% for all models which means that there are no rejects in a sample lot of 100 pieces.

FUNCTIONAL DIAGRAM



SPECIFICATIONS

(Typical @ +25°C with V_{CC} = +15V, V_{EE} = −15V, V_{LOGIC} = +5V, unless otherwise specified.)

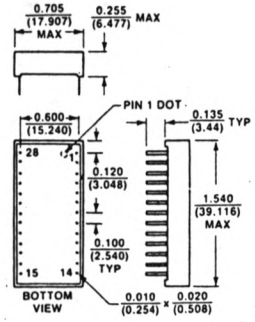
MODEL	HS 941XJ	HS 941XK
TRANSFER CHARACTERISTICS		
Resolution	12-Bits	*
Number of Channels	8 Single-Ended	*
Throughput Rate	28 kHz	*
ANALOG INPUTS		
Input Ranges[1] (Specified as a suffix in the model number. See Ordering Guide.)		
HS 9410	0 to +10V	*
HS 9411	±5V	*
HS 9412	±10V	*
Input Bias Current per Channel	30nA	*
Input Impedance		
ON Channel	10^{10} Ω‖ 100pf	*
OFF Channel	10^{10} Ω‖ 10pf	*
Crosstalk	80 dB	*
DIGITAL INPUTS		
Logic Inputs		
R/C, A_0		
Logic 1	+2.0V min, +5.5V max	*
Logic 0	−0.5V min, +0.8V max	*
Current	±50μA max	*
Capacitance	5pF	*
Minimum Start Pulse		
R/C-Negative	120ns	*
Multiplexer Address	3-Bit Binary Address	*
	1LS TTL Load	*
SIGNAL DYNAMICS		
Conversion Time		
12-Bit Conversion	35μs max	*
8-Bit Conversion	25μs max	*
DIGITAL OUTPUTS		
Logic Outputs		
DB_{11}-DB_0, STS		
Logic 0	+0.4V max, I_{SINK} ≤ 1.6mA	*
Logic 1	+2.4V min, I_{SOURCE} <500μA	*
Leakage (High Z State)	±40μA (Data Bits Only)	*
Capacitance	5pF	*
Parallel Data		
Output Codes		
Unipolar	Positive True Binary	*
Bipolar	Positive True Offset Binary	*
POWER SUPPLY		
V_{LOGIC}	+4.5 to +5.5 Volts @ 9.7mA	*
V_{CC}	+13.5 to +16.5 Volts @ 30mA	*
V_{EE}	−13.5 to −16.5 Volts @ 32mA	*
Power Dissipation	1W typ, 1.3W max	*
Rejection[2]		
V_{LOGIC}	±0.002%/%	*
V_{CC}, V_{EE}	±0.005%/%	*
ACCURACY		
Linearity (% of F.S.R. max)	±0.025	±0.012
Offset[3]		
Unipolar (% of F.S.R. max)	±0.05	*
Bipolar (% of F.S.R. max)	±0.25	±0.1
Gain[3] (% to F.S.R. max)	±0.3	*
STABILITY		
Linearity (ppm/°C max)		
0°C to +70°C	±0.5	*
Unipolar Offset (ppm/°C max)		
0°C to +70°C	±10	±5
Bipolar Offset (ppm/°C max)		
0°C to +70°C	±10	±5
Gain (Scale Factor)(ppm/°C max)		
0°C to +70°C	±50	±20
TEMPERATURE RANGE		
Operating	0° to +70°C	*
Storage	−25°C to +85°C	*

NOTES:
1. Input signal must not exceed 10V which can cause an OFF channel to turn ON.
2. Maximum change over rated supply voltage
3. Externally adjustable to zero. See Applications Information.

*Specifications same as HS 9410J.

PACKAGE OUTLINE

Dimensions shown in inches and (mm).



Pin 1 marked by a dot on top of package

PIN ASSIGNMENTS

PIN	FUNCTION	PIN	FUNCTION
1	DB_{10}/DB_2	28	DB_9/DB_1
2	DB_{11}(MSB)/DB_3	27	DB_8/DB_0
3	A_0	26	DB_7
4	R/C	25	DB_6
5	GROUND	24	DB_5
6	V_{LOGIC}	23	DB_4
7	V_{EE}	22	STS(STATUS)
8	MUX ADDRESS A_2	21	GAIN
9	MUX ADDRESS A_1	20	OFFSET
10	MUX ADDRESS A_0	19	V_{CC}
11	INPUT CH 1	18	INPUT CH 5
12	INPUT CH 2	17	INPUT CH 6
13	INPUT CH 3	16	INPUT CH 7
14	INPUT CH 4	15	INPUT CH 8

ABSOLUTE MAXIMUM RATINGS

V_{CC} to Common GND ..... 0 to +16.5V
V_{EE} to Common GND ..... 0 to −16.5V
V_{LOGIC} to Common GND ..... 0 to +7V
Control Inputs (A_0, R/C) to Common GND ..... −0.5V to V_{LOGIC} +0.5V
Power Dissipation 1W
Lead Temperature, Soldering ..... 300°C, 10Sec
Maximum Input Voltage ..... V_{CC} +20V
Minimum Input Voltage ..... V_{EE} −20V

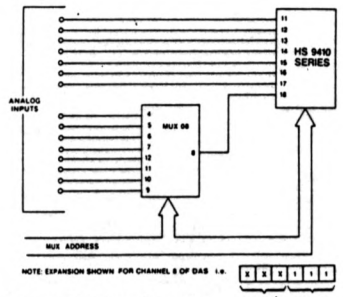
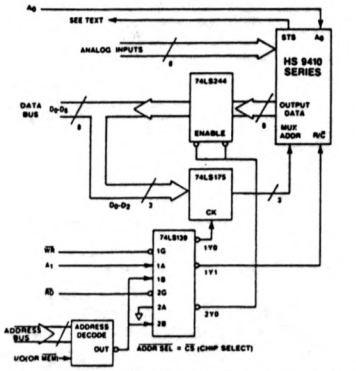


Figure 2. Multiplexer Expansion

MICROPROCESSOR INTERFACE

The HS 9410 Series DAS can be interfaced with popular 8-bit microprocessors. The DAS may be either positioned in a memory location (memory map) or as an I/O device. In the case of memory mapping, the DAS acts as a static RAM where READ and WRITE instructions are given to the selected address. When the DAS is connected as an I/O device, the I/O enable can be substituted for the $\overline{MEMR}$ or $\overline{MEMW}$ command. Figure 3 will show a typical scheme to implement this interface.

STS is not used in this example, the μP reads data approximately 35μs after conversion starts. This delay can be generated with NOP or other instructions inserted between the WRITE and READ functions. The STS line can also be used to cause the processor to WAIT or HALT or can be used as an interrupt line such as IREQ (in the case of 6800 or 6502).



A_0	A_1	$\overline{WR}$	$\overline{RD}$	$\overline{ADDR\ SEL}$	HS 9410 Function Read/Write	Operation
X	0	↓	1	0	WRITE	MUX ADDRESS
0	1	↓	1	0	WRITE	START 12-BIT CONV.
1	1	↓	1	0	WRITE	START 8-BIT CONV.
0	X	1	0	0	READ	HIGH BYTE (8 MSB's)
1	X	1	0	0	READ	LOW BYTE (4 LSB's)

NOTE:

1. 1 indicates logic HIGH. 2. 0 indicates logic LOW. 3. X indicates don't care.
4. ↓ indicates operation commences on low to high transition.
5. ↓ indicates operation commences on high to low transition.

Figure 3. Interfacing the HS 9410 Series

USING THE A_0 LINE

The state of the A_0 line at the start of a conversion places the DAS in either a full 12-bit conversion or in 8-bit 'short cycle' mode. During a READ at the end c conversion the A_0 line is used to the format of the dat as follows:

1. Prior to Conversion (WRITE)

	MODE
$A_0 = 1$	Short cycle 8-bit conver
$A_0 = 0$	Full 12-bit conversion

2. After Conversion (READ)

$A_0 = 1$	Data = Low Byte (LSB) followed by zeros
$A_0 = 0$	Data = High Byte (MSB) followed by middle byte

In a μP application the A_0 line can be considered a pa $\overline{W}/R$ locations as follows:

1. Prior to Conversion (WRITE)

	MODE
$\overline{W}/R = 0$ in low address ($A_0 = 0$)	Full 12-bit conversion
$\overline{W}/R = 0$ in high address ($A_0 = 1$)	Short cycle 8-bit conver

2. After Conversion (READ)

$\overline{W}/R = 1$ in high address ($A_0 = 1$)	LSB's & zeros
$\overline{W}/R = 1$ in low address ($A_0 = 0$)	8 MSB's only

POWER SUPPLY CONSIDERATION

Power supplies used for the DAS should be selected for low noise operation. In particular they should be free of high frequency noise. Unstable output codes may result with noisy power sources. It is important remember that 2.44mV is 1LSB for a 10 volt input.

Decoupling capacitors are recommended on all pov supply pins located as close to the converter as pos ble. Suitable decoupling capacitors are 10μF tantalu type in parallel with 0.1μF disc ceramic type.

GROUNDING CONSIDERATIONS

The common at pin 5 is the ground reference point the internal reference and is thus the high quality ground for the DAS. In order to achieve all of the hig accuracy performance available from the DAS in an environment of high digital noise content, care shou be taken when handling analog and digital grounds follows. Where analog and digital grounds are run separately on the PCB, these should be connected together at the package (pin 5). However, if the grounds are connected separately in the system for other reasons, then only the analog ground should connected at the package to pin 5. If digital commc contains high frequency noise beyond 200mV, this noise may feed through the converter, so that some caution will be required.

It is also important in the layout, to carefully conside the placement of digital lines. It is recommended th digital lines not be run directly under the DAS. For o timum system performance, if space permits, a grou plane is advised under the DAS. This should be cor nected to a digital ground. Finally, in packaging the assembled DAS, the designer should also try to mir mize any capacitive coupling that might occur at th top to the device.

CONTROL FUNCTIONS

The HS 9410 Series contains all control functions necessary to provide for complete microprocessor interface. All control functions are defined in Table 1, 2, and 3.

Function	Definition	Function
R/$\overline{C}$	Read/Convert	1. ↓ initiates conversion. 2. Low (0) disconnects data bus. 3. High (1) initiates read.
A_0	Device Address	1. Selects conversion mode. 12-bits if low (0). 8-bits if high(1). 2. In read mode A_0 selects the output format. If low (0) then 8 MSB's (high and middle byte) or if high (1) then only low byte and trailing zeroes.
MA_0 MA_1 MA_2	Multiplexer Address	Select Channels 1-8 (see MUX Logic Table 3)

Table 1. Defining the Control Functions

Control Inputs R/$\overline{C}$	Control Inputs A_0	Operation
↓	0	Initiates 12-bit conversion
↓	1	Initiates 8-bit conversion
1	0	Enables 8 MSB's (high byte)
1	1	Enables 4 LSB's (low byte) and 4 trailing zeros
0	X	Output data (DB) goes to high impedance state.

Table 2. Truth Table—Control Inputs

Mux Address Inputs MA_2	MA_1	MA_0	Channel Selected
0	0	0	1
0	0	1	2
0	1	0	3
0	1	1	4
1	0	0	5
1	0	1	6
1	1	0	7
1	1	1	8

NOTES:
1. 1 indicates logic HIGH.
2. 0 indicates logic LOW.
3. X indicates don't care.
4. ↓ indicates operation commences on high to low transition.
5. MSB → XXXX High Byte; XXXX Middle Byte
6. XXXX Low Byte

Table 3. Truth Table—Multiplexer Address

APPLICATIONS INFORMATION

TIMING

The timing diagrams are shown in Figure 1. Note that the MUX address must be set up prior to each convert command (R/$\overline{C}$ = ↓), not only does the R/$\overline{C}$ control the start of conversion, but when R/$\overline{C}$ = 0, the output goes to a high impedance state (high Z). This releases the data bus. As shown in the timing diagram, a second MUX channel may be selected while the ADC is converting the data on the first channel. This is sometimes referred to as the 'overlap' mode.

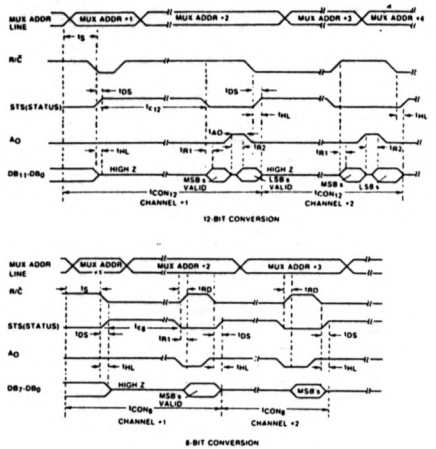


Figure 1. HS 9410 Series, Interface Timing

CONVERT CYCLE

Symbol	Parameter	
t_S	Multiplexer Address Setup time prior to a convert cycle	7μs typ, 10μs max
t_{R1}	Bus Access Time, 8 MSB's	70ns typ, 125ns max
t_{R2}	Bus Access Time, 4 LSB's	150ns typ, 200ns max
t_{C12}	ADC 12-Bit Conversion Time	30μs typ, 35μs max
t_{C8}	ADC 8-Bit Converstion Time	20μs typ, 25μs max
t_{AO}	Minimum A_0 Pulse Width	150ns + READ Time
t_{DS}	STS Delay from R/C	200ns max
t_{HL}	Output Float Delay	150ns max
t_{AO}	Minumum A_0	150ns + READ Time
$t_{CON_{12}}$	Total Channel Acquisition and Conversion Time, 12-Bits	35μs max
t_{CON_8}	Total Channel Acquisition and Conversion Time, 8-Bits	25μs max

INPUT EXPANSION

The DAS is configured with an 8 channel high level multiplexer input. This was done to optimize package size (28 pin DIP) and cost. In the event the user wishes to increase the number of input channels, a double rank MUX input is recommended (series connected). This typical configuration is shown in Figure 2.

ORDERING INFORMATION

Model Number[1]	Input Range	System Accuracy (% FSR)	Full Scale T.C. (ppm/°C)	Temp. Range	Screening
HS 94XXJ	SEE	±0.025	50.0	0°C to +70°C	0.4% AQL[2]
HS 94XXK	NOTE 1	±0.012	20.0	0°C to +70°C	0.4% AQL[2]

NOTES

1. HS 94XX

MODEL SUFFIX	INPUT RANGE
10	0 to +10V
11	±5V
12	±10V

Add letter suffix as required above.

2. **0.4% AQL.** All units have full burn-in at +85°C. Acceptable quality level (AQL) of 0.4% means that Hybrid Systems guarantees that there will be no rejects in a sample lot of 100 pieces. That's more than twice as tough as it has to be — even for military applications.

Specifications subject to change without notice.

Hybrid Systems CORPORATION

22 Linnell Circle
Billerica, MA 01821
TWX: 710-347-1575 Tel: 617/667-8700

OVERSEAS SALES OFFICES

FRANCE
Hybrid Systems SARL
14 Rue du Morvan
S.I.L.I.C. 525
F-94633 RUNGIS Cedex
Tel. (1) 687 8336

WEST GERMANY
Hybrid Systems GmbH
Rheinstrasse 32
D-6100 DARMSTADT
Tel: (06151) 29 15 95

UNITED KINGDOM
Hybrid Systems U.K. Ltd.
12A Park Street
CAMBERLEY, Surrey
Tel: (0276) 2 81 28

BULLETIN HS 9410 SERIES-12 83/Printed in U.S.A.

Specification: A550001N001 Date: Dec 11, 1984

Title: Monitor and Control Bus at VLBA Stations

Prepared by: B. G. Clark APPROVED By:_____

1.0 GENERAL DESCRIPTION

This specification describes the characteristics of a serial digital data bus for controlling and monitoring all equipment at a station in the VLBA. To avoid possible confusion, some terms will be defined here. A CONTROLLER is considered to the station computer and its interface to the bus. A MONITOR AND CONTROL INTERFACE, or simply INTERFACE, is something that connects a piece of equipment in a station to the bus. A DEVICE is such a piece of equipment: e.g., a front end or a local oscillator module might be a device.

The bus will consist of two logic signals, each on a shielded twisted pair, wired as a multi-drop party line. The signals are called Transmit Data (XMT) and Receive Data (RCV). There will be one Controller and numerous Interfaces connected to the bus. The Controller will be the only source of Transmit Data, and the Interfaces (one at a time) will be the sources of Receive Data. Data will be bit-serial at a rate of 57.6 kbaud and the transmissions will be byte asynchronous, each byte consisting of, in order, a start bit (binary 0), eight data bits (least significant bit first), one parity bit, and one stop bit (binary 1).

Detailed specifications of bus line characteristics, levels, timing tolerances, routing and other conventions are stated below. Unless otherwise specified, EIA standard RS-422 and RS-485 will be followed.

2.0 MESSAGE FORMAT AND SEMANTICS

Every message on the XMT line will be exactly five bytes long, with the bytes called SYN, Address High (ADH), Address Low (ADL), Control Data High (CDH), and Control Data Low (CDL). The control function bytes (SYN,ACK,DC1,DC2,NAK; see below) are transmitted in even parity; the data bytes (ADH,ADL,CDH,CDL,MOH,MOL) are transmitted in odd parity. The SYN byte indicates the beginning of a message, and is the only even bit parity byte on the XMT line (thus distinguishing it from all data bytes). It is followed by ADH. If the most significant bit of ADH is 1, then the message is a control message; otherwise it is a monitor request message. The remaining 15 bits of ADH/ADL form a binary address in the range of 0 through 32767.

Messages on the RCV line will be either a two byte command acknowledgement or a two or three byte monitor data response, as discussed below.

3.0 BUS PROTOCOLS

Each Interface must receive ADH and ADL of every message on the XMT line (that is, there must be no dead time during which an interface is not listening). Each Interface is assigned a block of contigous addresses to which it alone responds. The block may be any length, but it must be disjoint with the address blocks of all other Interfaces. The last few addresses of each block are dedicated to functions occuring within the Interface, as specified below.

If the address transmitted was within the assigned block of an Interface, then within 382

microseconds of the last bit (Stop) of ADL, that Device must begin to transmit a one byte acknowledge code (ACK) on the RCV line. If the message was a control message, then the Interface must also receive and store CDH and CDL, and within 382 microseconds of the end of CDL or 573 microseconds after the initial ACK was put on the line, whichever occurs later, it must begin to transmit a second acknowledge byte (DC1) on RCV. If the message was a monitor request, the (single) acknowledge byte must be followed by two bytes of monitor data obtained from the address specified by ADH/ADL, or by a negative acknowledgement if monitor data is unavailable from the device being controlled. The first acknowledge byte (ACK) will also function as a clear to send, granting the controller the right to begin its next message, and promising to yield the RCV line before it is needed for another interface's response. (The time available to it is at least 764 microseconds, but may be longer if the controller has not begun to transmit CDL on the XMT line at the time the interface begins to send ACK on the RCV line; the interface will have at least 573 microseconds after the end of transmission of CDL to disconnect from RCV. See figures for timing relationships.)

The Interface must check parity on all bytes received. If SYN, ADH or ADL has a parity error, the Interface shall not respond (just as if the address were outside its block), but shall increment an internal counter and look for the next valid SYN. If SYN, ADH and ADL have valid parity and a control message is specified, but CDH or CDL has a parity error , then the second acknowledge byte (DC1) must be replaced by a negative-acknowledge code (NAK), CDH/CDL must not be passed to other equipment and a second parity error counter shall be incremented. The values of both of these counters shall be assigned to monitor addresses. A second form of negative acknowledge byte (DC2) may also be returned if the interface is unable to complete its handshaking with the device to which it interfaces. This may occur (but does not necessarily occur) if the device is powered down or unplugged, and may also be useful to signal other conditions in which the state of the controlled device is not altered (for commands) or for which it is not meaningful to return monitor data.

The Controller may begin transmitting another message after a control message after the receipt of the acknowledge byte. The maximum-speed timing (in the sense that all transactions are limited by line speed) for a sequence of control messages and for a sequence of monitor requests is illustrated in Figure 1. The minimum speed timing (in the sense that the controller is responding as rapidly as permitted by this protocol, while the interfaces are responding as slowly as permitted by this protocol) is illustrated in Figure 2.

The Controller must also check parity on all bytes received on the RCV line. If an acknowledge byte has incorrect parity or an incorrect code, the Controller may note that there is a possible problem, and may take remedial action, but no particular action is specified. If no response byte is received within 382 microseconds plus two byte transmission times of sending ADL, then the Controller may again note a possible problem, and proceed to transmit the next message. If a parity error is detected on a monitor data byte, then both bytes of the monitor word shall be ignored.

4.0 BUS SIGNAL CHARACTERISTICS AND CONVENTIONS

It is expected that more than 32 Interfaces will be required at a station. In that case, the bus will be split into several lines with up to 32 drivers and/or receivers per line. Thus a RCV line would have up to 32 drivers and one receiver (the Controller), and a XMT line would have one driver (the Controller) and up to 32 receivers. At the Controller, each line shall have its own transmitter or receiver. Command messages shall be broadcast on all XMT lines; there shall not be any line selection based upon the presence of the addressed Interface on any given line.

These conventions shall be followed:

4.1 Transmission rate: 57.6 kbaud, including all framing and parity bits.

4.2 Transmission lines: #24 twisted pair, shielded, (roughly 100 ohms characteristic impedance), max length 500 feet, terminated with a 100 ohm resistor.

4.3 Drivers and receivers shall be bridged across lines with stubs less than 20 feet.

4.4 RCV drivers shall be tri-state, connected to the bus only when required to respond to a monitor request. Drivers shall maintain their high impedance state when the interface is unpowered.

4.5 Line HV safety: Clipping surge arrestors shall be used on the bus lines between the control building and the antenna. The surge arrestors shall be located at each end of the bus run and shall shunt the surge currents to a suitable ground.

4.6 Interfaces which service equipment subject to lightning-induced currents shall abe protected by high voltage isolators such as optical isolators. The isolators shall be interposed in the lines between the Interface and the Device. Examples of such Devices are the subreflector drive and the weather instruments.

4.7 Bus signals shall conform to EIA RS-422 and EIA RS-485; particular attention is called to the following items:

4.7.1 Mode - Differential transmission and reception, +-2 to +-6 volt signal range.

4.7.2 Drivers and receivers capable of operating in the presence of Common mode voltages of -7 to +12 volts.

4.7.3 No device damage due to line contention of two drivers.

4.7.4 Max driver output current, hi Z state - +- 100 ua.

4.7.5 Max driver output current, power off - +- 100ua.

4.7.6 Receiver input sensitivity - +- 200 mv, min.*

4.7.7 Receiver input resistance - 12 kohms, min.

4.7.8 Driver output signal - +- 1.5 V min. into a 54 ohm load.

4.7.9 No device damage due to loss of power on one or more drivers or receivers.

*(On the RCV line, which has numerous tri-state drivers, and which has a normal quiescent state in which none of them is connected, it may be necessary to raise the threshold in order to avoid spurious transitions in the presence of noise in this high impedance state. If needed, this will be done by biasing or by hysteresis, the particular method not yet having been chosen. It is not anticipated that any action need be taken at the interfaces.)

5.0 STANDARD CODES

The defined hexadecimal codes for special characters are given below. These bytes will be transmitted in even parity, so that they are truly unique and will never be encountered in data.

SYN	16
ACK	06

NAK	15
DC1	11
DC2	12

6.0 ADDRESS AND DATA CONVENTIONS

6.1 The last sixteen monitor addresses of an interface's block are reserved for internal functions of an interface; all interfaces must report the following information when a monitor request with one of these addresses is received:

Address	Value
BE-15 thru BE-10	(reserved for future use)
BE-9	Address of last control message received
BE-8	Control data for last control message received
BE-7	Address parity error counter, all messages
BE-6	Control data parity error counter, all messages
BE-5	Invalid SYN character counter
BE-4	Control data parity error counter, messages in block
BE-3	(reserved for special use by standard interface)
BE-2	Count of correctly received control messages
BE-1	Count of correctly received monitor data requests
BE-0	Address of beginning of block

where BE is the Block End address. When a control message is received with an address of one of the counters, that counter shall be loaded with the control data (normally zero, to reset the counter).

6.2 It is strongly recommended that each device devote at least one monitor address to identification information. This should include revision level of the device, especially if it affects the required control word formats or the meaning of monitor data. The designer must decide at what level of complexity to specify this information (e.g., circuit board, module, subsystem). It is recognized that a subsystem may use more than one interface, and that an interface may service two or more logically separate devices.

6.3 It is also strongly recommended that logically distinct functions not be mixed within a single control word, even if this means that only a few bits of each word are used.

6.4 When a monitor word is used to convey status information, the syntax for "normal" status should include at least one bit set to logical 1 and at least one bit set to logical 0. This avoids having certain failure states (where all bits appear the same) interpreted as "normal."

6.5 It is recommended that each address used by a device for control messages have a corresponding monitor address (preferably the same address) on which the last control data received may be read back. It is also helpful if monitor data which represents the state of a device has a format similar to that of the control data which set its state (i.e., corresponding bits should have the same meaning). It is strongly suggested that distinct addresses be used for monitor and control functions, except for the read back function mentioned above.

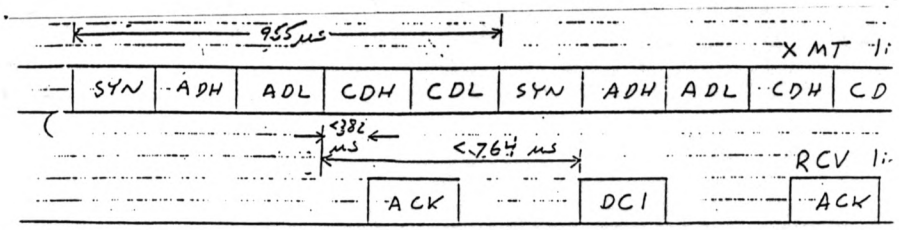


Fig 1A - Maximum Rate command sending.

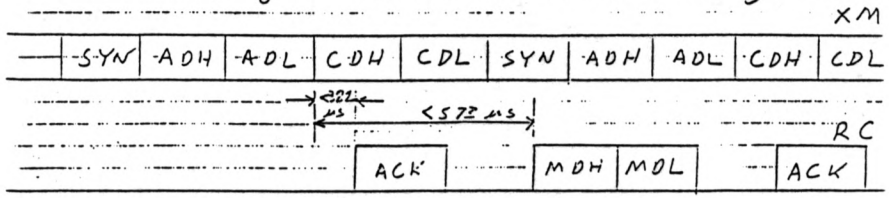


Fig 1B - Maximum Rate monitoring

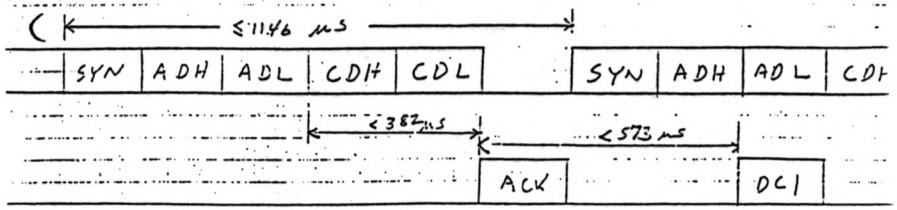


Fig. 2A - Minimum rate command sending

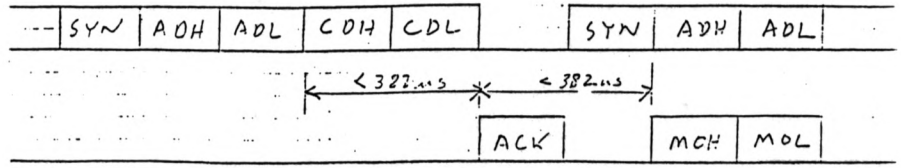


Figure 2B - Minimum rate monitoring

SPECIFICATION: A55001N002-A DATE: 25 NOVEMBER, 1985

TITLE: Monitor and Control Standard Interface

PREPARED BY: L. R. D'Addario APPROVED BY:_____

Revision History:
850510 Original Issue, by D. Weber and B. Clark
851125 Revision A, by L. D'Addario: added ID REQ line and description of its use. Revised assignments for internal address block. Revised power supply requirements. Changed A/D conversion gain from 5 Mv/bit to 4.8828 mV/bit.

1. GENERAL DESCRIPTION

This specification describes the characteristics of the Standard Interface between the VLBA serial digital command/data bus and the devices to be controlled. This document assumes familiarity with the Monitor and Control Bus Specification, A55001N001.

In order to keep the interfaces between the VLBA Monitor and Control Bus and the various controlled devices as similar as possible, a Standard Interface described in this specification shall be used whenever possible. The Standard Interface is a microcumputer-based serial to parallel converter which connects on one (serial) side to the two bus signals, and on the other (parallel) side to specialized equipment (referred to as here as "devices"). A block diagram of the Standard Interface is shown in Figure 1. The microprocessor will be a member of the Intel MCS 51 family or a compatible device.

2. INTERFACE LINES

The connections on the serial side consist of Transmit Data and Receive Data, as described in A55001N001. The complete set of serial, parallel and power connections are:

2.1 Parallel Signals (Connector P1)

INTERFACE I/O			
Relative Address	(RA-x)*	8 lines	out
Read/not Write	(R/-W)	1 line	out
Control/Monitor Data	(CON/MON)#	16 lines	out/in
Device Request	(DEV REQ)	1 line	out
Device Acknowledge	(DEV ACK)	1 line	in
Analog Signals	(ANLG-yz)**	8 pairs	in
External Analog Mux Enable	(ANENB)	1 line	in
HI/Lo Select	(HI/LO SEL)	1 line	in
POWER			
+/- 15 Common	(HIQ GND)	1 line	in
+15 Volts	(+15V)	1 line	in
-15 Volts	(-15V)	1 line	in
+5 Volts	(+5V)	1 line	in
5 Volt Common	(5V COMM)	1 line	in

* x denotes address lines 0,1,2,....7, bit 7 is msb
# n denotes bits 0,1,.....15, bit 15 is msb
** y denotes signal 0,1,2,3.....7, z denotes H (hi) and L (lo) signal polarity
In the "S" version the ANLG-yL signals are connected together and to HIQ GND on the interface board.
CON/MON are bi-directional tri-state lines used for command and monitor data argument transfer.

2.2 Serial Bus Signals, Activity Indicators, Redundant Power, ID Request (Connector P2):

BUS SIGNALS			
Transmit Data	(XMT- +/-)	1 pair	in
Receive Data	(RCV- +/-)	1 pair	out
ACTIVITY INDICATORS			
XMT Line Active	(XACT)	1 line	out
Message Active	(MSG)	1 line	out
Interface Active	(BUSY)	1 line	out
Data Output	(DOUT)	1 line	out
XMT Parity	(PARX)	1 line	out
REDUNDANT POWER			
+-15 Common	(HIQ GND)	1 line	out
+15 Volts	(+15V)	1 line	in
-15 Volts	(-15V)	1 line	in
+5 Volts	(+5V)	1 line	in
5 volt common	(5V COM)	1 line	in
ID Request	(ID REQ)	1 line	in

2.3 Notes

Redundant power is available on P2 to enable the Standard Interface to operate with P1 disconnected. P1 and P2 are "D" series connectors; details of connector types and pin assignments are given in Section 12.

3.0 POWER REQUIREMENTS

The worst case power demand from each power supply is:

	+5 V	+15 V	-15 V
1) Model S	700	30	32 ma
2) Model D	700	30	60
3) Either model with analog chips not installed	650	0	0

4.0 OPERATION

Upon receipt of a message on the XMT line with a valid address in the defined block, the difference between the address and the first address of the block, the Relative Address, is placed on RA. If a monitor request was received, R/-W is set true; if a control message, then R/-W is set false and CON/MON is set to CDH, CDL. When all output lines have been set, DEV REQ is set true. The device must now respond. For a control message, it must accept the data on the CON/MON lines and route it as specified by RA. In the case of a monitor request for digital data, it must place the data from the device specified by RA on the CON/MON lines. In either case, when it is ready it must set DEV ACK true. When DEV ACK becomes true, the interface will latch the data from the CON/MON lines (if appropriate) and then set DEV REQ to false. In the case of a monitor request for analog data, the device shall respond with ANENB instead of DEV ACK.

The time available for the device to respond is variable up to certain time-out limits as follows:

For control messages, the value on CON/MON will be held until either DEV ACK is sent by the device or a 500 microsecond time-out occurs. In the event that this time-out occurs, DEV REQ will be set to false and the CON/MON lines shall revert to the tri-state disconnect. Occurance of the time-out will be regarded as a device failure, to be handled as described under Command Timing, below.

Monitor data must be available within 500 microseconds from the time that DEV REQ goes true. If RA designates an analog data address, ANENB must be set true by the device logic within 5 microseconds from the time that DEV REQ goes true. The Standard Interface provides about 200 microseconds analog multiplexer settling time before the analog signal is sampled for A/D conversion; the conversion operation is completed by about 460 microseconds. Therefore, the analog signal must be present and stable at the appropriate pins of P1 within 200 microseconds of DEV REQ becoming true. In the event that RA designates digital monitor data, the device must respond with DEV ACK within 500 microseconds. If the device fails to respond in these times, the situation will be regarded as a device failure, to be handled as described under Monitor Timing, below.

The 8-bit RA capacity enables to Standard Interface to service a device having up to 256 command addresses and up to 256 monitor addresses in addition to the 16 reserved at the end of the block.

The address block of the interface shall, at the time power is applied, be initialized to 7FF0 to 7FFF hexadecimal. In addition, the interface shall respond to addresses 2N and 2N+1, where N is a 7-bit number called the ID byte; it is supplied by external circuitry, as described below. A monitor request to address 2N shall return the current length of the address block (initially 16), and a monitor request to address 2N+1 shall return the current starting address of the block (initially 7FF0 hex). A control message to each of these addresses shall reset the corresponding value. Each interface connected to the bus must have a unique value of N.

The station computer shall, upon initialization and periodically thereafter, send monitor requests to addresses 2N and 2N+1 for all N on which interfaces are expected. If necessary, it shall send control messages to reset the address block of each interface to its assigned range. It will also occasionally try to read an interface address block at address 7FFF hex.

The Standard Interface will determine its ID byte by setting output line ID Request (ID REQ) to LOW (true). This line will be capable of driving one LS load. Whenever this line is true, an external circuit must place the ID byte on lines CON/MON-0 through CON/MON-6 and must set line CON/MON-7 so that CON/MON-0:7 will have odd parity. The lines must be stable within 10 microseconds of the time that ID REQ becomes true, and the data lines must revert to tri-state disconnect within 20 microseconds after ID REQ becomes false. It is suggested that this be implemented by using ID REQ to enable a tri-state buffer. The eight bits may be wired onto the inputs of this buffer, or the ID byte may be set by switches (in which case a parity generation chip is recommended), or it may be determined from wiring to a module plug (in which case and even number of lines should be brought through the plug and they should have odd parity when correctly set). The Standard Interface will set ID REQ during its power up sequence and periodically thereafter. None of the other handshaking lines will be altered during this sequence; in particular, DEV REQ will not be set.

The Standard Interface shall check for correct parity of the ID byte. If the parity is incorrect, it shall not respond to messages addressed to 2N or 2N+1, but only those in its assigned or default block. In any case, it shall report the value of the ID byte last read, including parity whether or not correct, in response to a monitor request to address BE-3, where BE is the address at the end of the block.

5. ADDRESS UTILIZATION

The following bus addresses are reserved for standard interfaces, and may not be assigned to other interfaces:

0000-00FF: used for address pointers, two addresses per standard interface.

7FF0-7FFF: default address block of a standard interface whose assigned block has not yet been

set by the computer, or whose address block cannot be set because of a fault.

The last 16 addresses of the assigned block of a standard interface (and all 16 addresses of the default block) are for internal use of the interface. Some of these are assigned specific functions for all interfaces (see specification A55001N001), and others are here assigned functions for the standard interface. The complete list (where BE is the block end address) is :

Address	Value
BE-15 thru BE-13	(reserved for future use)
BE-12*	No control response counter (see Section 9.1)
BE-11*	No monitor response counter (see Section 9.2)
BE-10	Interface type and revision code
BE-9	Address of last control message received
BE-8	Control data for last control message received
BE-7	Address parity error counter, all messages
BE-6	Control data parity error counter, all messages
BE-5	Invalid SYN character counter
BE-4	Control data parity error counter, messages in block
BE-3*	ID byte from device (see Section 3)
BE-2	Count of correctly received control messages
BE-1	Count of corectly received monitor data requests
BE-0	Address of beginning of block

* These assignments are specific to the standard interface.

6. ACTIVITY INDICATORS

Five time-stretched LED drive lines shall be provided for front panel indication of interface activity. These lines shall be capable of sinking 8ma (current limiting resistors will be provided in the interface). (THIS STATEMENT IS INCORRECT, CURRENT LIMITING LINES ARE NOT INCLUDED IN THE INTERFACE, THEY MUST BE PROVIDED BY THE DEVICE CIRCUITRY). These lines indicate:

1) XACT - XMT line is active, i.e. signal transitions are present on the XMT line. In conjunction with 2), this permits detection of garbled messages on the XMT line.

2) MSG - Message active, i.e. valid messages are being detected on the XMT line.

3) BUSY - The interface has detected a message within its address block.

4) DOUT - The interface is transmitting monitor data on the RCV line.

5) PARX - A parity error has been detected in a message on the XMT line.

7. DIGITAL SIGNAL DRIVE/LOAD SPECIFICATIONS

These specifications apply to all digital signals at the device interface, except the ID REQ line.

Signal sense levels: positive true, all lines
Signal outputs: +2.7 volts min = logic 1
+0.7 volts max = logic 0

Logic inputs: +2.0 volts min = logic 1
+0.7 volts max = logic 0

Signal drive/loading: all outputs shall be capable of driving 20 standard 74LS loads. All inputs

shall present no more than 1 standard LS load.

Tri-state (disconnect) output current: +-20 ua.

Quiescent (between message) states:*
RA - static at most recent RA state
CON/MON - Tri-state
R/-W - logic 0 (write state)
DEV REQ - logic 0
DEV ACK - must be logic 0
*Quiescent state may end as much as 200 us before DEV REQ is set.

8. ANALOG DATA CAPABILITIES

8.1 General

The Standard Interface shall have an optional analog to digital converter and 8 channel analog multiplexer. In applications where this feature is used, the lower portion of the RA space is reserved for this analog data. Analog data signals connected to any of these 8 inputs shall be multiplexed, sampled and converted if the RA designated by a monitor data request falls in the RA space reserved for analog data.

In the event that the Interface is not used to gather analog data, the A/D Converter and Multiplexer need not be installed and the Analog Enable (ANENB) to the Interface shall be logic 0 (logic ground). In this case, the entire RA space is available for digital monitor data.

The A/D quantization shall be 5 mv/bit, 2's complement format and the analog signal range is +10 to -10 volts. Data will be returned in the most-significant 12 bits of the 16-bit monitor data word.

8.2 Sub-Multiplexing

In the event that more than 8 analog data signals need to be multiplexed and converted, remote multiplexers in the device may be connected to the board analog inputs to provide additional channels. Eight channel multiplexers are recommended, which allows extending the capacity to 64 channels.

The following paragraph describes the recommended scheme for providing capacities of 8 through 64 channels. Other schemes are possible, including extentions beyond 64 channels, within the constraints of the on-board multiplexing.

The first (8 channel) multiplexer is connected to ANLG-1. If more analog multiplexing is required, the second is connected to ANLG-2. Additional multiplexers may be added in this ascending sequence as required. The table below details these connections. When an external multiplexer is being used, the lower order RA bits control its channel selection and the higher order RA bits control the selection of the external multiplexer via the on-board multiplexer. The logic terms HI/LO SEL and ANENB, which are further described below, must be generated by the device from the current RA. The fact that the channel capacity does not increase in groups of 8 is a consequence of the use of the on-board multiplexer in conjunction with the external multiplexers. For example: if an 8-channel multiplexer is connected to the ANLG-1 input, the remaining 7 on-board analog inputs (ANLG-1,ANLG-2 through ANLG-7) may be used. Thus a multiplexing capacity of 15 channels is realized by the addition of just one 8-channel multiplexer. If a second 8 channel multiplexer is connected to ANLG-2, the remaining 6 on-board analog inputs may be used which provides a multiplexing capacity of 22 channels. The table below indicates the resultant channel capacity for all cases.

Table 8.2.1: Recommended Sub-Multiplexing Schemes

EXT MUX	CONNECT TO:	HI/LO SEL*	ANENB DECODE*	MUX CAPACITY
NONE	--	ALWAYS LOW	0 <= RA <= 7	8 CHANNELS
1ST	ANLG-1	10 <= RA <= 17	0 <= RA <= 17	15 CHANNELS
2ND	ANLG-2	10 <= RA <= 27	0 <= RA <= 27	22 CHANNELS
3RD	ANLG-3	10 <= RA <= 37	0 <= RA <= 37	29 CHANNELS
4TH	ANLG-4	10 <= RA <= 47	0 <= RA <= 47	36 CHANNELS
5TH	ANLG-5	10 <= RA <= 57	0 <= RA <= 57	43 CHANNELS
6TH	ANLG-6	10 <= RA <= 67	0 <= RA <= 67	50 CHANNELS
7TH	ANLG-7	10 <= RA <= 77	0 <= RA <= 77	57 CHANNELS
8TH	ANLG-0	ALWAYS HIGH	0 <= RA <= 77	64 CHANNELS

* Addresses in octal code, high true

Whether or not the recommended scheme is used, the device must generate ANENB and HI/LO SEL. HI/LO SEL is used by the interface to select either the lowest 3 RA bits (RA:0,1,2) or the next 3 higher (RA:3,4,5) for control of the on-board multiplexer. These multiplexer control terms must be switched as a function of the current RA so that the remaining on-board (i.e. those not connected to external multiplexers) multiplexer inputs can be used in conjunction with the external multiplexers. If only the on-board multiplexer is used, the HI/LO SEL line must be grounded. The ANENB signal must be a logic product of R/-W, DEV REQ and the inclusive decode of RA over the address space dedicated to analog data.

In the event that the Monitor Data request was for analog data, the device shall not set DEV ACK true.

From settling time considerations, tandem connections of external multiplexers beyond one tier are not recommended. The entire multiplexer chain may be required to settle to 0.05% accuracy within 200 us after the DEV REQ becomes true. External multiplexers must have break-before-make switching characteristics.

The character of the data being gathered (i.e. analog data via internal or external multiplexers-A/D versus digital data on the CON/MON lines) shall be transparent to the Standard Interface in that its firmware shall not depend upon whether the data was input via the A/D or via CON/MON.

Details on interfacing to typical controlled devices will be presented in a "Standard Interface User's Guide" (in preparation).

8.3 A/D Versions

Two versions of the Standard Interface are available which differ only in the type of analog multiplexer used. These area:

VERSION S - which uses a low-cost, single-ended 8 channel multiplexer - A/D Converter. This version is recommended for applications in which common-mode noise effects are minimal. An example of this application is installation of the Standard Interface board in a module in which all analog data is generated within the module. A typical example might be an IF Processor module.

VERSION D - which uses a differential 8 channel analog multiplexer - A/D Converter for applications where common-mode noise is more likely to be a problem. Examples of this appplication are cases in which analog signals to be sampled and converted originate from sources with ground references different from the Interface ground reference or situations in which signals are noise-contaminated by cable or wire runs. An example might be a Front End control module monitoring voltages within the Dewar, a nearby

amplifier, etc.

Both versions have identical connector pin assignments.

8.3.1 Version S Analog Signal Conversion Specifications

Analog data acquisition chips: Hybrid Systems Inc. HS 9412 Data Acquisition System, or equivalent, or better.

1) 8 input signals. single-ended, max signal voltage:
-11 volts < Vin < +11 volts. Analog signals shall not exceed these bounds or inter-channel cross- talk may occur.

2) Conversion signal range: -10.000 to + 10.000 v (4.8828 mv/bit)

3) Input bias current: +- 30 na max per channel

4) Input Impedance:
On channel: 100 ohms, shunted by 250 pf
Off channel: 100 ohms, shunted by 100 pf

5) Cross-talk between channels: < 80 db

6) Resolution: 12 bits

7) Gain error: < 0.3% FSR, adjustable to zero via an on-board pot; adjustment range +- 13 lsb

8) Offset error: < 0.25% FSR, adjustable to zero via an on-board pot; adjustment range +- 20 lsb

9) Temperature coefficients:
gain: +- 50ppm/deg C
offset: +- 10 ppm/deg C
linearity: +- 0.5 ppm/deg C

10) Analog signal settling time before S/H acquisition: 200 us

11) Sample/Hold acquistion time: 10 us

12) Multiplexer switching transitions are break-before-make with a make delay of 1 microsecond

8.3.2 Version D Analog Signal Conversion Specifications

Analog data acquisition chips: Burr-Brown SDM 854BG Data Acquisition System and Burr-Brown INA101 Instrumentation Amplifier, or equivalents, or better.

1) 8 input signals, differential, max signal voltage (each line): -15 volts < Vin < +15 volts. Analog input signals must not exceed these bounds or device damage may occur.

2) Conversion signal range: -10.000 to + 10.000 V (4.8828 mv/bit)

3) Input bias current: +- 50 na max per channel

4) Input Impedance:
on channel: 100 ohms shunted by 250 pf
off channel: 100 ohms shunted by 210 pf

5) Common Mode Rejection: 70 db min @ 1 khz, 20 volts P-P common mode signal, 1000 ohms source impedance

6) Cross-talk between channels: any off channel to any on channel, < 80 db @ 1khz, 20 volts P-P off signal

7) Resolution: 12 bits

8) Gain error: < 0.05% FSR, adjustable to zero via an on-board pot.
adjustment range +-1%

9) Offset error: < 10 mv, adjustable to zero via an on-board pot.
adjustment range +-4 lsb

10) Linearity error: +-1/2 lsb, max

11) Differential linearity error: +-1 lsb, max

12) Relative accuracy: +-0.025% of FSR

13) Noise error: < 1 mv P-P, 0 to 1 khz

14) Temperature coefficients:
gain: +-30 ppm/deg C, max
offset: +-30 ppm/deg C, max
differential linearity: no missing codes over the 0 deg to 70 deg C range

15) Analog signal settling time before S/H acquisition: 200 us

16) Sample/Hold acquisition time: 18 us

17) S/H feed through: < 1.4 mv

18) Multiplexer switching transitions are break-before-make with a make delay of 1 microsecond

9. TIMING

Times stated below are firmware-dependant. Although "typical" times given are approximate, times quoted as limits will be maintained.

9.1 Control Message Timing

This section applies when a control message was detected for any address within the address block of the interface.

Signals: RA, R/-W, DEV REQ, CON/MON, DEV ACK.

1) RA is set at least 5 us (typically 140 us) and at most 200 us before DEV REQ is set.

2)CON/MON lines are set at least 300 ns before DEV REQ goes true. (R/-W is in the low state during quiescent times and remains so during command execution.)

3) If DEV ACK from the device goes true within 500 us after DEV REQ goes true, the Interface interprets this to mean that the device has decoded the RA and read the command argument. DEV ACK must be held true until DEV REQ becomes false, which will be no more than 100 us and no less than 1 us (typically 50 us) after DEV ACK goes true. DEV ACK must then become false within 5 us after DEV REQ. CON/MON will revert to the tri-state disconnect no sooner than 100 ns (typically 1 us) after DEV REQ goes false. R/-W will remain in the logic 0 (write) state when DEV REQ goes false.

4) If DEV ACK has not been made true until after 500 us, the Interface assumes that the device was unable to respond. In this case DEV REQ will go false typically 520 us after it was raised and CON/MON will revert to the tri-state disconnect. R/-W will remain in the logic 0 (write) state when DEV REQ goes false. This NO RESPONSE condition will increment a counter called No Control Response. The contents of this counter shall be returned in response to a monitor request to address BE-11.

5) RA stays at the value for this message until the next message is received.

9.2 Monitor Request Timing, Digital Data

This section applies when a monitor request was detected and the address was within the portion of the address block dedicated to digital monitor data.

Signals: RA, R/-W, DEV REQ, CON/MON, DEV ACK.

1) RA is set at least 5 us (typically 140 us) and at most 200 us before DEV REQ is set.

2) R/-W is set high at least 200 ns (typically 1 us) before DEV REQ goes true.

3) DEV ACK must go true within 500 us after DEV REQ goes true. This signals the Interface that monitor data is available and stable on the CON/MON lines. No sooner than 100 ns (typically 1 us) after DEV ACK goes true, the Interface will sample the data. DEV ACK must be held true until DEV REQ goes false.

4) DEV REQ will be set false no more than 100 us and at least 1 us (typically 50 us) after DEV ACK went true. DEV ACK must revert to the false state within 5 us after DEV REQ went false.

5) In the event that DEV ACK does not go true within 500 us after DEV REQ went true, a negative acknowledgement will be sent from the Interface to the controller, and an internal counter called No Monitor Response shall be incremented; the contents of this counter shall be returned in response to a monitor request to address BE-10.

6) The CON/MON lines must revert to the tri-state disconnect within 5 us of the time that DEV REQ goes false.

7) RA stays at the value for this message until the next message is received. R/-W will be set false no less than 100 ns (typically 1 us) after DEV REQ goes false.

9.3 Monitor Request Timing, Analog Data

This section applies when a monitor request was detected and the address was within the portion of the address block dedicated to analog data.

Signals: RA, R/-W, DEV REQ, ANENB, HI/LO SEL.

1) RA is set at least 5 us (typically 140 us) and at most 200 us before DEV REQ is set.

2) R/-W is set high at least 200 ns (typically 1 us) before DEV REQ goes true.

3) ANENB must go true withi 5 us after DEV REQ is set. RA must be connected to external multiplexers (if any) so that the analog signals may begin to settle, and HI/LO SEL must be set to the correct state.

4) The selected analog signal will be sampled and A/D conversion initiated approximatly 240 us after ANENB goes true.

5) DEV REQ will go false at most 400 us (typically 250 us) after ANENB is set. ANENB must revert to false within 5 us after DEV REQ becomes false. After the A/D conversion the Interface will read and format the data.

6) If ANENB is not set within 5 us of DEV REQ, a negative acknowledgement will be returned to the controller and the No Monitor Response counter will be incremented. DEV REQ will be set false typically 500 us after it was initially set true. Note that it is not possible for the interface to distinguish between analog non-response and digital non-response.

7) The device must not attempt to drive the CON/MON lines during the processing of an analog data request.

8) RA stays at the value for this message until the next message is received. R/-W will be set false no less than 100 ns (typically 1 us) after DEV REQ goes false.

10. ADDITION OF USER MICROPROGRAMMING TO THE STANDARD INTERFACE

The microprocessor in the Standard Interface may be used for special applications provided that the command and monitor data capabilities and timings described in this specification adr not compromised.

Users may provide any or all of the following subroutines, which will then replace dummy subroutines in the standard firmware.

10.1 The command processor subroutine

This subroutine will be called whenever a control message within the interface's address block is received. Upon entry, the RA (Relative Address) will be pushed onto the processor stack below the subroutine return address and the control data will be in the AB register. If, on return, the F0 flag is reset, the data from the AB registers will be output on the MON/CON lines as usual. If the F0 flag is set, no further processing of the command message will take place. This subroutine is called during interrupt servicing, and so must complete operations and return within 250 us (approximatly 200 instructions) in order to be ready for the next message.

10.2 The monitor processor subroutine

This subroutine will be called whenever a monitor request message within the interface's address block is received. Upon entry, the RA will be pushed onto the processor's stack. The routine may reset the F0 flag to indicate no interest in the RA, or return with F0 flag set and a computed monitor value to be

placed on the RCV bus in the AB register. This routine also executes during interrupt servicing, and must return within 100 us if the F0 flag is reset or within 250 us if a value is returned.

10.3 Background task

This routine will be called when no message is being processed. It has no restrictions on timing but is restricted in what it may do to the device interface: for instance, if it is to set DEV REQ it must first set R/-W, and the device logic must be constructed to tolerate the unexpected dropping of DEV REQ when a command or monitor request arrives while the background has it set. The background task will be restarted at its entry point after any message to this interface is processed. The receipt of messages not addressed to this interface (outside its address block) will not affect the background, except for loss of CPU time for processing.

11. PHYSICAL DESCRIPTION

Figure 2 depicts the Standard Interface Board package, physical envelope, mounting hole locations and connector orientations. The board envelope is 6.25(H) x 5.5(W) x .75(T) inches including I/O connectors. The board may be mounted on the NRAO module rails or on standoff spacers. Up to 0.25 inches may be trimmed from each side of the 6.25 dimension for installation in an rfi shielded enclosure or mounting between the rails. I/O connectors are compact "D" series connectors mounted on one edge of the board.

12. CONNECTOR DETAILS

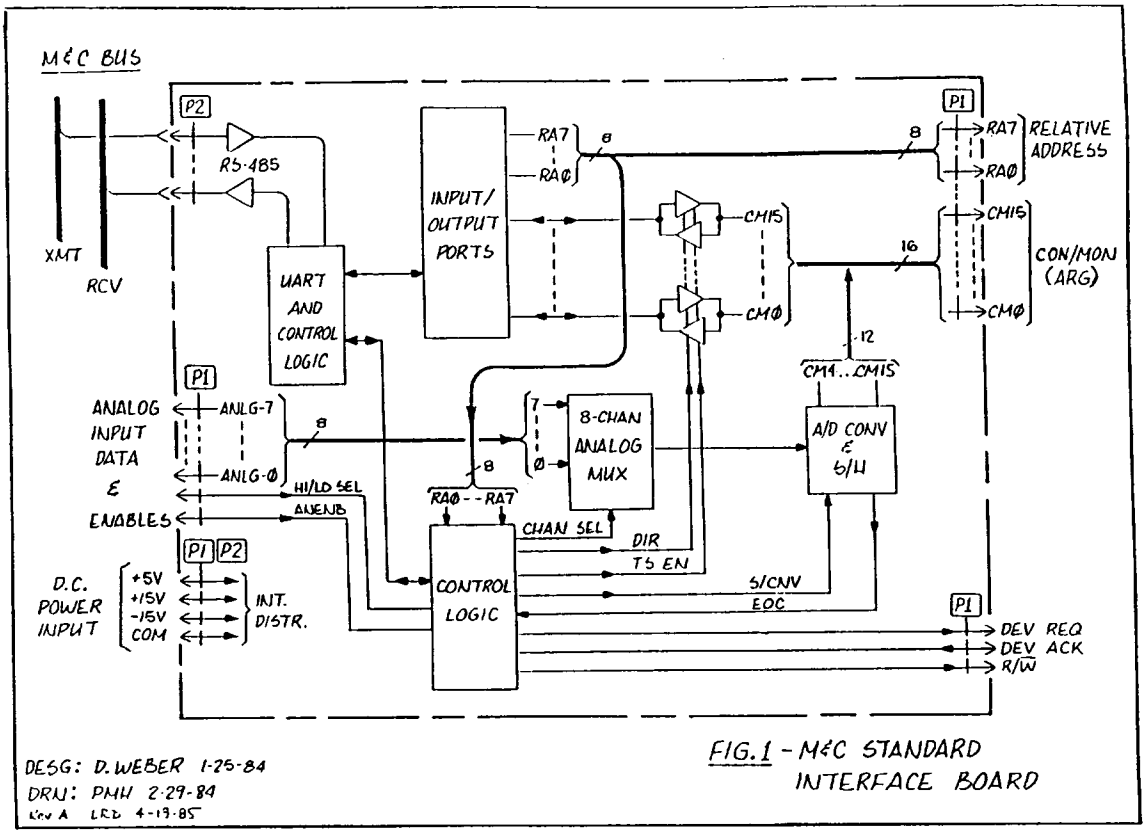
Both versions have identical connections to P1 and P2 so that either board may be plugged into a given interface. On Version *S* (single-ended analog multiplexing/conversion), the ANLG-0L through ANLG-7L (analog signal low connections) are tied to HIQ GND. Detailed information about each connector is given in the following tables.

Parallel I/O - P1
Type: Cinch or Amphenol DD-50PC
Mating type: DD-50S

Pin	Signal
1	ANLG-0H
2	ANLG-1H
3	ANLG-2H
4	ANLG-3H
5	ANLG-4H
6	ANLG-5H
7	ANLG-6H
8	ANLG-7H
9	CON/MON-15
10	CON/MON-13
11	CON/MON-11
12	CON/MON-9
13	CON/MON-7
14	CON/MON-5
15	CON/MON-3
16	CON/MON-1
17	HI/LO SEL
18	ANLG-0L
19	ANLG-1L
20	ANLG-2L
21	ANLG-3L
22	ANLG-4L
23	ANLG-5L
24	ANLG-6L
25	ANLG-7L
26	CON/MON-14
27	CON/MON-12
28	CON/MON-10
29	CON/MON-8
30	CON/MON-6
31	CON/MON-4
32	CON/MON-2
33	CON/MON-0
34	5V COMM
35	DEV REQ
36	DEV ACK
37	AMENB
38	HIQ GND
39	-15V
40	+15V
41	RA-7
42	RA-6
43	RA-5
44	RA-4
45	RA-3
46	RA-2
47	RA-1
48	RA-0
49	R/-W
50	+5V

Serial I/O - P2
Type: Cinch or Amphenol DB-25PC
Mating type: DB-25S

Pin	Signal
1	+5V
2	(reserved)
3	(reserved)
4	(not used)
5	(reserved)
6	(reserved)
7	(reserved)
8	(reserved)
9	ID REQ
10	DOUT
11	PARX
12	MSG
13	5V COMM
14	+5V
15	+15V
16	-15V
17	HIQ GND
18	XACT
19	RCV+
20	RCV-
21	XMT+
22	XMT-
23	(reserved)
24	BUSY
25	5V COMM



DESG: D. WEBER 1-25-84
DRN: PMH 2-29-84
Rev A LRD 4-19-85

FIG. 1 - M&C STANDARD INTERFACE BOARD

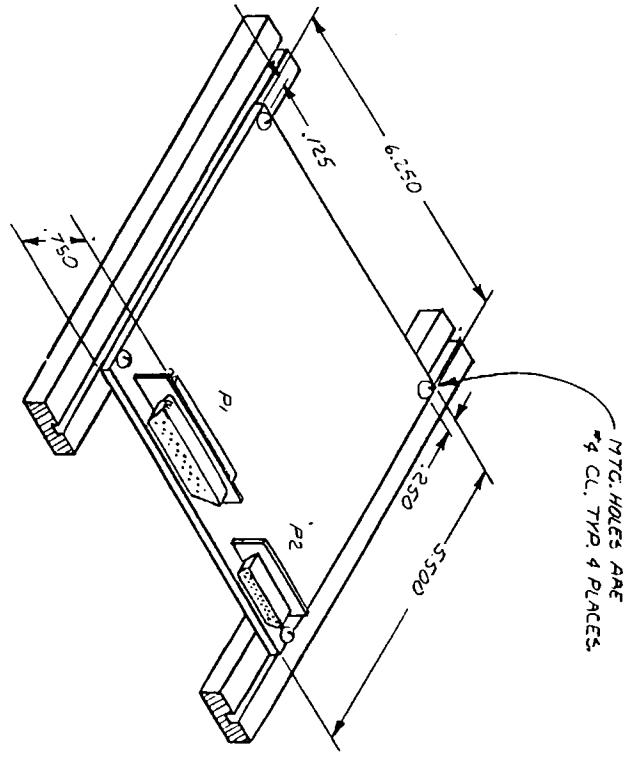


FIG. 2
STANDARD INTERFACE BOARD PACKAGE
THIS IS DRAWN AT HALF SCALE.

EIA STANDARD

STANDARD FOR
ELECTRICAL CHARACTERISTICS
OF
GENERATORS AND RECEIVERS
FOR USE IN
BALANCED DIGITAL MULTIPOINT SYSTEMS

RS-485

APRIL 1983

Engineering Department

ELECTRONIC INDUSTRIES ASSOCIATION

NOTICE

EIA Engineering Standards and Publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for his particular need. Existence of such Standards and Publications shall not in any respect preclude any member or non-member of EIA from manufacturing or selling products not conforming to such Standards and Publications, nor shall the existence of such Standards and Publications preclude their voluntary use by those other than EIA members, whether the standard is to be used either domestically or internationally.

Recommended Standards and Publications are adopted by EIA without regard to whether or not their adoption may involve patents on articles, materials, or processes. By such action, EIA does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the Recommended Standard or Publication.

This EIA Recommended Standard is considered to have international standardization implications, but the IEC (or ISO) activity has not progressed to the point where a valid comparison between the EIA Recommended Standard and the IEC (or ISO) Recommendation can be made.

Published by

ELECTRONIC INDUSTRIES ASSOCIATION
Engineering Department
2001 Eye Street, N.W.
Washington, D.C. 20006

PRICE: $8.00

Printed in U.S.A.

EIA RECOMMENDED STANDARD RS-485

STANDARD FOR ELECTRICAL CHARACTERISTICS OF GENERATORS AND RECEIVERS FOR USE IN BALANCED DIGITAL MULTIPOINT SYSTEMS

Prepared by

EIA TR-30.1 Subcommittee on Signal Quality

TABLE OF CONTENTS

LIST OF FIGURES

STANDARD FOR ELECTRICAL CHARACTERISTICS OF GENERATORS AND RECEIVERS FOR USE IN BALANCED DIGITAL MULTIPOINT SYSTEMS

(From Standards Proposal No. 1488, formulated under the cognizance of Subcommittee TR-30.1 on Signal Quality.)

1. SCOPE

This standard specifies the electrical characteristics of generators and receivers that may be employed when specified for the interchange of binary signals in multipoint interconnection of digital equipments. When implemented within the guideline of this standard, multiple generators and receivers may be attached to a common interconnecting cable.

An interchange system includes one or more generators connected by a balanced interconnecting cable to one or more receivers and terminating resistors. The electrical characteristics of the circuit are specified in terms of the required voltage, current, and resistance values obtained by measurements at the generator and receiver interconnect points. Loading caused by the circuits is defined as the number of or fractions of "unit loads" the circuit presents. Guidance is given with respect to limitations on data signaling rates imposed by the parameters of cable length, balance, and terminations for individual installations.

The parameter values specified in this standard are similar to those in RS-422-A. These values allow generators and receivers to be designed that can be used to meet the requirements of both standards. This standard does not specify other characteristics, such as signal quality, timing, protocol, pin assignments, etc., that are essential for proper operation of interconnected equipments. It is intended that this standard be referenced by other standards and specifications that specify the additional characteristics necessary to assure satisfactory interfacing of equipment.

Any devices which are represented as complying with limits established by this standard shall meet the applicable specifications within the ranges of those factors which are described as appropriate for the operation of the equipment; such as: power supply voltages, ambient temperature, and humidity.

2. APPLICABILITY

The provisions of this standard apply only to the generator and receiver components employed in communications between equipments where the information being conveyed is in the form of binary signals that may have a DC component. This standard is not, therefore, an interface standard, but shall be referenced by those interface standards or specifications employing generators and receivers having these electrical characteristics.

The circuits whose characteristics are specified herein will be utilized normally in data, timing, or control interconnections where the data signaling rate is up to 10 megabit/s. The parameters are so configured that any device in the system can operate with a common mode voltage as great as $\pm$ 7 volts. Operation with or without generator offset voltage is provided; however, the common mode voltage tolerance is lowered if the offset voltage is zero.

Requirements of the generators have been structured so that more than one generator on at once (contention) will not cause damage. Some methods of protecting the generators in contention situations may result in a delay before recovery. While generator contention may exist, there is no provision within this recommended standard for the detection of contention.

3. ELECTRICAL CHARACTERISTICS

Figure 3.1 shows an interconnection application of generators and receivers having the electrical parameters specified in this standard. The elements in the application are: generators, receivers, transmission cables, and termination resistances (R_t). The load on an active generator shall be defined in terms of unit loads (U.L.s). The load on the system caused by each receiver and passive generator shall be specified by the number of unit loads each presents. A unit load is defined by the current-voltage characteristics specified in paragraph 3.1. A load is defined as a passive generator (G), a receiver (R), or both. The electrical parameters specified in the following paragraphs are selected so that a generator can drive a total load having the value of 32 unit loads and an effective total termination resistance as low as 60 ohms.

The electrical characteristics that are specified are those that are measured at an interconnect point supplied by the device manufacturer. Tests made at the using equipment interface by referencing standards should recognize that the criterion at that point may be different due to internal wiring and other receivers and passive generators within the equipment. In subsequent figures, points A, B, and C represent generator interconnect points while A', B', and C' represent the interconnect points associated with receivers. The terminating resistors are considered part of the interconnecting means.

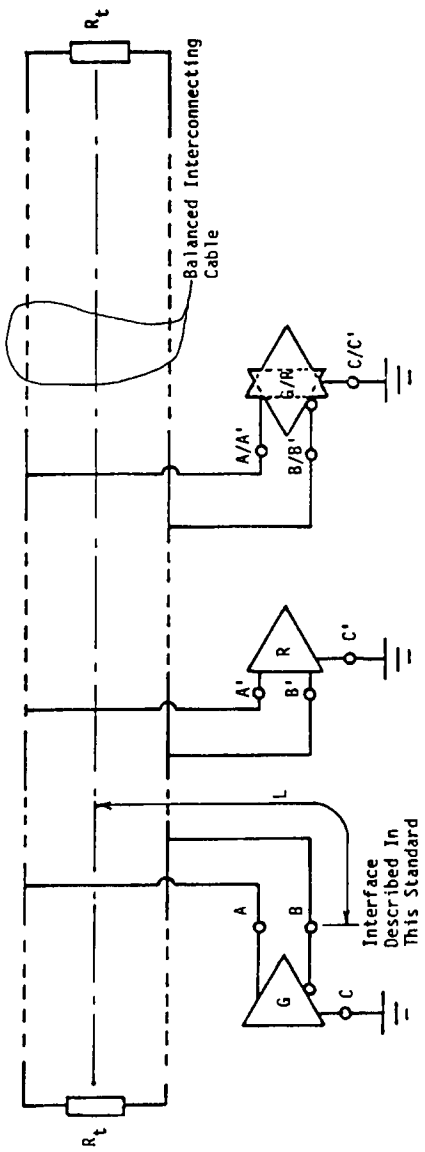
3.1 Load Characteristics

The loading caused by receivers and passive generators on the interconnect means must be considered in defining the device electrical characteristics. Two areas are of concern: the DC load and the AC load characteristics. The DC load each device places on the system is defined as a number or fractions of "unit loads" as defined below. The AC loading is not standardized but must be considered in the design of a system using the devices meeting this standard.

3.1.1 DC Unit Load Specification

In order that the current required from a generator in the active state be limited to a practical value, the loading effect of generators in the passive state and receivers is specified in terms of the loading produced by a hypothetical unit load (U.L.). With the voltage V_{ia} (or V_{ib}) ranging between -7 and +12 volts, while V_{ib} (or V_{ia}) is held at 0 volts (grounded), the resultant input current I_{ia} (or I_{ib}) shall remain within the shaded region shown in the graph in Figure 3.2 for one unit load. The actual curve of current versus voltage shall always have a positive slope to lower the possibility of oscillations caused by negative input resistance.

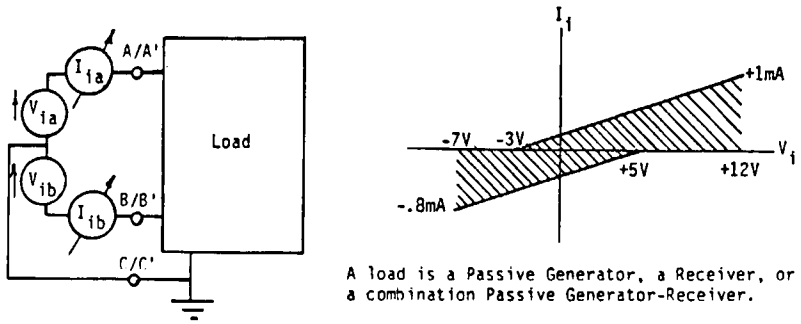
To determine the number of unit loads a passive generator or a receiver exhibits, the slope of the bounds in Figure 3.2 shall be modified to the minimum slope required to fully contain the current-voltage characteristics, while the -3V and +5V intercept points are maintained. The number of unit loads is then equal to the



G - Generator
R - Receiver
G/R- Combination Generator/Receiver
L - Length of stub; the guidelines assume length of stub to be effectively zero.
R_t- Termination resistance; location and value are not specified in this standard, but a generator can drive 32 unit loads plus two termination resistances of 120 ohms each.

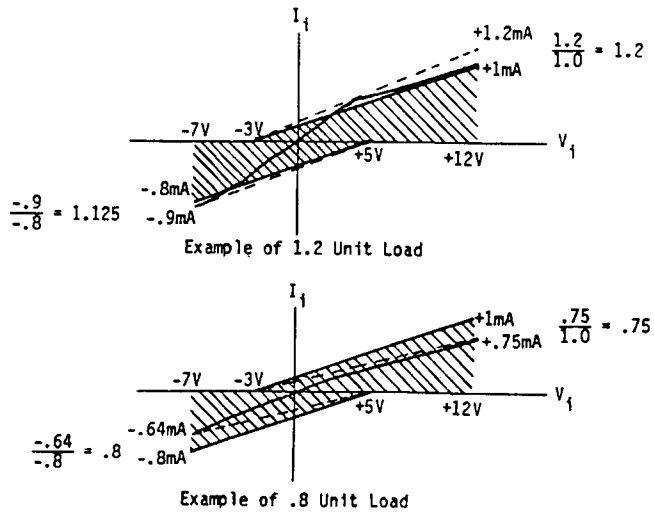
Guidance on the length of the interconnecting cable is given in the Appendix.

MULTIPOINT INTERCONNECT APPLICATION
FIGURE 3.1



A load is a Passive Generator, a Receiver, or a combination Passive Generator-Receiver.

UNIT LOAD INPUT CURRENT-VOLTAGE MEASUREMENT
FIGURE 3.2



Example of 1.2 Unit Load

Example of .8 Unit Load

EXAMPLES OF LOADS THAT ARE NOT ONE UNIT LOAD
FIGURE 3.3

larger of the two ratios of the new currents to the original currents at the -7V and +12V intercept points. Example determinations are shown in Figure 3.3. The number of equivalent U.L.s is represented by nU.L. in subsequent paragraphs.

The above described determinations should be performed in both power-on and power-off conditions. The system user should consider both of these conditions in determining the total number of loads presented to a generator. The current-voltage characteristics as a load's power supply is turned on or off is not standardized, but a user should recognize that such a transition may have some effect on system operation.

3.1.2 AC Load Characteristics

Devices connected to the interconnect means also display reactive characteristics which will have effects on the transmission characteristics of the interconnect means. These lumped effects will result in:

reflections
unbalance
distortion of the signal

Parameters causing these effects are such characteristics as the differential impedance, the impedance from A' to C' and from B' to C' on a receiver or the impedance from A to C and from B to C on a passive generator. Further discussion of these parameters is contained in the appendix. Standardization of these parameters has not been undertaken at this time but may be the subject of a future revision.

3.2 Generator Characteristics

A generator is in either the passive or the active state. While in the passive state, the generator load characteristics shall be specified in terms of the number of equivalent U.L.s, as defined in paragraph 3.1 and Figures 3.2 and 3.3. While in the active state, the generator electrical characteristics are specified in accordance with the measurements illustrated in Figures 3.4 through 3.6 and described in paragraphs 3.2.1 through 3.2.3. A generator circuit meeting these requirements results in a low impedance balanced voltage source that will produce a differential voltage applied to the interconnecting cable in the range of 1.5 volts to 5.0 volts. The signaling sense of the voltages appearing across the interconnection cable are defined as follows:

a. The A terminal of the generator shall be negative with respect to the B terminal for a binary 1 (MARK or OFF) state.

b. The A terminal of the generator shall be positive with respect to the B terminal for a binary 0 (SPACE or ON) state.

3.2.1 Open Circuit Measurement (Figure 3.4)

For either binary state, the magnitude of the differential voltage (V_O) measured between the two generator output terminals shall be not less than 1.5 volts and not more than 6.0 volts; and the magnitude of V_{oa} and V_{ob} measured independently between each generator output terminal and generator circuit ground shall be not more than 6.0 volts.

3.2.2 Test Termination Measurement (Figure 3.5)

Two load termination tests shall be performed. In one, a total load of 54 ohms is used and the output voltages and offset voltage are measured. In the other, a worst case situation, the common mode voltage is varied from -7 volts to +12 volts and the output voltage is measured.

Test Termination Measurement 1, with 54 ohm Load. With a test load of two resistors, 27 ohms each, connected in series between the generator output terminals, the magnitude of the differential voltage (V_t) measured between the two output terminals shall be not less than 1.5 volts nor greater than 5.0 volts for either binary state. For the opposite binary state, the polarity of V_t shall be reversed ($\overline{V_t}$). The magnitude of the difference in the magnitudes of V_t and $\overline{V_t}$ shall be less than 0.2 volts. The generator offset voltage V_{os} measured between the center point of the test load and generator circuit ground shall be between -1 volt and +3 volts for both binary states. While the common mode specifications are designed for deliberate offset voltage, the user is permitted to use zero offset. However, if zero offset is used, the tolerance to common mode is reduced. The magnitude of the difference between V_{os} for one binary state and $\overline{V_{os}}$ for the opposite binary state shall be less than 0.2 volts.

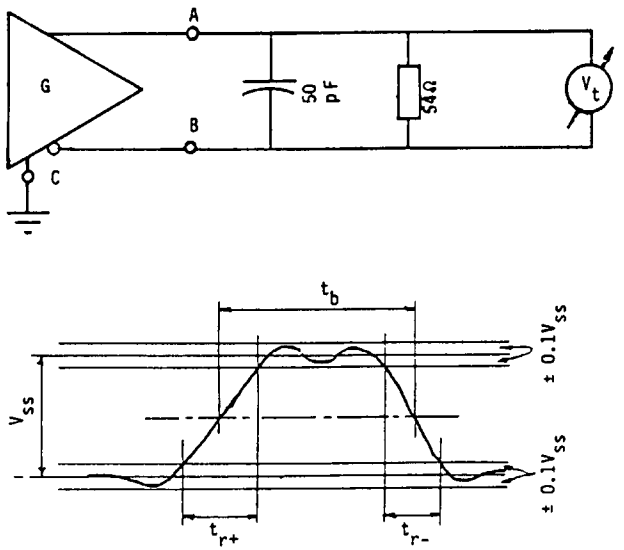
Test Termination Measurement 2, with Varying Common Mode Voltage. This test is performed with a test load of a 60 ohm resistor, paralleled with two 375 ohm resistors connected in series. The center connection of the two 375 ohm resistors is connected to a voltage source that is varied from -7 volts to +12 volts, to simulate the common mode voltages. The magnitude of the generator output voltage (V_t) shall not be less than 1.5 volts nor greater than 5.0 volts over the entire -7 volt to +12 volt voltage range. The test shall be performed for both binary states, V_t and $\overline{V_t}$.

3.2.3 Output Signal Wave Form (Figure 3.6)

During transitions of the generator output between alternating binary states (one-zero-one-zero, etc.), the differential voltage measured across a test load composed of a 54-ohm resistance paralleled with a 50 picofarad capacitance connected between the generator output terminals shall be such that the voltage monotonically changes between 0.1 and 0.9 of V_{ss} within less than 0.3 of the unit interval. Thereafter, the signal voltage shall not vary more than 10 percent of V_{ss} from the steady state value, until the next binary transition occurs, and at no time shall the instantaneous magnitude of V_t or $\overline{V_t}$ exceed 5.0 volts. V_{ss} is defined as the voltage difference between the two steady state values of the generator output.

3.3 Receiver Characteristics.

The receiver electrical characteristics are specified in accordance with the load specifications in paragraph 3.1 and with the sensitivity and balance measurements described in paragraphs 3.3.1 and 3.3.2 and illustrated in Figures 3.9 and 3.10. A circuit meeting these requirements results in a differential receiver having a high input impedance and a threshold that lies in the region between -0.2 and +0.2 volts.



t_b = Time duration of the unit interval at the applicable data signalling rate

$t_r < 0.3t_b$

V_{ss} = Difference in steady state voltages

$V_{ss} = |V_t - \overline{V_t}|$

GENERATOR OUTPUT SIGNAL WAVEFORM
FIGURE 3.6

3.3.1 Input Sensitivity Measurement (Figure 3.7)

The allowable range of input voltages ($V_{A'}$ and $V_{B'}$) appearing at the receiver input terminals (A' and B') measured with respect to receiver common (C') shall be between -7 volts and +12 volts. For any combination of receiver input voltages within this allowable range, the receiver shall assume the intended binary state with an applied differential input voltage (V_i) of ±0.2 volt or more. Additionally, the receiver shall not be damaged for any combination of receiver input voltages within this allowable range. Operation outside of this allowable range is not covered by this standard. The receiver sense shall be as follows:

a. the A' terminal of the receiver shall be negative with respect to the B' terminal for a binary 1 (MARK or OFF) state.

b. The A' terminal of the receiver shall be positive with respect to the B' terminal for a binary 0 (SPACE or ON) state.

(NOTE: Designers of terminating hardware should be aware that slow signal transitions with noise present may give rise to instability or oscillatory conditions in the receiving device, and therefore appropriate techniques should be implemented to prevent such behavior. For example, hysteresis may be incorporated into the receiver to help prevent such conditions.)

3.3.2 Receiver Unit Load Measurement

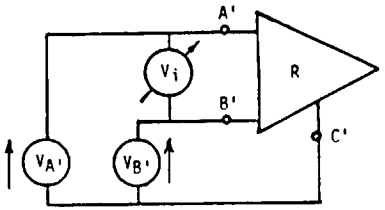
The receiver unit load (U.L.) value shall be determined according to the measurements specified in paragraph 3.1. (The number of equivalent U.L.s is represented by nU.L.)

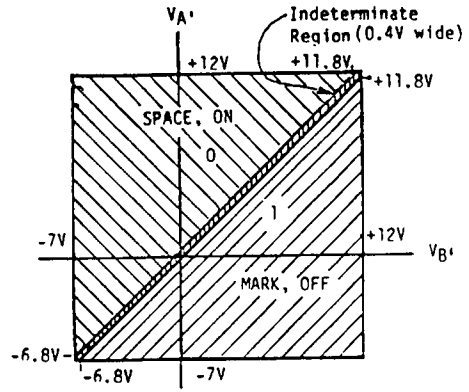
3.3.3 Input Balance Measurement (Figure 3.8)

The balance of the receiver input voltage-current characteristics and bias voltages shall be such that the receiver will remain in the intended binary state when a differential voltage (V_{R3}) of ±0.400 volts is applied through matched resistors equal to 1500/nU.L. ohms to each input terminal, as shown in Figure 3.8, with the input voltages (V_{R1} and V_{R2}) ranging between -7 volts and +12 volts. When the polarity of V_{R3} reverses, the opposite binary state shall be maintained under the same conditions. (The resistor values used will vary depending upon how many unit loads (U.L.) or fractions of a unit load the receiver loading represents.)

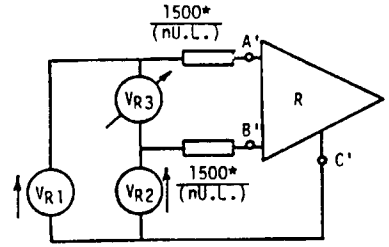
3.4 Fault Conditions Measurements

The following tests are intended to assure that devices will not be damaged due to single fault condition. While the major mechanisms of potential circuit failure are tested by the sections which follow, certain other dynamic conditions may stress the device. These conditions may require additional testing to assure the user that the device will perform properly under operating conditions.

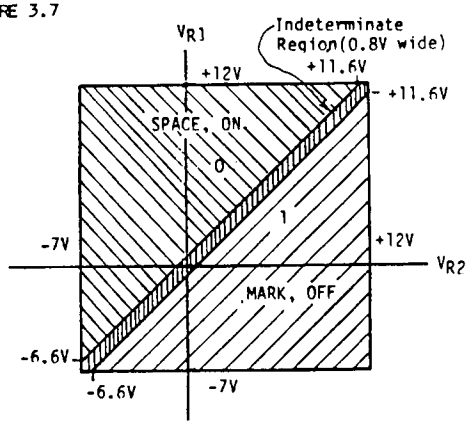




RECEIVER INPUT VOLTAGE RANGE
FIGURE 3.7



*Matched



RECEIVER INPUT BALANCE MEASUREMENT
FIGURE 3.8

3.4.1 Generator Short Circuit Measurement (Figure 3.9)

With the generator output terminals short-circuited to each other, the generator shall not be damaged.

3.4.2 Generator Contention Test (Figure 3.10)

A generator shall not sustain any damage as a result of connecting its output terminals to a voltage source, variable from -7 to +12 volts, under any output condition, binary 0, 1, or passive.

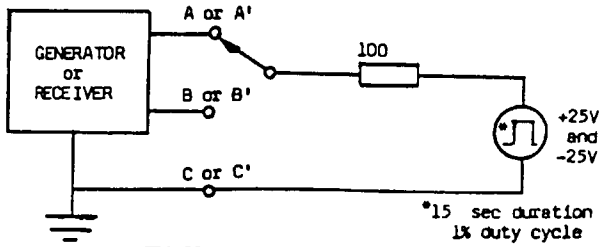
3.4.3 Generator Current Limit (Figure 3.10)

Under any conditions of the applied voltage test specified in 3.4.2, with the slew rate of the variable voltage source equal to or less than 1.2 volts per microsecond, the peak current in any lead to the generator terminals shall not exceed 250 mA. This criterion is not to be interpreted as a requirement that a generator be capable of sourcing 250 mA. Rather, the sinking generator shall not permit a composite current in excess of 250 mA, if multiple (sourcing) generators are providing that current. Appendix A.4 gives additional guidance on generator operation in a contention situation.

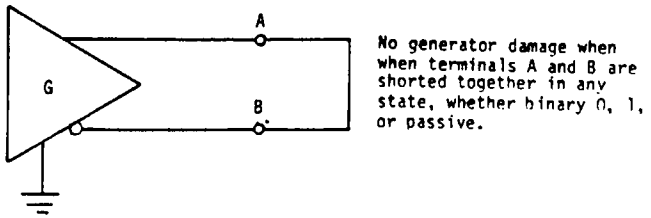
3.4.4 Transient Over Voltage Requirement (Figure 3.11)

This requirement applies to both generators and receivers and is necessary to provide for protection from transients that may occur on a line when the high current due to a single contending pair is interrupted. The test shall be performed in both a power on and power off condition. Further discussion of the need for this requirement is in the appendix (section A.4).

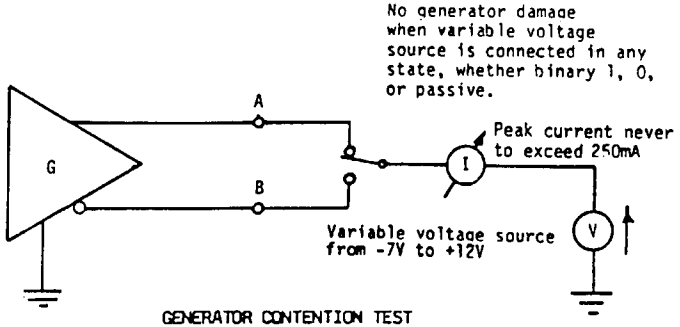
A passive generator or a receiver must be able to withstand without failure applied. pulses of 15 microseconds duration at 1 per cent duty cycle from a 25 volt source having 100 ohm source impedance. Both positive and negative pulses shall be applied between A and C and between B and C on passive generators and between A' and C' and between B' and C' on receivers. If the passive device should experience breakdown during the applied pulse, the device shall return to the passive state within 200 nanoseconds of the termination of the applied pulse.



TRANSIENT OVER VOLTAGE TEST
FIGURE 3.11



GENERATOR SHORT CIRCUIT MEASUREMENT
FIGURE 3.9



GENERATOR CONTENTION TEST
FIGURE 3.10

APPENDIX

GUIDELINES AND EXPLANATORY NOTES

This appendix is not a part of this recommended standard but is included for informational purposes only.

In applying the generators and receivers defined in this standard, several important considerations must be kept in mind. Among these, the first to be considered is the actual configuration of the system with regard to the number of generators and receivers, the operating speed of the system, the method of interconnecting the equipment, and the system margin. The performance capabilities of the equipments will necessarily have to be considered in establishing the margin allotments. The using standard is expected to specify the interconnecting means, and guidance for the specification of the interconnection is contained in this explanatory note.

A.1 General System Configuration.

A.1.1 The configuration of a system will normally consist of several generators, several receivers, and terminating resistors. In addition to the terminating resistors, each generator can drive up to 32 unit loads (U.L.s) consisting of both receivers and generators in the passive state. Special cases of one generator and multiple receivers or one receiver with multiple generators may also be configured. The terminating resistors should be placed at the extreme locations of the line. Figure 3.1 shows the general case.

A.1.2 The generators and receivers conforming to this standard can operate with a common mode voltage between -7 volts and +7 volts (instantaneous). The common mode voltage is defined to be any uncompensated combination of generator-receiver ground potential difference and longitudinally coupled peak noise voltage measured between the receiver circuit ground and cable with the generator ends of the cable short circuited to ground, plus the generator offset voltage (V_{OS}).

A.1.3 The developer of a system using these generators and receivers must consider the possible situation in which all generators may be in the passive state. Under this condition, no specific state can be assumed for any receiver. The user should provide for this condition with protocol or other fail safe considerations which are beyond the scope of this standard.

A.2 Interconnecting Means

The interconnecting means is that part of the system which connects the interface points and includes the cables, connectors, and terminating resistors. While the actual configuration of interconnect means is application dependent and beyond the scope of this standard, some guide lines which may be helpful in the choice of the interconnecting means are provided. The guidelines are for use where there are no stubs used to interconnect the interface points to the interconnecting cable.

A.2.1 The cable is not standarized; however, the following guidance for the selection of cable for some specific application may be useful. The important parameters that influence cable selection are:

a. Signalling rate, and hence the unit interval.
b. Minimum signal voltage to be presented at the receiver.
c. Maximum acceptable signal distortion.
d. Cable length required.

A.2.1.1 The unit interval (U.I.) of the signal determines the minimum time between transmitted signal transitions and thus the time available for the signal to achieve its final steady state. If the signal does not achieve its final steady state before the next transition occurs, the transition will appear to the receiver to be displaced in time and the signal will suffer from "intersymbol distortion." In chosing the cable, the relationship between unit interval and the rise time of the signal at the distant terminal must be considered. This ratio, t_r/U.I., is used in the guidance for selection of cable.

A.2.1.2 The minimum signal voltage presented to the receiver must be equal to or larger than the worst allowable receiver threshold. Any receiver input voltage in excess of this value is margin. The amount of margin needed in a system will depend upon noise consideration, allowable error rate, and amount of allowable signal distortion. To determine the cable characteristics, the user must first decide on the amount of receiver voltage desired to be presented to the worst case receiver. This information will be used to determine the minimum cable characteristics.

A.2.1.3 Signal distortion is a measurement of the displacement in time from the ideal instant that a significant event, such as a transition, occurs. Distortion is usually expressed as a percent of a unit interval. Some equipment is more tolerant of distortion than others. Knowing the maximum allowable distortion for a given application will provide an additional necessary input to determine the interconnecting cable characteristics.

A.2.1.4 The final ingredient in the cable characteristics determination is the length needed to interconnect all parties in the multipoint system. This is the maximum actual cable that is needed to interconnect without the use of stubs. System layout should be such as to keep this as short as possible consistent with the needs of the system. A determination made following the method below will establish some guideline to the expected performance of a cable over the length required.

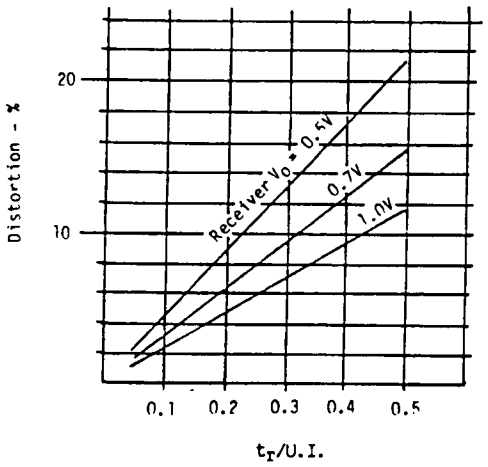
A.2.2 Choosing the Interconnecting Cable. Guidance for the choice of cable may be taken from the curve in Figure A.1. This curve is to be used as follows:

a. Determine the minimum unit interval for the signal to be exchanged over the interconnect means.

b. Choose the minimum output voltage (V_O) from the cable desirable in this system. This is the voltage that needs to be present at the receiver from the worst case driver located at the most distant point from a receiver.

c. Choose the maximum value of distortion allowable in the system.

d. Determine cable length needed for the system.



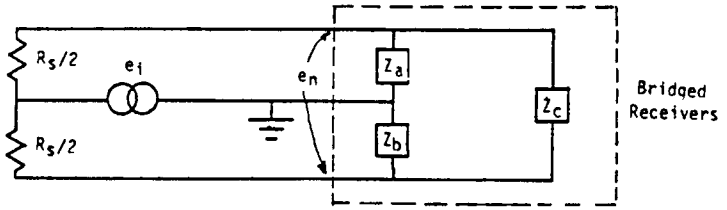
CURVE FOR DETERMINING DISTORTION FROM CABLE RISE TIME (t_r), UNIT INTERVAL (U.I.), and SIGNAL VOLTAGE TO RECEIVER
FIGURE A.1

Reflections are a result of the loading each node presents to the transmission medium. This load is an impedance which is predominantly capacitive. It consists of the connectors, wiring or printed circuit etch, and the receiving/driving element. This capacitive load causes a reflection which is a function of the signal rise time and the magnitude of the capacitive load.

This standard describes devices which may be used over a wide range of data signaling rates (up to 10M bit/s). The edge speeds (rise/fall time) of the driving elements must be fast enough to allow this high rate. The designer must be aware that even though a design may be required to operate at, for example, 19 kbit/s, the signal edge times may be less than 10 nanosecond and receivers may have correspondingly short response times. Disregarding the transmission line effect, even at low data signaling rates, can have a major impact on the data integrity of the design.

A.2.4 Interference and Balance

The susceptibility of a network to interference, whether the result of electromagnetic inductive or capacitive coupling to the media, is determined in part by the imbalance of its impedance to ground. Assuming the coupling of interference to each of the two conductors is equal, the magnitude of the component of the interference which appears between conductors will generally be determined by the imbalance of the impedance to ground. Consider an active generator at one end of a cable (pair) and several off generators and receivers bridged at the other end. Neglecting the generator output signal, the configuration can be approximated by:



Where: R_S is, at high frequencies, the characteristic impedance of cable, and at low frequencies, the loop resistances of the cable.

Z_a, Z_b, & Z_C are the corresponding impedances of the combination of the bridged receivers.

e_i is the magnitude of the interfering signal, as would appear to ground at one end of the cable with the other end shorted to ground.

e_n is the conductor-to-conductor component of the interference resulting from impedance imbalance.

Note that an active generator provides a low impedance to ground from both conductors of the cable, and therefore, at low frequencies, a common mode voltage will appear at the bridged receiver end of the cable as a voltage to ground with a source impedance of $R_S/4$ ($R_S/2$ for each conductor).

For the equivalent circuit shown, the balance of concern is the ratio of the voltage of the common mode interference to the resultant conductor-to-conductor noise voltage, e_n, or

$$\text{Bal} = 20 \log \left| \frac{e_i}{e_n} \right|$$

and, for $G_S = 1/R_S$ and $Y_X = 1/Z_X$,

$$\frac{e_i}{e_n} = \frac{(2G_S + Y_a)(2G_S + Y_b) + Y_C(4G_S + Y_a + Y_b)}{2G_S(Y_b - Y_a)}$$

Let: $Y_b - Y_a = Y_d$

and assuming : $Y_a \ll G_S$, $Y_b \ll G_S$, and $Y_C \ll G_S$,

as may be typical for the configuration, the following approximation is obtained:

$$\frac{e_i}{e_n} = \frac{2G_S}{Y_d}$$

This suggests that the imbalance of the configuration is inversely proportional to the sum of the differences in the admittances to ground for the two input terminals of the bridged receivers and that it is essentially independent of common mode admittance to ground ($Y_a + Y_b$) of the receivers.

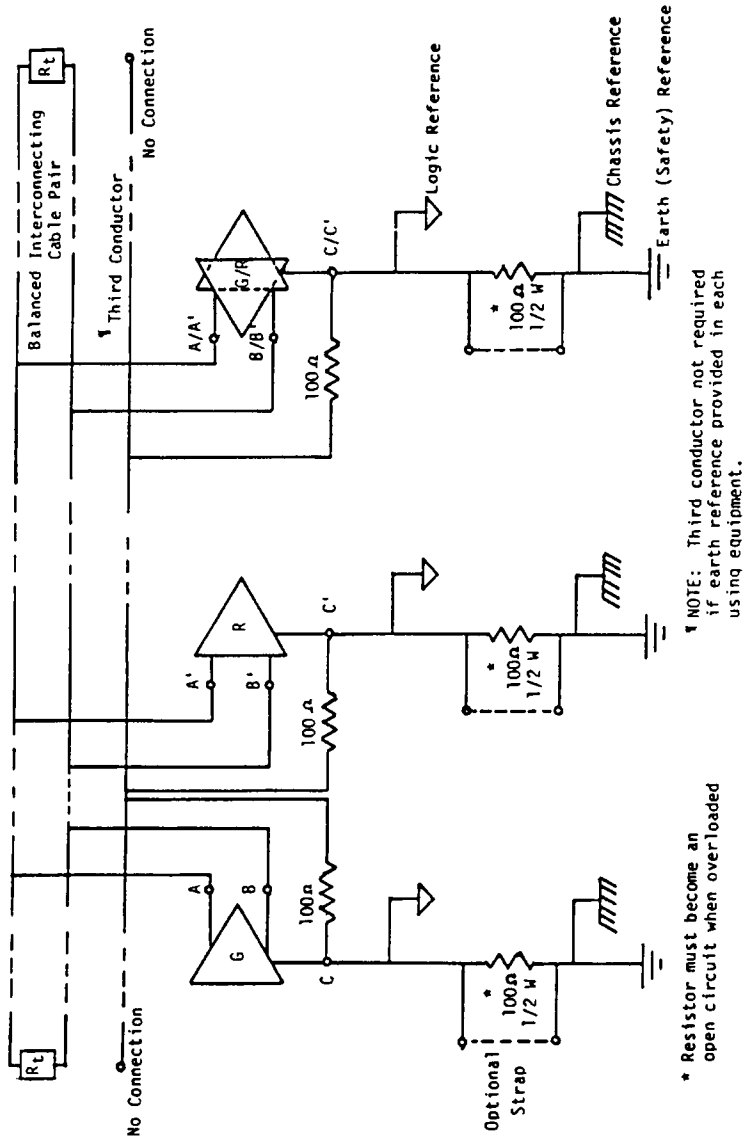
Balance is of concern up to at least the maximum frequency of a signal to which receivers will respond. Differences in the capacitance to ground, from the two receivers' input terminals, of only a few pico-farads can cause significant imbalance if the response of receivers extends to many magahertz. For example, 10 receivers, each having a capacitance difference (to ground) of 10 picofarads bridged on a 120-ohm cable, would result in a balance, at 10 MHz, of about 10dB. At higher frequencies (e.g., 50 MHz) the configuration would appear to have one conductor grounded.

The configuration and associated relationship discussed here are only typical. The relationships would not be valid for many other configurations. They are intended only to demonstrate that, in using devices which conform to the requirements of this standard, a designer of a network must consider the imbalance in the devices and associated equipment.

A.3 OPTIONAL GROUNDING ARRANGEMENTS

A.3.1 Signal Reference

Proper operation of the generator and receiver circuits requires the presence of a signal return path between the circuit grounds of the equipment at each end of the interconnection. The circuit reference may be established by a third conductor connecting the common leads of devices, or it may be provided by connections in each using equipment to an earth reference. The grounding arrangements are shown in Figure A.2. Where the circuit reference is provided by a third conductor, the connection between circuit common and the third conductor must contain some resistance (e.g., 100 ohms) to limit circulating currents when other ground connections are provided for safety.



GROUNDING ARRANGEMENTS
FIGURE A.2

A.3.2 Shield Ground

Some applications may require use of shielded interconnecting cable for EMI or other purposes. When employed, the shield shall be connected only to frame ground at either or both ends, depending on the specific application. The means of connection of the shield and any associated connector are beyond the scope of this standard.

A.4 Generator Contention

When two or more generators are connected to the same transmission line, a potential condition exists whereby both generators are simultaneously in the active state. If one generator (or more) is sourcing current while another generator is sinking current, excessive power dissipation may occur within either the sourcing or sinking element. This condition is defined as Generator Contention, since multiple generators are competing for one transmission line. Since system requirements may dictate that more than one generator is active simultaneously, the parameters in the Generator Contention Test (paragraph 3.4.2) of this standard were selected so that a practical limitation could be put on the dissipation within a generator.

Generator contention will occur under any or all of the following three conditions:

a. System Power-up
When a system is powered or regains power, multiple generators may be active simultaneously, during the interval of initialization.

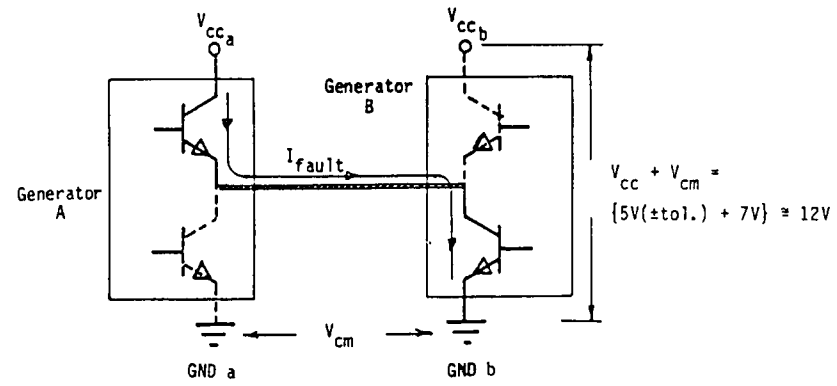
b. System Fault
A failure may occur in either hardware or software which may result in Generator Contention.

c. System Protocol
Certain system protocols may intentionally cause multiple generators to be active for brief periods while switching from one transmission to another. Also, other protocols may allow stations that are sharing one line to compete for a transmission, resulting in multiple generators being active simultaneously. However, one station will eventually succeed in acquiring the line, thereby ending the contention.

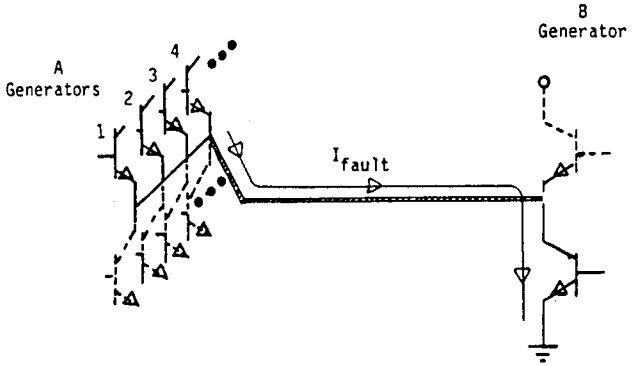
The generator failure mechanisms may be best described with the aid of Figures A.3 and A.4. These figures illustrate both the driving mechanisms.

Figure A.3 depicts two generators attached to a common transmission line. Generator A will be delivering its short circuit current to the sinking element of Generator B. This situation, worsened by the possible presence of a common mode voltage ($-7V<V_{CM}<+7V$) between two generators, can cause generator A to dissipate potentially excessive power for the generator design. As an example, if the generator's short circuit current is 250mA and the combination of supply voltage and common mode potential difference equals 12 volts, Generator A must dissipate approximately three watts.

Figure A.4 illustrates the condition where multiple generators are driving their short circuit current into one sinking element. As the sinking element comes out of saturation, the combination of collector to emitter voltage in conjunction with the large current that is being supplied may cause excessive power within Generator B. This may lead to the second type of failure, that of the sinking element.



CONTENTION WITH SINGLE SOURCING GENERATOR
FIGURE A.3



FAULT CONDITIONS WITH MULTIPLE SOURCING GENERATORS
FIGURE A.4

The examples in Figures A.3 and A.4 show that a means of protection must be designed within the generator to prevent either type of failure. The two most obvious solutions fall into the categories of current limiting and thermal shutdown. Although the solution to the generator contention problem may be either current limiting, thermal shutdown, or some combination of both, these means do not preclude some other means of protection as long as that protection is built into the generator.

Current limiting simply does not allow excessive dissipation within the generator, by restricting the amount of current under the contention condition. This means of protection has the advantage that it can recover quickly to deal with contention protocols. Thermal shutdown exhibits the opposite properties of current limiting in its slow recovery time from the contention situation and its inherent ability to sense power overload versus only high current levels.

When contention exists, the high currents cause energy to be stored in the line. When the current is abruptly interrupted, a voltage will be developed across the transmission line of

$$V = \frac{I_s Z_o}{2}$$

where V = developed voltage, in volts
I_s = short circuit current, in amperes
Z_o = line characteristic impedance, in ohms

With a 250 mA peak current limit, this voltage will equal about 15 volts. If four or more drivers are ON (line current up to 500 mA) and the unlikely event occurs where two drivers are turned off such that the current interruptions concur on the line, the differential voltage could rise to a larger value. The user is cautioned to consider this possibility in the system design. Paragraph 3.4.4 requirements are designed to protect when only one such event occurs.

A.5 Similarity With RS-422-A.

In certain instances, it may be possible to produce generators and receivers that meet the requirements of both RS-422-A and of RS-485. The following differences in parameter specifications exist between the two documents.

Characteristic	RS-422-A	RS-485
Generator		
Min. output voltage	2V into 100 ohms >1/2 open circuit V	1.5V into 54 ohms
$I_{short\ to\ ground}$	150 mA maximum	
$I_{short\ to\ -7,+12\ volts}$		250 mA peak
$t_{rise\ time}$	$<0.1t_b$ 100 ohm load	$<0.3t_b$ 54 ohm, 50 pF load