SYSTEM 80/10
MICROCOMPUTER
HARDWARE REFERENCE MANUAL

Manual Order Number: 9800316B

Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051

PREFACE

This manual provides general information, installation, programming information, principles of operation, and service information for the Intel System 80/10 Microcomputer using either the SBC 80/10 or the SBC 80/10A. Unless specified otherwise references to the System 80/10 are valid for both Single Board Computers. The areas where differences occur are identified individually. Additional systems information and component part details are available in the following documents:

- Intel Microcomputer Systems Data Book,
  Part No. 98-414

- Intel 8080 Microcomputer Systems User's Manual,
  Part No. 98-153

- Intel Multibus Interfacing Application Note,
  AP-28

- Intel 8255 Programmable Peripheral Interface Application Note,
  AP-15

- Intel 8251 Universal Synchronous/Asynchronous Receiver/Transmitter Application Note,
  AP-16

TABLE OF CONTENTS

APPENDICES

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1
INTRODUCTION

The System 80/10 is a member of Intel's complete line of OEM Computer systems that take full advantage of Intel's LSI technology to provide economical computer solutions for OEM applications. The System 80/10 is a completely packaged, self-contained computer system in a compact 3.5 inch high, 19 inch wide RETMA compatible chassis. The CPU, system clock, read/write memory, non-volatile read-only-memory, parallel I/O ports and drivers, serial communications interface, system backplane, power supply, fans, and OEM front panel are all included in one slim-line chassis.

Throughout this manual references to the system 80/10 are valid for systems using either the SBC 80/10 or the SBC 80/10A. Areas where differences occur are identified as SBC 80/10 only and SBC 80/10A only.

A Single Board Computer, the SBC 80/10 or the SBC 80/10A, provides the processing power for the System 80/10. The System has expansion capacity for an additional three expansion boards inside the System chassis. Intel's powerful 8-bit n-channel MOS 8080A CPU, fabricated on a single LST chip, is the central processor for the System 80/10. The 8080A contains six general purpose registers and an accumulator. The six general purpose registers may be addressed individually or in pairs providing both single and double precision operators. The 8080A has a 16-bit program counter which allows direct addressing of up to 64K bytes of memory, and the System 80/10 can be expanded with standard expansion boards to utilize up to 53K words of the addressing space. An external stack, located within any portion of memory, may be used as a last-in first-out stack to store and retrieve the contents of the program counter, flags, accumulator and all of the six general purpose registers. A sixteen bit stack pointer controls the addressing of this external stack. This stack provides almost unlimited subroutine nesting. Sixteen line address and eight line bidirectional data busses are used to facilitate easy interface to memory and I/O.

1.1 8080A ARCHITECTURE

The powerful 8080A instruction set allows the user to write efficient programs in a minimum amount of time. The accumulator group instructions include arithmetic and logical operators with direct, register indirect,

and immediate addressing modes. Move, load, and store instruction groups provide the ability to move either 8 or 16 bits of data between memory, the six working registers and the accumulator using all addressing modes. The ability to branch to different portions of a program is provided with jump, jump conditional, and computer jumps. The ability to conditionally and unconditionally call to and return from subroutines is provided. The RESTART (or single byte call instruction) is used for interrupt operation. Double precision operators such as stack manipulation and double add instructions extend both the arithmetic and interrupt handling capability of the 8080A. The ability to increment and decrement memory, the six general registers and the accumulator is provided as well as extended increment and decrement instructions to operate on the register pairs and stack pointer.

1.2 SYSTEM ORGANIZATION

The System 80/10 may contain either an SBC 80/10 or an SBC 80/10A, depending on date of manufacture. The only difference is in the type and quantity of memory available on each board.

The System 80/10 with the SBC 80/10 contains 1K 8-bits words of read/write memory using Intel's 8111 Low Power Static RAMs. Sockets for up to 4K 8-bit words of non-volatile read-only memory may be added in 1K byte increments using Intel's 8708 Erasable and Electrically Reprogrammable ROM's (EPROMs) or Intel's 8308 Metal Masked ROMs.

The System 80/10 with the SBC 80/10A contains 1K 8-bit words of read/write memory using Intel's 8102 Low Power Static RAMs. Sockets for up to 4K or 8K words of non-volatile read-only memory are provided on the SBC 80/10A. Up to 4K words of read-only memory may be added in 1K byte increments using Intel's 8708 erasable and electrically reprogrammable ROMs (EPROMs), Intel's 2758 Erasable and Electrically Reprogrammable ROMs (EPROMs), or Intel's 8308 Metal Mask ROMs. Optionally up to 8K words of read-only memory may be added in 2K byte increments using Intel's 2716 Erasable and Electrically Reprogrammable ROMs (EPROMs) or Intel's 8316E Metal Masked ROMs.

The System 80/10 contains 48 programmable parallel Input/Output (I/O) lines implemented using two Intel 8255 Programmable Peripheral Interface devices. The software is used to configure the I/O lines in combinations of unidirectional input/output, and bidirectional ports. Therefore, the I/O interface may be customized to meet specified peripheral requirements in order to take full advantage of the large number of possible I/O line drivers and terminators. Hence, the flexibility of the I/O interface is further enhanced by the capability of selecting the appropriate optional line drivers and terminators for each application.

A programmable serial communications interface using Intel's 8251 Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the board. The USART can be programmed by the systems software to provide virtually any serial data transmission technique presently in use (including IBM Bi-Sync). The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity and asynchronous serial transmission rate (within limitations given later) are all under program control. The 8251 provides full duplex, double buffered transmission and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The inclusion of jumper selectable TTY, or RS232C compatible interfaces on the board in conjunction with the USART provide a direct interface to a TTY, CRT, RS232C compatible devices, and asynchronous and synchronous modems.

A single-level interrupt may originate from any one of six sources including the USART, programmable I/O interface, and two user designated interrupt request lines. When an interrupt request is recognized, a RESTART 7 instruction is generated. The processor responds by suspending program execution and executing a user defined interrupt service routine.

Inside the compact chassis you will find a complete, quad output power supply with ample capability to support most combinations of computing power, I/O and memory expansion boards. Dual cooling fans are provided to circulate the air across the computer boards and power supply. The I/O section of the System 80/10 is easily accessible from the rear of the System chassis and the power supply as well as the system bus is accessible from behind the OEM front panel.

1.3 SYSTEM MONITOR

A standard feature of the System 80/10 is a software monitor which is programmed in two Intel ROMs. The monitor provides the user with two basic capabilities: 1) it gives the user access to console input and output routines as well as paper tape input and output control software and 2) the monitor along with a TTY or CRT provides the user with a "virtual console" that has immediate access to memory and registers and has control commands to begin execution and to display or alter the contents of the memory or registers. The system monitor ROMs are installed in the first two ROM sockets of the SBC 80/10 computer, occupying locations 0 to $2{,}048_{10}$.

The development cycle of System 80/10 based OEM products may be significantly reduced using the Intellec MDS. The resident assembler, text editor, and system monitor greatly simplify the design, development, and debug of System 80/10 based system software. A unique In-Circuit Emulator (ICE-80) Intellec MDS option provides the capability of executing and debugging OEM system software directly on the System 80/10.

Intel's high level language, PL/M, can be used to significantly decrease the time required to develop OEM system software.

CHAPTER 2
SYSTEM OVERVIEW

The System 80/10 is a fully packaged microcomputer utilizing either the SBC 80/10 or SBC 80/10A single board computers. The System can be divided into five functional blocks (see Figure 2-1). They are as follows:

1) SBC 80/10 or SBC 80/10A
2) Front Panel
3) Modular cardcage and backplane assembly
4) Power Supply
5) System Monitor

SBC 80/10

The SBC 80/10 is a complete computer system on a single 6.75 x 12 inch printed circuit card. The CPU, system clock, RAM, non-volatile ROM, I/O ports and drivers, serial communication interface, bus control logic and drivers all reside on the board.

Front Panel

The System 80/10 Front Panel consists of the AC power ON/OFF switch and the system reset circuitry. The simplicity of the System 80/10 Front Panel allows the OEM user to add his own mylar overlay or structural foam cover to meet his own design needs.

Modular Cardcage and Backplane Assembly

The Modular Cardcage and Backplane Assembly is installed in the chassis to house the SBC 80/10 and provide an easily accessible bus interface. The cardcage houses the SBC 80/10 and up to three expansion boards. All SBC 80 bus signals are present on each mating connector.

Power Supply

The System 80/10 power supply provides regulated DC output power at +12, +5, -5 and -12 volt levels. The power supply is chosen to provide

power for a fully loaded SBC 80/10 and most combinations of up to three SBC memory, I/O or combination expansion boards. The power supply also provides an "AC low" power failure output, TTL logic level, for system power-down control.

System Monitor

The System 80/10 Monitor is an Intel® 8080 program provided in two pre-programmed 1K ROMs. The monitor accepts and acts upon user commands to operate on the System 80/10 memory and I/O. It also provides input and output facilities in the form of I/O drivers for user console devices. The monitor provides the following facilities:

- Display selected areas of memory and processor registers.
- Initiate execution of user programs.
- Modify contents of memory and processor registers.
- Insert instruction(s) into memory.
- Reposition blocks of data in memory.
- Input hexadecimal file from TTY reader to memory.
- Output hexadecimal file to TTY punch from memory.

The monitor communicates with the user through an interactive console device, normally a TTY or CRT terminal. The dialogue between the operator and monitor consists of user originated commands in the monitor's command language, and monitor responses, either in the form of a printed message or an action being performed. The monitor begins the dialogue by printing the sign-on message "SBC 80P Monitor" on the console and requesting a command by presenting a prompt character, "." (period).

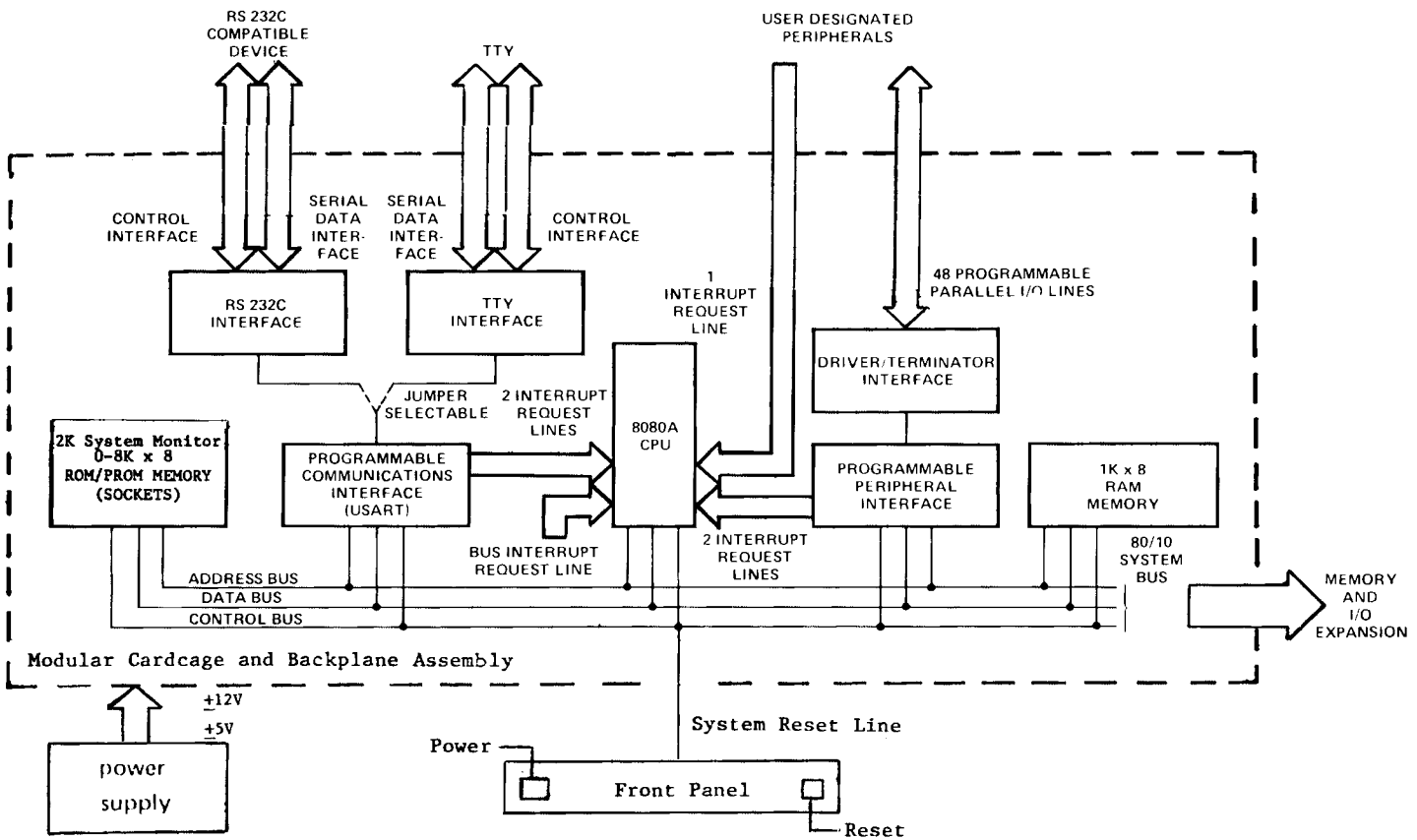


FIGURE 2-1 SYSTEM 80/10 BLOCK DIAGRAM

CHAPTER 3

SBC 80/10 AND 80/10A MODULE

3.1 FUNCTIONAL ORGANIZATION OF THE SBC 80/10 AND SBC 80/10A MODULE

For descriptive purposes, the circuitry on the SBC 80/10 and 80/10A can be divided into six functional blocks:

1) CPU Set
2) System Bus Interface
3) Random Access Memory (RAM)
4) Read Only Memory (ROM/PROM) Logic
5) Serial I/O Interface
6) Parallel I/O Interface

as shown in Figure 3-1.

The CPU Set consists of the 8080A Control Processor, the 8224 Clock Generator and the 8238 System Controller. The CPU Set is the heart of the SBC 80/10. It performs all system processing functions and provides a stable timing reference for all other circuitry in the system. The CPU Set generates all of the address and control signals necessary to access memory and I/O ports both on the SBC 80/10 and external to the SBC 80/10. The CPU Set is capable of fetching and executing any of the 8080's seventy-eight instructions. The CPU Set responds to interrupt requests originating both on and off the SBC 80/10, to HOLD requests from modules wishing to acquire control of the system bus, and to WAIT requests from memory or I/O devices having an access time which is slower than the 8080's cycle time.

The System Bus Interface includes an assortment of circuitry which gates interrupt requests, HOLD requests, READY (no wait inputs and the system reset input to the appropriate pins of the CPU Set. Other circuits drive the various external system control signals. The System Bus Interface also includes two 8216 bidirectional bus drivers which drive the memory data bus on the SBC 80/10. Six 8226 devices drive the external system data and address busses.

The Random Access Memory (RAM) provides the System 80/10 user with 1024 x 8-bits of on board read/write storage. Eight Intel 8111 Low Power Static RAM chips (256 x 4-bits each) are mounted on the SBC 80/10. The

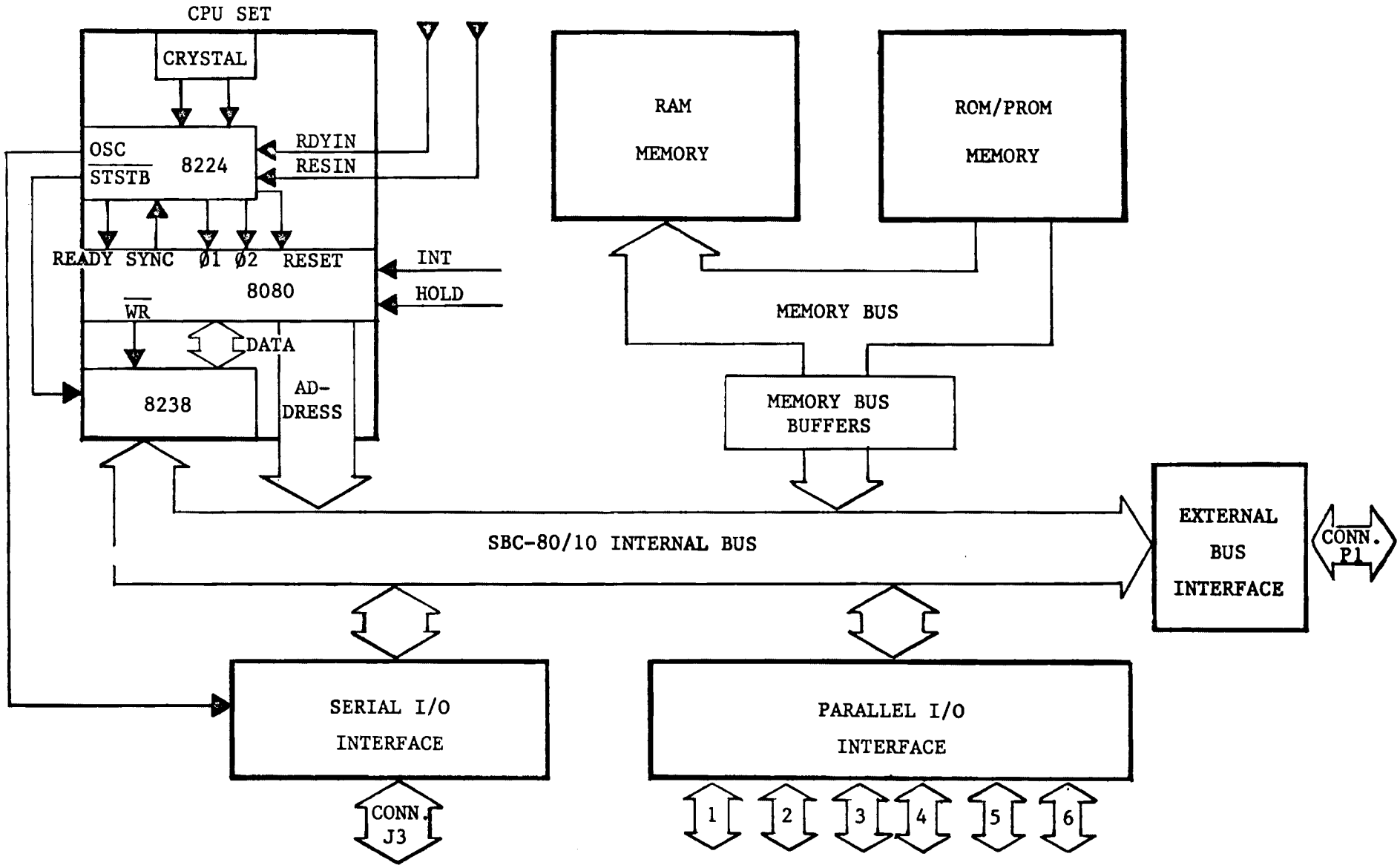


FIGURE 3-1 SBC 80/10 FUNCTIONAL BLOCK DIAGRAM

SBC 80/10A has eight Intel 8102 Low Power Static RAM chips (1024 x 1-bit each).

The Read Only Memory (ROM/PROM) section provides the user with the necessary provisions for installing up to 4096 x 8-bit of ROM or PROM on the SBC 80/10 and up to 8192 x 8-bits of ROM or PROM on the SBC 80/10A. The 80/10 and 80/10A have four 24-pin sockets that can accept either Intel 8708 Erasable and Electrically Reprogrammable Read Only Memory chips or Intel 8308 Metal Masked Read Only Memory Chips. Optionally the SBC 80/10A accepts Intel 2716 Erasable and Electrically Reprogrammable ROM (EPROM) chips, Intel 2758 Erasable and Electrically Reprogrammable ROM (EPROM) chips, or Intel 2316E Metal Masked ROM chips. The total ROM/PROM memory capacity using 8708, 8308 or 2758 chips is 4K x 8-bits and 8K x 8-bits using 2716 or 2316E chips.

The Serial I/O Interface, using Intel's 8251 USART device, provides a bidirectional serial data communications channel that can be programmed to operate with most of the current serial data transmission protocols. Synchronous or asynchronous mode, baud rate, character length, number of stop bits and the choice of even, odd or no parity are all program selectable. The user also has the option of configuring the Serial I/O Interface as an EIA RS232 interface or as a Teletype-compatible current loop interface.

The Parallel I/O Interface, using two Intel®8255 Programmable Peripheral interface devices, provides 48 signal lines for the transfer and control of data to or from peripheral devices. Eight lines already have a bidirectional driver and termination network permanently installed. This bidirectional network allows these eight lines to be inputs, outputs, or bidirectional (selected via jumpers). The remaining 40 lines, however, are uncommitted. Sockets are provided for the installation of drivers or termination networks as required to meet the specific needs of the user system.

3.2 THEORY OF OPERATION

In the preceding chapter we introduced each of the SBC 80/10 functional blocks and defined what each block was capable of doing. In this chapter we shall go one step further and describe how each block performs its particular function(s). The text will constantly refer to the SBC 80/10 schematics, provided in Appendix B.

Note: Both active-high (positive true) and active-low (negative true) signals appear on the SBC 80/10 schematics. To eliminate any confusion when reading this chapter, the following convention will be adhered to: whenever a signal is active-low, its mnemonic is followed by a slash; for example, MRDC/ means that the level on that line will be low when the memory read command is true (active). If the signal is subsequently inverted, thus making it active-high, the slash is omitted; for example, MRDC means that the level on that line will be high when the memory read command is true.

3.2.1 THE CPU SET

The CPU Set consists of three Intel® integrated circuit devices:

* 8080A Central Processor Unit
* 8224 Clock Generator
* 8238 System Controller

and an 18.432 MHz crystal that establishes the frequency of oscillation for the 8224 device via a 10pF capacitor, as shown in Figure 3-2. Together the elements in the CPU Set perform all central processing functions. The following paragraphs describe how the elements within the CPU Set interact with all other logic on the SBC 80/10. The interaction between the ICs within the CPU Set, however, is not described. Instead, the reader is referred to the Intel "8080 Microcomputer Systems User's Manual" for a detailed description of the 8080, 8224 and 8238 devices.

The CPU Set is shown on sheet 1 of the SBC 80/10 schematic (Appendix B).

3.2.1.1 INSTRUCTION TIMING

The activities of the CPU Set are cyclical. The CPU fetches an instruction, performs the operations required, fetches the next instruction, and so on. This orderly sequence of events requires precise timing. The 8224 Clock Generator, provides the primary timing reference for the CPU Set. The crystal in conjunction with a 10pF capacitor tunes an oscillator within the 8224 to precisely 18.432 MHz. The 8224 "divides" the oscillations by nine to produce two-phase timing inputs (Ø1 and Ø2) for the 8080. The Ø1 and Ø2 signals define a cycle of approximately 488 ns. duration. A TTL level phase 2 (Ø2TTL) signal is also derived and made

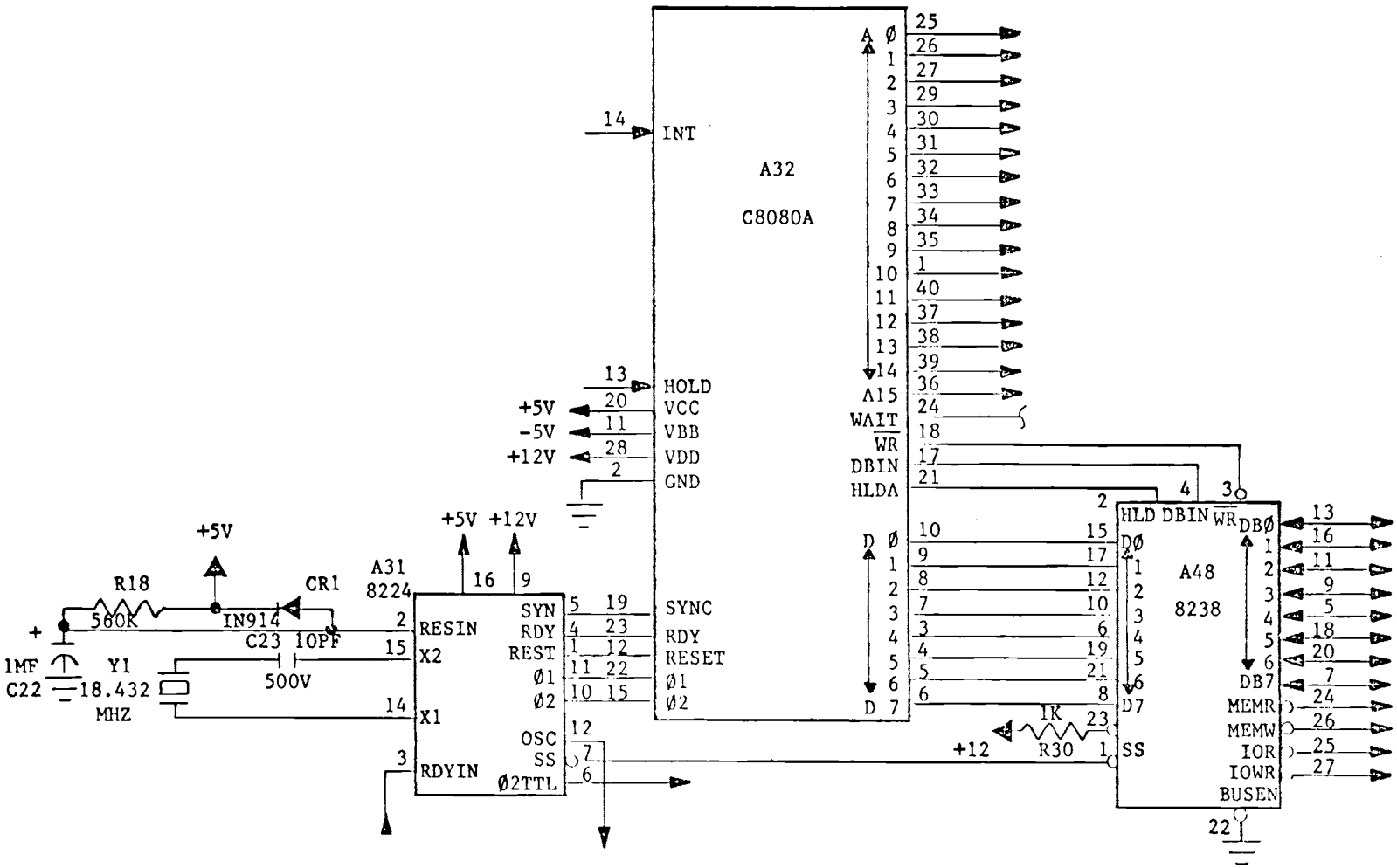


Figure 3-2 THE CPU SET

available to external logic. In addition, the output of the oscillator is buffered and brought out on OSC so that other system timing signals can be derived from this stable, crystal controlled source (e.g., the serial I/O baud rate is derived from OSC). All processing activities of the CPU Set are referred to the period of the Ø1 and Ø2 clock signals.

Within the 8080 CPU Set, an instruction cycle is defined as the time required to fetch and execute an instruction. During the fetch, a selected instruction (one, two or three bytes) is extracted from memory and deposited in the CPU's operating registers. During the execution part, the instruction is decoded and translated into specific processing activities.

Every instruction cycle consists of one, two, three, four or five machine cycles. A machine cycle is required each time the CPU accesses memory or an I/O port. The fetch portion of an instruction cycle requires one machine cycle for each byte to be fetched. The duration of the execution portion of the instruction cycle depends on the kind of instruction that has been fetched. Some instructions do not require any machine cycles other than those necessary to fetch the instruction; other instructions, however, require additional machine cycles to write or read data to/from memory or I/O devices.

Each machine cycle consists of three, four or five states. A state is the smallest unit or processing activity and is defined as the interval between two successive positive-going transitions of the Ø1 clock pulse.

There are three exceptions to the defined duration of a state. They are the WAIT state, the hold (HLDA) state and the halt (HLTA) state. Because the WAIT, the HLDA, and the HLTA states depend upon external events, they are by their nature of indeterminate length. Even these exceptional states, however, must be synchronized with the pulses of the driving clock. Thus the duration of all states, including these, are integral multiples of the clock pulse.

To summarize, then, each clock period marks a state; three to five states summarize a machine cycle; and one to five machine cycles comprise an instruction cycle. A full instruction cycle requires anywhere from

four to seventeen states for its completion, depending on the kind of instruction involved.

There is just one consideration that determines how many machine cycles are required in any given instruction cycle: the number of times that the processor must reference a memory address or an I/O address, in order to fetch and execute the instruction. Like many processors, the 8080 is so constructed that it transmits one address per machine cycle. Thus, if the fetching and execution of an instruction requires two memory references, then the instruction cycle associated with that instruction consists of two machine cycles. If five such references are called for, then the instruction cycle contains five machine cycles.

Every instruction cycle has at least one reference to memory, during which the instruction is fetched. An instruction cycle must always have a fetch, even if the execution of instruction requires no further references to memory. The first machine cycle in every instruction cycle is therefore a FETCH. Beyond that, there are no fast rules. It depends on the kind of instruction. The input (INP) and the output (OUT), instructions each require three machine cycles: a FETCH, to obtain the instruction; a MEMORY READ, to obtain the address of the object peripheral; and an INPUT or an OUTPUT machine cycle, to complete the transfer.

Every machine cycle within an instruction cycle consists of three to five active states (referred to as T1, T2, T3, T4, and T5). The actual number of states depends upon the instruction being executed, and on the particular machine cycle within the greater instruction cycle. Figure 3-3 shows the timing relationships in a typical FETCH machine cycle. Events that occur in each state are referred to transitions of the Ø1 and Ø2 clock pulses.

At the beginning of each machine cycle (in state T1), the 8080 activates its SYNC output and issues status information on its data bus. The 8224 accepts SYNC and generates an active-low status strobe (STSTB/) as soon as the status data is stable on the data bus. The status information indicates the type of machine cycle in progress. The 8238 System Controller accepts the status bits from the 8080 and STSTB/ from the 8224, and uses them to generate the appropriate control signals (MEMR/, MEMW/, IOR/

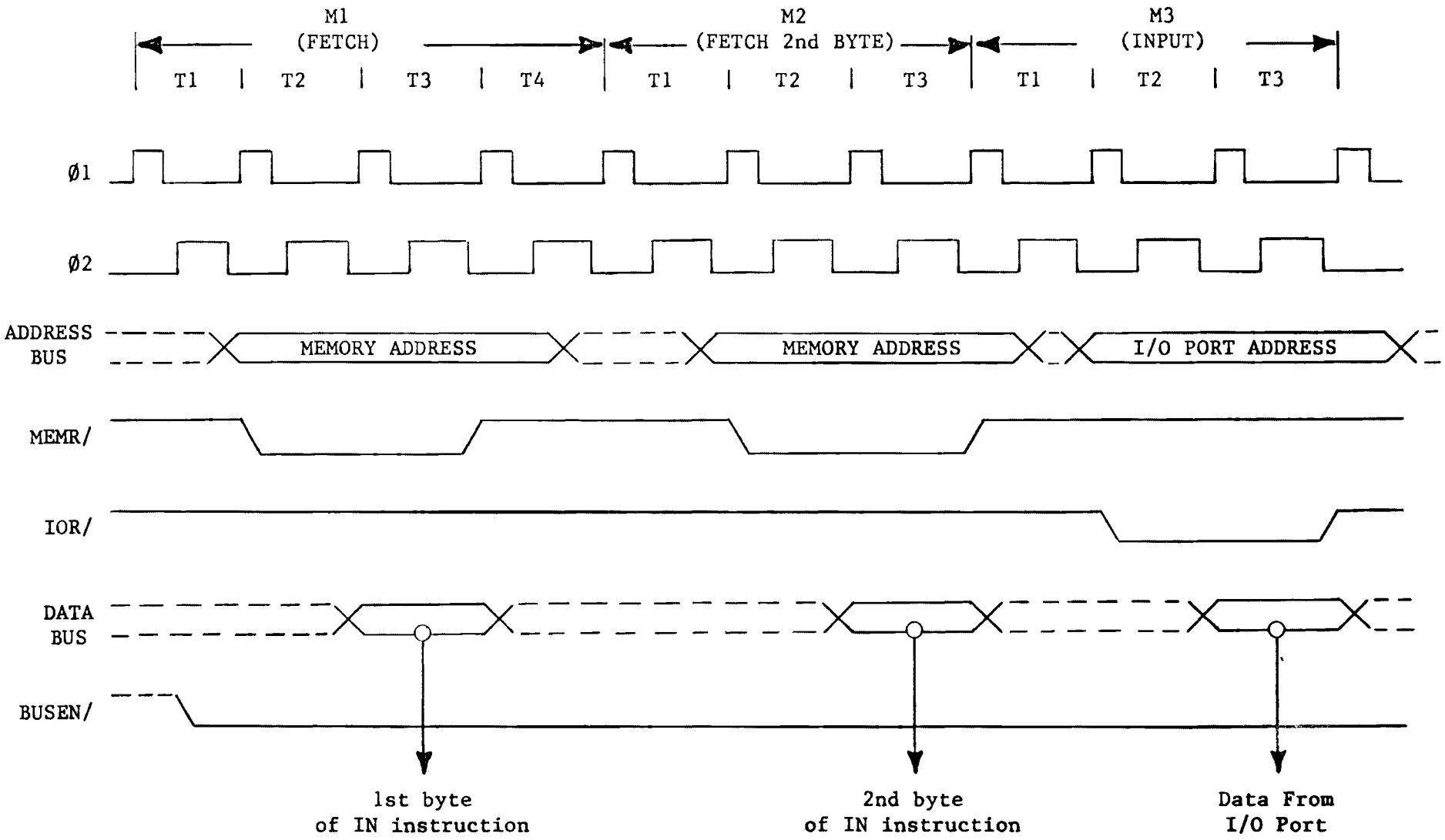


FIGURE 3-3 TYPICAL FETCH MACHINE CYCLE

and IOWR/) for the current machine cycle.

The rising edge of Ø2 during T1 loads the processor's address lines (A0 - A15). These lines become stable within a brief delay of the Ø2 clocking pulse, and they remain stable until the first Ø2 pulse after state T3. This gives the processor ample time to read the data returned from memory.

Once the processor has sent an address to memory, there is an opportunity for the memory to request a WAIT. This it does by pulling the 8224's RDYIN line low. As long as the RDYIN line remains low, the CPU Set will idle, giving the memory time to respond to the addressed data request. The 8224 synchronizes RDYIN with internal processor timing and applies the result to the 8080's READY input. The processor responds to a wait request by entering an alternative state (TW) at the end of T2, rather than proceeding directly to the T3 state. A wait period may be of indefinite duration. The 8080 remains in the waiting condition until its READY line again goes high. The cycle may then proceed, beginning with the rising edge of the next Ø1 clock. A WAIT interval will therefore consist of an integral number of TW states and will always be a multiple of the clock period.

The events that take place during the T3 state are determined by the kind of machine cycle in progress. In a FETCH machine cycle, the CPU Set interprets the data on its data bus as an instruction. During a MEMORY READ, signals on the same bus are interpreted as a data word. The CPU Set itself outputs data on this bus during a MEMORY WRITE machine cycle. And during I/O operations, the CPU Set may either transmit or receive data, depending on whether an INPUT or an OUTPUT operation is involved. Consider the following two examples.

Figure 3-4 illustrates the timing that is characteristic of an input instruction cycle. During the first machine cycle (M1), the first byte of the two-byte IN instruction is fetched from memory. The 8080 places the 16-bit memory address on the system bus near the end of state T1. The 8238 activates the memory read control signal (MEMR/) during states T2 and T3 (and any intervening wait states, if required). During the next machine cycle (M2), the second byte of the instruction is fetched. During

the third machine cycle (M3), the IN instruction is executed. The 8080 duplicates the 8-bit I/O address on address lines ADR0-7 and ADR8-F. The 8238 activates the I/O read control signal (IOR/) during states T2 and T3 of this cycle. In all cases the system bus enable input (BUSEN/) to the 8238 allows for normal operation of the data bus buffers and the read/write control signals. If BUSEN/ goes high the data bus output buffers and control signal buffers are forced into a high-impedance state.

Figure 3-5 illustrates an instruction cycle during which the CPU Set outputs data. During the first two machine cycles (M1 and M2), the CPU Set fetches the two-byte OUT instruction. During the third machine cycle (M3), the OUT instruction is executed. The 8080 duplicates the 8-bit I/O address on lines ADR0-7 and ADR8-F. The 8238 activates an advanced I/O write control signal (IOWR/) at the beginning of state T2 of this cycle. The nature and implications of the 8238 timing will be explained later (page 3-17). The 8238 outputs the data onto the system bus at the end of state T2. Data on the bus remains stable throughout the remainder of the machine cycle. BUSEN/ must be low to prevent the output and control buffers from being forced into the high impedance state.

Observe that a RDYIN signal is necessary for completion of an output machine cycle. Unless such an indication is present, the processor enters the TW state, following the T2 state. Data on the output lines remains stable in the interim, and the processing cycle will not proceed until the RDYIN line again goes high.

The 8080 generates a WR/ output for qualification of the advanced I/O write (IOWR/) and memory write (MEMW/) control signals from the 8238, during those machine cycles in which the CPU Set outputs data. The negative going leading edge of WR/ is referred to the rising edge of the first Ø1 clock pulse following T2. WR/ remains low until re-triggered by the leading edge of Ø2, during the state following T3. Note that any TW states intervening between T2 and T3 of the output machine cycle will necessarily extend WR/.

All processor machine cycles consist of at least three states: T1, T2, and T3 as just described. If the CPU Set has to wait for a RDYIN response, then the machine cycle may also contain one or more TW states.

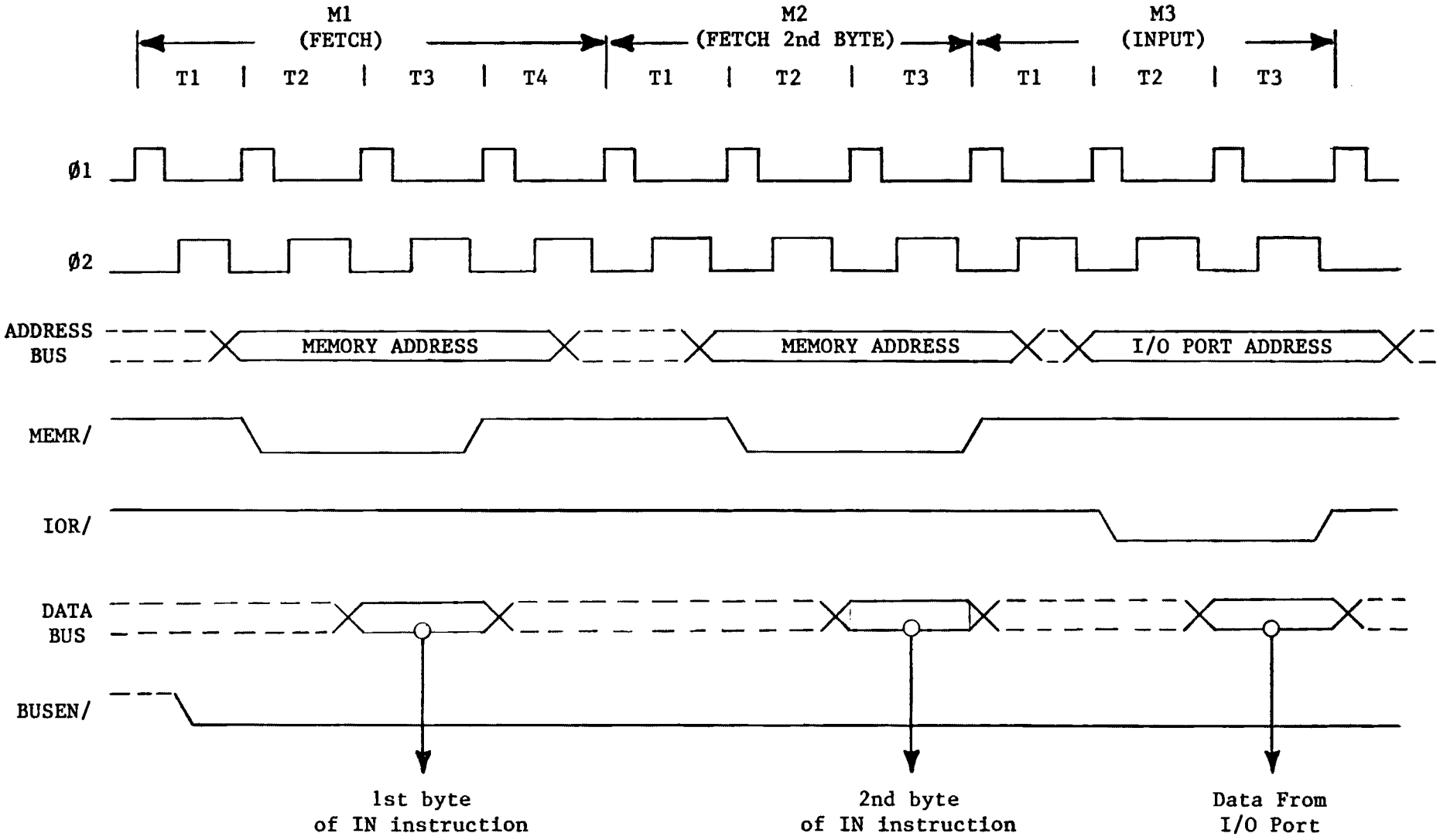


FIGURE 3-4. INPUT INSTRUCTION CYCLE

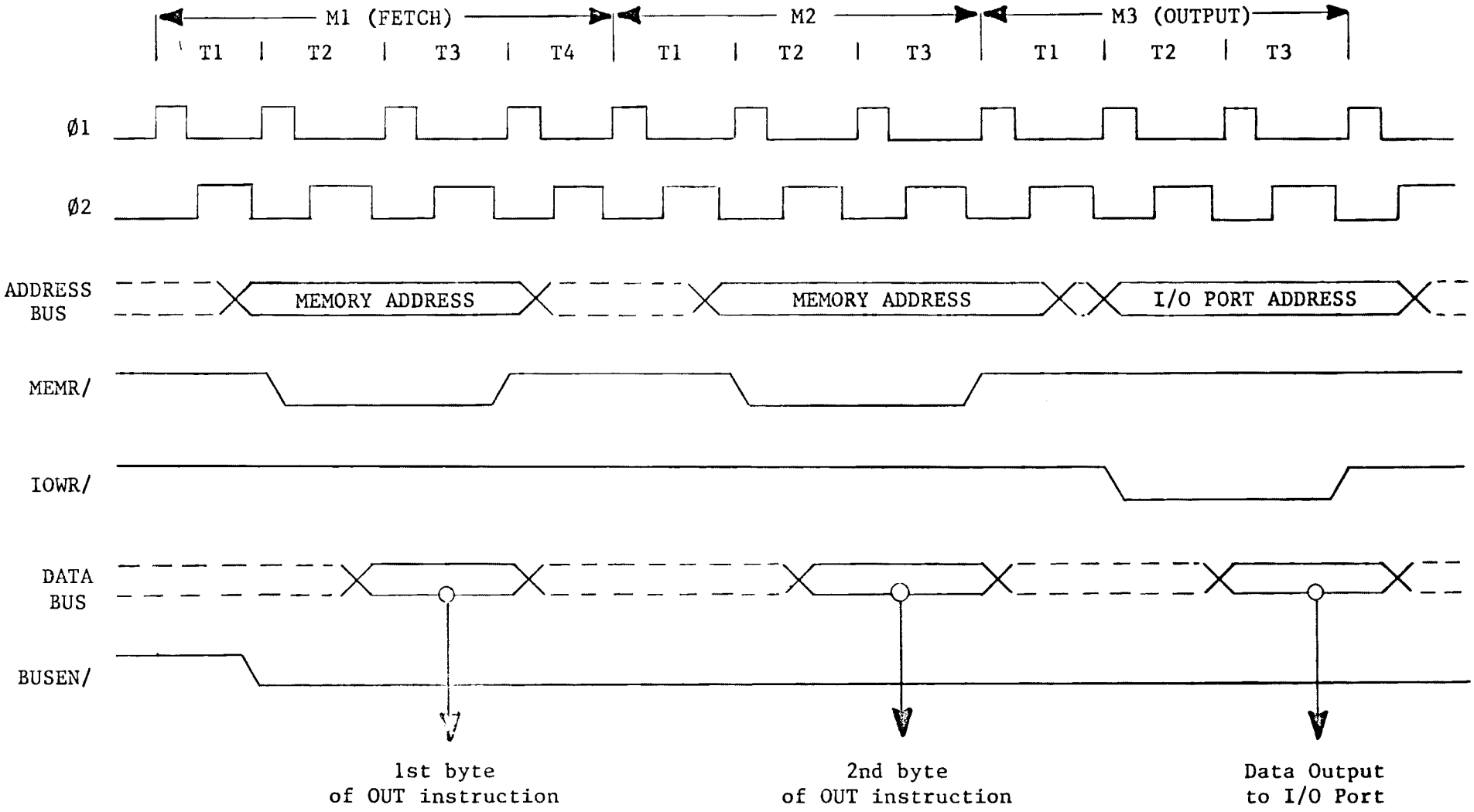


FIGURE 3-5. OUTPUT INSTRUCTION CYCLE

During the three basic states, data is transferred to or from the CPU Set.

After the T3 state, however, it becomes difficult to generalize. T4 and T5 states are available, if the execution of a particular instruction requires them. But not all machine cycles make use of these states. It depends upon the kind of instruction being executed, and on the particular machine cycle within the instruction cycle. The processor will terminate any machine cycle as soon as its processing activities are completed, rather than proceeding through the T4 and T5 states every time. Thus the 8080 may exit a machine cycle following the T3, the T4, or the T5 state and proceed directly to the T1 state of the next machine cycle.

3.2.1.2 INTERRUPT SEQUENCES

The 8080 has the built-in capacity to handle external interrupt requests. Peripheral logic can initiate an interrupt simply by driving the processor's interrupt (INT) line high. The interrupt (INT) input is asynchronous, and a request may therefore originate at any time during any instruction cycle. Internal logic re-clocks the external request, so that a proper correspondence with the driving clock is established. An interrupt request (INT) arriving during the time that the interrupt enable line (INTE) is high, acts in coincidence with the Ø2 clock to set the internal interrupt latch. This event takes place during the last state of the instruction cycle in which the request occurs, thus ensuring that any instruction in progress is completed before the interrupt can be processed.

The INTERRUPT machine cycle which follows the arrival of an enabled interrupt request resembles an ordinary FETCH machine cycle in most respects. The contents of the program counter are latched onto the address lines during T1, but the counter itself is not incremented during the INTERRUPT machine cycle, as it otherwise would be. In this way, the pre-interrupt status of the program counter is preserved, so that data in the counter may be saved in the stack. This in turn permits an orderly return to the interrupted program after the interrupt request has been processed.

Because the 8238's INTA/ output (pin 23) is tied to +12 volts, the 8238 blocks incoming data and automatically inserts a Restart (RST 7) instruction onto the 8080 data bus during state T3, when the interrupt is acknowledged by the 8080. RST is a special one-byte call instruction that facilitates the processing of interrupts (the ordinary program call instruction is three bytes long). The RST 7 instruction causes the 8080 to branch program control to the instruction being stored in memory location 38_{16}.

3.2.1.3 HOLD SEQUENCES

By activating the 8080's HOLD input, an external device can cause the CPU Set to suspend its normal operations and relinquish control of the address and data busses. The CPU Set responds to a request of this kind by floating its address and data outputs, so that these exhibit a high impedance to other devices sharing the busses. At the same time, the processor acknowledges the HOLD by placing a high on its HLDA output pin. During an acknowledged HOLD, the address and data busses are under control of the peripheral which originated the request, enabling it to conduct memory transfers without processor intervention.

3.2.1.4 HALT SEQUENCES

When a halt instruction (HLT) is executed, the 8080 enters the halt state after state T2 of the next machine cycle. There are only three ways in which the 8080 can exit the halt state:

- A high on the 8224 reset input (RESIN/) will always reset the 8080 to state T1; reset also clears the program counter.
- A HOLD input will cause the 8080 to enter the hold state, as previously described. When the HOLD line goes low, the 8080 re-enters the halt state on the rising edge of the next Ø1 clock pulse.
- An interrupt (i.e., INT goes high while INTE is enabled) will cause the 8080 to exit the halt state and enter state T1 on the rising edge of the next Ø1 clock pulse. NOTE: The interrupt enable (INTE) flag must be set when the halt state is entered; otherwise,

the 8080 will only be able to exit via a reset signal.

3.2.1.5 START-UP SEQUENCE

When power is applied initially to the 8080, the processor begins operating immediately. The contents of its program counter, stack pointer, and the other working registers are naturally subject to random factors and cannot be specified. For this reason, the CPU Set power-up sequence begins with a reset. An external RC network is connected to the 8224's RESIN/ input. The slow transition of the power supply rise is sensed by an internal Schmitt Trigger which converts the slow transition into a clean, fast edge on the RESIN/ line when the input level reaches a predetermined value.

An active RESIN/ input to the 8224 produces a synchronized RESET signal which restores the processor's internal program counter to zero. Program execution thus begins with memory location zero, following a reset. Systems which require the processor to wait for an explicit start-up signal will store a halt instruction (HLT) in this location. A manual or an automatic INTERRUPT will be used for starting. In other systems, the processor may begin executing its stored program immediately. Note, however, that the reset has no effect on status flags, or on any of the processor's working registers (accumulator, indices, or stack pointer). The contents of these registers remain indeterminate, until initialized explicitly by the program.

In addition to generating a RESET signal, the RESIN/ input causes the 8224's status strobe (STSTB/) output to remain true (low). This allows both the 8080 and 8238 to be reset by a power-up sequence or an externally generated RESIN/ condition.

3.2.2 SYSTEM BUS INTERFACE LOGIC

The System Bus Interface logic consists of three general groups of circuitry:

1) assorted gates that accept the various bus control signals, the interrupt request lines, the ready indications and then applies

these signals to the CPU Set,

2) the system bus drivers, and

3) the Failsafe circuitry which generates an acknowledgment during interrupt sequences and during those cycles in which an acknowledgment is not returned because a non-existent device was inadvertently addressed.

Each group is described in the following paragraphs.

3.2.2.1 SYSTEM CONTROL SIGNAL LOGIC

Interrupt Requests:

Four interrupt request lines are OR'd together at A17-6 (ref. Appendix B) and applied to the 8080's INT input. Two of the interrupt request lines are from external sources: EXT INTR 1/ which enters the SBC 80/10 at connector J1 pin 49 and EXT INTR 2/ which enters the SBC 80/10 at P1-42. The other two interrupt requests originate on the SBC 80/10: INT 55/ is an interrupt request from ports 1 or 2 in the Parallel I/O Interface (see Section 3.2.6.2); and INT 51/ is an interrupt request from the 8251 USART in the Serial I/O Interface (see Section 3.2.5.4).

Hold Requests:

If the system 80/10 is operating with other modules sharing the bus, one of the modules can acquire control of the external bus by activating the 8080's HOLD/ input (connector pin P1-15). HOLD/ is inverted and applied to the 8080's HOLD pin. As described in Section 3.2.1.3, the 8080 will subsequently activate its hold acknowledge (HLDA) output. HLDA is, in turn, latched by a 74LS74 flip-flop (at A29). The Q output from the D-type latch (DHLDA) disables the 8097 circuits (A47) that drive the external read/write control outputs: MRDC/, MWTC/, IORC/ and IOWC/. DHLDA also disables the external system address and data bus drivers by asserting a high at their active-low chip select (CS/) input pins. As a result of DHLDA, all of the above-mentioned drivers enter the high-impedance state. The $\overline{Q}$ output from the DHLDA output informs other modules of this condition via the BUSY/ output (connector pin P1-17). BUSY/ is driven by transistor Q5.

System Reset:

Connector pin P1-14 on the SBC 80/10 can be used to accept an externally generated SYSTEM RESET signal and to transfer an SBC 80/10 generated RESET signal to other modules in the system. If jumper pair 54-55 is connected, a RESET from the 8224 will be gated through the Q4 transistor to connector pin P1-14, thus resetting other modules in the system during power-up sequences. An externally generated SYSTEM RESET is accepted at P1-14, buffered, applied to the 8080's RESET input and made available to other logic on the SBC 80/10.

I/O Ready Generation

During each serial or parallel I/O cycle, a "ready" indication (IORDYIN/) is returned to the CPU Set. The three chip select lines for the 8251 and the two 8255 devices are OR'd together (at A17-8 on sheet 3 of the schematic). The resultant output is then NANDed (at A44-11) with the I/O read (IOR) or the advanced I/O write (ADV IOW) signal to produce IORDYIN/. Recall from Section 3.2.1 that the 8238 System Controller (in the CPU Set) generates the I/O write control output at the beginning of all I/O write cycles. The IOW/ signal, alone, is labeled ADV IOW/. IOW/ is also synchronized with the 8080's WR/ output to produce the system write command IOWC/. ADV IOW/ allows the ready indication to be returned early enough to avoid an unnecessary wait state (see Figure 3-6). The IOWC/ signal causes an I/O device to actually write the data, later in the I/O cycle.

Ready Inputs:

Recall from Section 3.2.1.1 that the CPU Set must see a ready indication before proceeding to internal state T3 during all machine cycles. The 74S20 section at A57 on sheet 1 of the schematic ORs the following ready indications:

1) INT ACK/ or TIME OUT ACK/ from the Failsafe logic (see Section 3.2.2.3),
2) IORDYIN/ from the Serial and Parallel I/O Interfaces,
3) PROM RDYIN/ from the ROM/PROM logic (see Section 3.2.4), and
4) RAM RDYIN/ from the RAM section (see Section 3.2.3).

The resultant output indicates an on-board memory or I/O access and is used to disable the external data bus drivers at A53 and A54. This output from A57-8 is also OR'd (at A30-3) with the externally generated AACK/ (connector pin P1-25) and XACK/ (connector pin P1-23) inputs. The output from A30-3 is then applied to the CPU Set's RDYIN input (pin 3 on the 8224). When the SBC 80/10 CPU Set accesses an external module, the AACK/ or XACK/ input informs the CPU Set that the external device is ready. AACK/ is an advanced acknowledge that allows certain OEM modules to be accessed faster.

Figure 3-6 illustrates basic timing for the ready indications.

Bus Clock Generation:

The OSC output from the CPU Set (18.432 MHz frequency) is applied to the clock input of a 74LS74 D-type flip-flop (at A29-11 on sheet 1 of the schematic). The $\overline{Q}$ output from this latch is tied to its own D input. Consequently, the Q output exhibits half the frequency of the OSC input. This 9.216 MHz output is buffered and made available to external modules on the common clock (CCLK/) line (via connector pin P1-31) and the bus clock (BCLK/) line (via connector pin P1-13).

3.2.2.2 SYSTEM BUS DRIVERS

The SBC 80/10 internal memory data bus (DM0-DM7) is driven by two 8216 bidirectional bus drivers, shown at A55 and A56 on sheet 3 of the schematic. All data being transferred to/from the RAM memory (see Section 3.2.3) or ROM/PROM memory (see Section 3.2.4) is routed through these two devices. The chip select (CS/) input is provided by the MEM CMD/ signal which is the result of ORing RAM RDYIN/ and PROM RDYIN/. The direction enable (DIEN) input to the 8216s is provided by the memory read (MEMR) signal.

When the SBC 80/10 communicates with an external module, the data is driven by two 8226 bidirectional data bus drivers at A53 and A54 on sheet 1 of the schematic. The direction input to the 8226s is provided by the OR of memory read (MEMR) and I/O read (IOR). The 8226 devices will be disabled during 8080 HOLD sequences. The eight data bus lines to the 8226 bus drivers enter/leave the SBC 80/10 via the P1 edge connector.

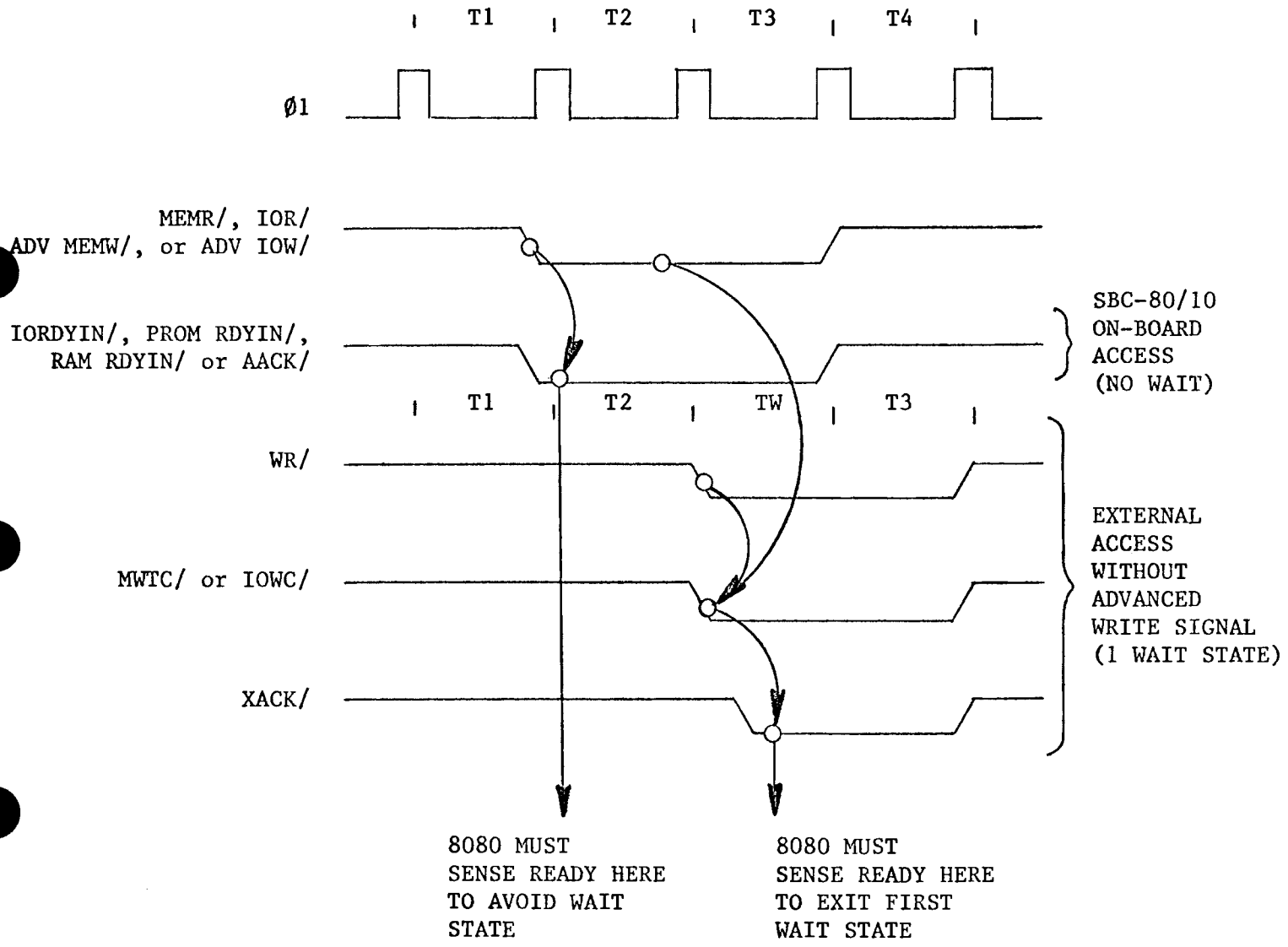


FIGURE 3-6 READY TIMING

The external 16-bit system address bus is driven by four 8226 bidirectional bus drivers. However, because the direction enable pin (EN/) on these 8226 devices is tied to ground, they can only be used to transmit addresses to external modules; they will not receive addresses from external modules. Consequently, the SBC 80/10 can access other modules, but other modules cannot access the memory or I/O controllers on the SBC 80/10. Like the data bus drivers, these 8226 devices are disabled during 8080 HOLD sequences.

3.2.2.3 FAILSAFE TIMER

When the 8080 acknowledges an interrupt request, the 8238 System Controller "forces" an RST 7 instruction onto the 8080s data bus (see Section 3.2.1.2). In order to read this RST 7 instruction, however, the 8080 must sense a ready indication. The 8080 acknowledges an interrupt by setting status bit 0 (D0) during the status output portion of each machine cycle (i.e., when STATUS STROBE is true). When this occurs, the 9602 one-shot (shown at A28 on sheet 5 of the schematic) is reset causing a low signal on its output (INTR ACK/). This output is then gated through to the RDYIN pin on the 8224 as described in Section 3.2.2.1.

The Failsafe timer also performs another function. If the CPU Set tries to access a memory or I/O device but that device, for some reason, does not return a ready indication, then the 8080 remains in a wait state until ready is received. The Failsafe timer is designed to prevent hanging the system up in this way. The 9602 one-shot is triggered by STATUS STROBE at the beginning of each machine cycle. If the one-shot is not re-triggered (i.e., if another cycle does not begin) within 9 ms., then the 9602 times out and its output (also labeled TIME OUT ACK/) is gated through to the RDYIN pin on the 8224, thus allowing the 8080 to exit the wait state. This can be very helpful during system debugging.

3.2.3 RANDOM ACCESS MEMORY

The Random Access Memory (RAM) provides the user with 1024 (1K) x 8-bits of read/write storage that requires no clocks or refresh to operate. The SBC 80/10 and SBC 80/10A utilize two different configurations, therefore each configuration is discussed separately in paragraphs 3.2.3.1 and 3.2.3.2.

3.2.3.1 SBC 80/10 RAM

The RAM logic consists of eight Intel 256 x 4-bit Static MOS RAM chips, an Intel 8205 three-to-eight decoder for chip selection and assorted gates as shown on sheet 2 of the SBC 80/20 schematic (Appendix B).

The RAM devices used on the SBC 80/10 have a maximum access time of 500 nsec. Each chip has eight address inputs (A0-A7) that select one of the 256 four-bit segments, active-low write (W/) and chip enable (CE/) inputs and an output disable (OD) input. Each chip also has four common data input/output pins (I/01-I/04). A high on the OD input disables output and allows the I/O pins to be used for input. During memory read accesses, the data is read out nondestructively and has the same polarity as the input data.

The least significant system address lines (ADR0-ADR7) are applied to the eight address input pins on each RAM. The most significant eight system address lines (ADR8-ADRF) feed 3205 decoder. Each of the four most significant decoder outputs are applied to the chip enable (CE/) inputs on two RAM chips. One RAM in each pair reads or writes data bits 0 to 3 (DM0-DM3) while the other RAM reads or writes data bits 4 to 7 (DM4-DM7) for each RAM access. One of the decoder outputs will be activated (low) whenever the value on the system address bus is within the range 3C00-3FFF (hexadecimal).

During memory write cycles, the advanced memory write signal (ADV MEMW/) is applied to the write input (W/) on each RAM. A high on the active-low memory read line (MEMR/) allows the selected RAM's I/O pins to be used to accept the data which is to be written into the addressed location. During memory read cycles, the level on ADV MEMW/ is high but is low on MEMR/ thus allowing the addressed data to be read out and onto the data bus.

During all RAM access cycles, the active decoder output is NANDed with ADV MEMW or MEMR (at A44-3) to produce a ready indication for the CPU Set (RAM RDYIN/). The 8238 System Controller (see Section 3.2.1) generates ADV MEMW or MEMR early enough in the memory cycle to allow RAM RDYIN/ to appear at the CPU Set in time to prevent the occurrence of any wait states. Figure 3-7 illustrates RAM access timing.

Whenever RAM is accessed, the data is transferred to/from the RAM chips on the memory data bus (DM0-DM7). Lines DM0-DM7 are interfaced to the system data bus through two Intel 8216 bidirectional bus drivers (shown at A55 and A56 on sheet 3 of the schematic) as described in Section 3.2.2.

3.2.3.2 SBC 80/10A RAM

The RAM logic consists of eight Intel 8102 1024 x 1-bit Low Power Static RAM chips, an Intel 3205 three-to-eight decoder, and assorted gates as shown on sheet 2 of the SBC 80/10A schematic (Figure B-3).

The 8102 RAM devices used on the SBC 80/10A have a maximum access time of 450 nsec. Each RAM chip has ten address inputs (ADR0-ADR9) that select one of the 1024 bits, an active low write (ADV MEM W/) and chip enable. A high on the ADV MEM W/ input allows a memory read access.

The ten least significant address lines (ADR0-ADR9) are applied to the ten address input pins on each 8102 RAM. The six most significant address lines (ADRA-ADRF) feed a 3205 decoder. The output of the 3205 decoder is applied to each Chip Enable/ (CE/) input to the eight 8102 RAM's. When the value on the system address bus is within the range 3C00-3FFF the decoder output will be activated (low).

During all RAM access cycles, the active decoder output produces a ready indication for the CPU set (RAM RDY IN/). The 8238 System Controller (see Section 3.1) generates ADV MEM W/ or MEM R/ early enough in the memory cycle to allow RAM RDY IN/ to appear at the CPU set in time to prevent the occurrence of any wait states. Figure 3-7 illustrates RAM access timing.

Whenever SBC 80/10A RAM is accessed, the data is transferred to/from the RAM chips on the memory data bus (DM0-DM7). Lines DM0-DM7 are interfaced to the system data bus through two Intel 8216 bidirectional bus driver (shown at A55 and A56 on sheet 3 of the schematic) as described in Section 3.2.2.

3.2.4 READ-ONLY-MEMORY (ROM/PROM)

The System 80/10 has provisions for installing 4096 (4K) x 8-bit words of read only memory in sockets already on the PC board. Four Intel 8708 1K x 8-bit Erasable and Electrically Reprogrammable Read Only Memory (EPROM) chips or four 8308 1K by 8-bit Metal Masked Read Only Memory (ROM) chips can be installed in the four 24-pin sockets shown on sheet 3 of the schematic (Appendix A). Optionally the SBC 80/10A has provisions for installing 4096 (4K) x 8-bits of read only memory in the sockets using four Intel 2758 1K x 8-bits Erasable and Electrically Reprogrammable Read Only Memory (EPROM) chips or installing 8192 (8K) x 8-bit words of read only memory using either Intel 2716 2K x 8-bit Erasable and Electrically Re-

programmable Read Only Memory (EPROM) chips or Intel 2316E 2K x 8-bit Metal Masked Read Only Memory (ROM) chips.

When addressing up to 4K of ROM address lines ADR0-ADR9 are applied to the address pins A0-A9 at each of the four sockets. The remaining address lines, ADRA-ADRF are decoded by the 3205 device at A42. Each of the four least significant decoder outputs are applied to the chip select (CS/) pin at one of four sockets. One chip select line will be activated whenever the value on the system address bus is between 0000 and 0FFF (hexadecimal). In addition, when the four most significant address lines are low (i.e., the address is less than 0FFF) during a memory read cycle, the output from the 74LS00 section at A39-3 is NANDed with MEMR to produce a ready indication (PROM/ RDYIN/) for the CPU Set. PROM RDYIN/ is thus generated in time to allow all ROM/PROM reads to occur without any wait states. PROM RDYIN/ has the same timing as RAM RDYIN/, as shown in Figure 3-7.

When using the optional 2716, or 2316E chips with the 80/10A, address lines ADR0-ADRA are applied to the address pins to each of the four sockets. The remaining address lines, ADRB-ADRF are decoded by the 3205 three-to-eight decoder. Each of the four least significant decoder outputs are applied to the Chip Select (CS/) pin at one of four sockets. One chip select line will be enabled when the value on the system address bus is between 0000 and 1FFF (hexadecimal).

In addition when the three most significant address lines are low (i.e. the address is less than 1FFF) during a memory read cycle, the output from the 74LS00 at A39-3 is NANDed with MEMR/ to produce a ready indication PROM RDY IN/ for the CPU set. PROM RDY IN/ is generated in time to allow all ROM/PROM reads to occur without any wait states. PROM RDY IN has the same timing as RAM RDY IN/, as shown in Figure 3-7.

Whenever one of the ROM/PROM devices are read, the data from the chip's output pins (01-08) is placed on the memory data bus (DM0-DM7) which is interfaced to the system bus via two Intel 8216 bidirectional bus drivers (at A55 and A56), as described in Section 3.2.2.

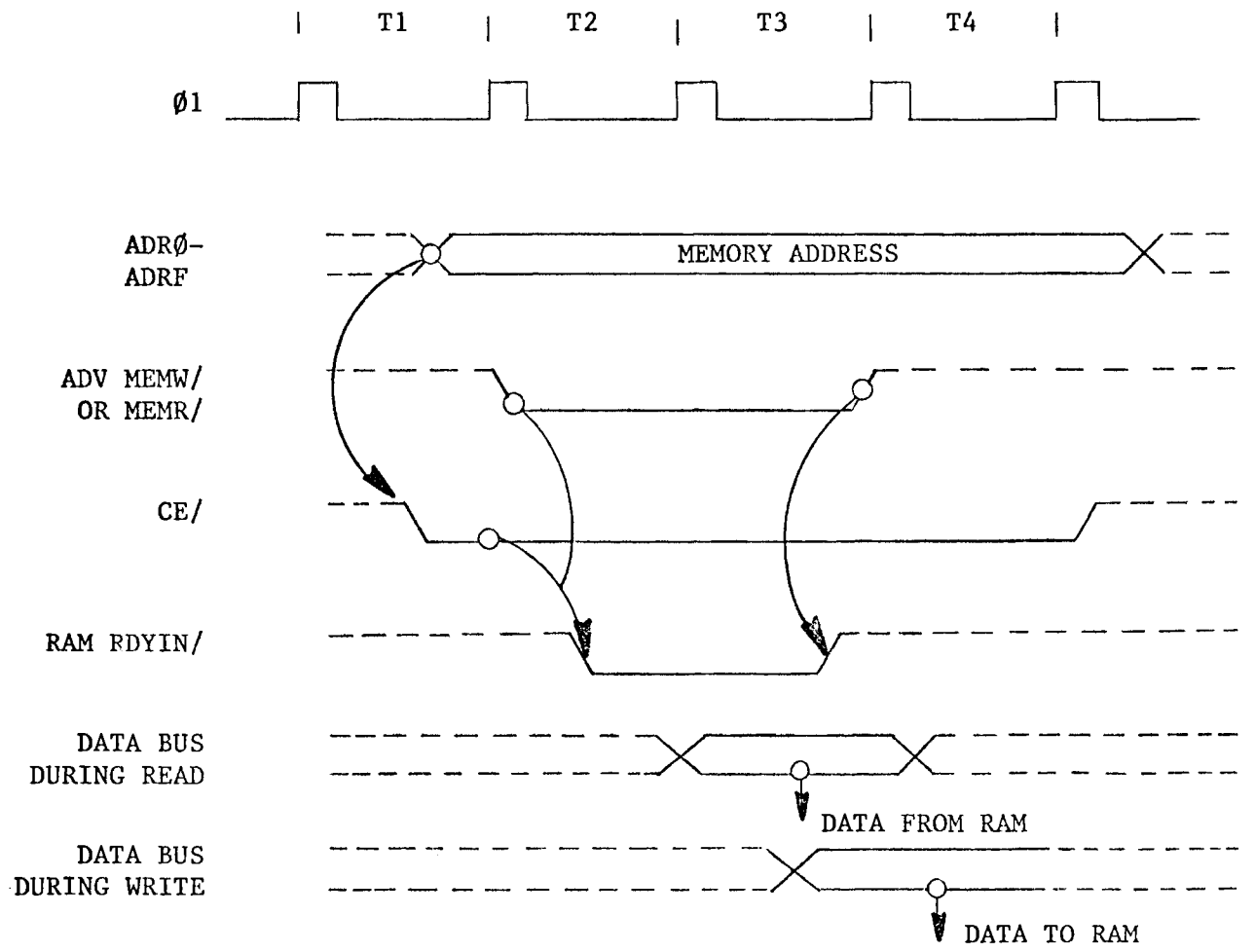


FIGURE 3-7 RAM ACCESS TIMING

3.2.5 SERIAL I/O INTERFACE

The Serial I/O Interface logic provides the System 80/10 with a serial data communications channel that can be programmed to operate with most of the current serial data transmission protocols, synchronous or asynchronous. Baud rate, character length, number of stop bits and even/odd parity are program selectable. In addition, the serial I/O Interface can be configured (through jumper connections) as an EIA RS232C interface or as a Teletype-compatible current loop interface.

The Serial I/O Interface logic consists primarily of an Intel 8251 USART device and a counting network for baud rate selection, as shown on sheet 4 of the SBC 80/10 schematic (Appendix B). Before describing the specific operation of the Serial I/O logic however, we will summarize the general operational characteristics of the 8251 USART, because it essentially defines the character of the Serial I/O Interface.

3.2.5.1 INTEL® 8251 OPERATIONAL SUMMARY

The 8251 is a Universal Synchronous/Asynchronous Receiver/Transmitter designed specifically for the 8080 Microcomputer System. Like other I/O devices in the 8080 Microcomputer System its functional configuration is programmed by the systems software for maximum flexibility. The 8251 can support virtually any serial data technique currently in use (including IBM "Bi-Sync").

Modem Control

The 8251 has a set of control inputs and outputs that can be used to simplify the interface to almost any Modem. The modem control signals are general purpose in nature and can be used for functions other than Modem control, if necessary.

$\overline{\text{DSR}}$ (Data Set Ready)

The $\overline{\text{DSR}}$ input signal is general purpose in nature. Its condition can be tested by the CPU using a Status Read operation. The $\overline{\text{DSR}}$ input is normally used to test Modem conditions such as Data Set Ready.

$\overline{DTR}$ (Data Terminal Ready)

The $\overline{DTR}$ output signal is general purpose in nature. It can be set "low" by programming the appropriate bit in the Command Instruction word. The $\overline{DTR}$ output signal is normally used for Modem control such as Data Terminal Ready or Rate Select.

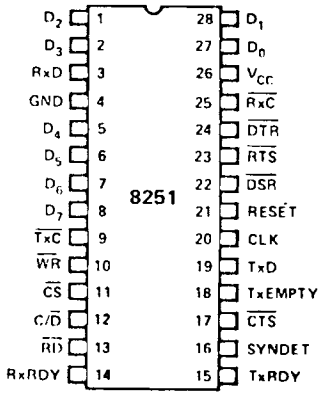
$\overline{RTS}$ (Request to Send)

The $\overline{RTS}$ output signal is general purpose in nature. It can be set "low" by programming the appropriate bit in the Command Instruction word. The $\overline{RTS}$ output signal is normally used for Modem control such as Request to Send.

$\overline{CTS}$ (Clear to Send)

A "low" on this input enables the 8251 to transmit data (serial) if the TxEN bit in the Command byte is set to a "one". This is very important to remember!

USART
PIN CONFIGURATION



Pin Name	Pin Function
D_7 D_0	Data Bus (8 bits)
C/D	Control or Data is to be Written or Read
RD	Read Data Command
WR	Write Data or Control Command
CS	Chip Enable
CLK	Clock Pulse (TTL)
RESET	Reset
TxC	Transmitter Clock
TxD	Transmitter Data
RxC	Receiver Clock
RxD	Receiver Data
RxRDY	Receiver Ready (has character for 8080)
TxRDY	Transmitter Ready (ready for char. from 8080)

Pin Name	Pin Function
DSR	Data Set Ready
DTR	Data Terminal Ready
SYNDET	Sync Detect
RTS	Request to Send Data
CTS	Clear to Send Data
TxE	Transmitter Empty
VCC	+5 Volt Supply
GND	Ground

FIGURE 3-8 8251 PIN ASSIGNMENTS

$\overline{TXRDY}$ (Transmitter Ready)

This output signals the CPU that the transmitter is ready to accept a data character. It can be used as an interrupt to the system or for

polled operation when the CPU can check TXRDY using a status read operation. $\overline{\text{TXRDY}}$ is active only when $\overline{\text{CTS}}$ is enabled. $\overline{\text{TXRDY}}$ is automatically reset when a character is loaded from the CPU.

TXE (Transmitter Empty)

When the 8251 has no characters to transmit, the TxE output will go "high". It resets automatically upon receiving a character from the CPU. TXE can be used to indicate the end of a transmission mode, so that the CPU "knows" when to "turn the line around" in the half-duplexed operational mode.

In synchronous mode, a "high" on this output indicates that a character has not been loaded and the SYNC character or characters are about to be transmitted automatically as "fillers".

$\overline{\text{TXC}}$ (Transmitter Clock)

The Transmitter Clock controls the rate at which the character is to be transmitted. In the synchronous transmission mode, the frequency of $\overline{\text{TXC}}$ is equal to the actual Baud Rate (1X). In asynchronous transmission mode, the frequency of $\overline{\text{TXC}}$ is a multiple of the actual baud rate. A portion of the mode instruction selects the value of the multiplier; it can be 1X, 16X or 64X the baud rate.

For example:

If Baud Rate equals 110 Baud,

$\overline{\text{TXC}}$ equals 110 Hz (1X)

$\overline{\text{TXC}}$ equals 1.76 kHz (16X)

$\overline{\text{TXC}}$ equals 7.04 kHz (64X).

If Baud Rate equals 9600 Baud,

$\overline{\text{TXC}}$ equals 614.4 kHz (64X).

The falling edge of $\overline{\text{TXC}}$ shifts the serial data out of the 8251.

RXRDY (Receiver Ready)

This output indicates that the 8251 contains a character that is ready to be input to the CPU. RXRDY can be connected to the interrupt structure of the CPU or for polled operation the CPU can check the con-

dition of RXRDY using a status read operation. RXRDY is automatically reset when the character is read by the CPU.

$\overline{\text{RXC}}$ (Receiver Clock)

The Receiver Clock controls the rate at which the character is to be received. In synchronous mode, the frequency of $\overline{\text{RXC}}$ is equal to the actual baud rate (1X). In asynchronous mode, the frequency of $\overline{\text{RXC}}$ is a multiple of the actual baud rate. A portion of the mode instruction selects the value of the multiplier; it can be 1X, 16X or 64X the baud rate.

For example:

If Baud Rate equals 300 Baud,

$\overline{\text{RXC}}$ equals 300 Hz (1X)

$\overline{\text{RXC}}$ equals 4800 Hz (16X)

$\overline{\text{RXC}}$ equals 19.2 kHz (64X).

If Baud Rate equals 2400 Baud,

$\overline{\text{RXC}}$ equals 2400 Hz (1X)

$\overline{\text{RXC}}$ equals 38.4 kHz (16X)

$\overline{\text{RXC}}$ equals 153.6 kHz (64X).

Data is sampled into the 8251 on the rising edge of $\overline{\text{RXC}}$.

Note: In most communications systems, the 8251 will be handling both the transmission and reception operations of a single link. Consequently, the Receive and Transmit Baud Rates will be the same. Both $\overline{\text{TXC}}$ and $\overline{\text{RXC}}$ will require identical frequencies for this operation and can be tied together and connected to a single frequency source (Baud Rate Generator) to simplify the interface.

SYNDET (SYNC Detect)

This pin is used in SYNCHronous Mode only. It is used as either input or output, programmable through the Control Word. It is reset to "low" upon RESET. When used as an output (internal Sync mode), the SYNDET pin will go "high" to indicate that the 8251 has located the SYNC character in the Receive mode. If the 8251 is programmed to use double Sync characters, then SYNDET will go "high" in the middle of the last bit of the second Sync character. SYNDET is automatically reset upon a

Status Read operation.

When used as an input, (external SYNC detect mode), a positive going signal will cause the 8251 to start assembling data characters on the falling edge of the next $\overline{RXC}$. Once in SYNC, the "high" input signal can be removed. The duration of the high signal should be at least equal to the period of $\overline{RXC}$.

Programming the 8251

Prior to starting data transmission or reception, the 8251 must be loaded with a set of control words generated by the CPU. These control signals define the complete functional definition of the 8251 and must immediately follow a Reset operation (internal or external).

The control words are split into two formats:

1. Mode Instruction,
2. Command Instruction.

Both the Mode and Command instructions must conform to a specified sequence for proper device operation. The Mode Instruction must be inserted immediately following a Reset operation, prior to using the 8251 for data communication.

All control words written into the 8251 after the Mode Instruction will load the Command Instruction. Command Instructions can be written into the 8251 at any time in the data block during the operation of the 8251. To return to the Mode Instruction format a bit in the Command Instruction word can be set to initiate an internal Reset operation which automatically places the 8251 back into the Mode Instruction format. Command Instructions must follow the Mode Instructions or Sync characters (see Figure 3-9).

Mode Instruction:

This format defines the general operational characteristics of the 8251. It must follow a Reset operation (internal or external). Once the Mode instruction has been written into the 8251 by the CPU, SYNC characters or Command instructions may be inserted.

The 8251 can be used for either synchronous or asynchronous data communications. The two least significant bits of the Mode Instruction control

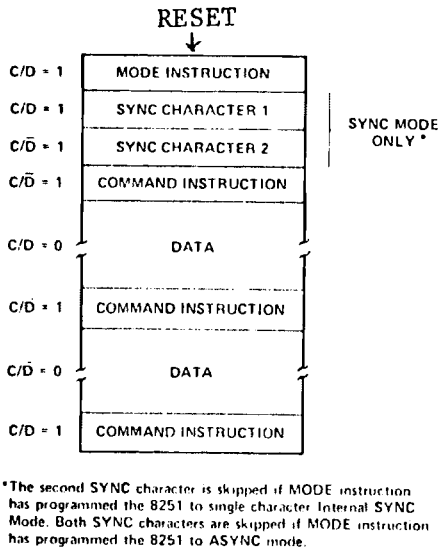


FIGURE 3-9 TYPICAL 8251 DATA BLOCK

word specify synchronous or asynchronous operation. The format for the remaining bits in the control word depends on the mode chosen by bits 0 and 1. Figure 3-10 shows the control word format for the asynchronous mode, while Figure 3-11 illustrates the control word format for the synchronous mode.

Command Instruction:

Once the functional definition of the 8251 has been programmed by the Mode Instruction and the Sync Characters are loaded (if in Sync Mode) then

the device is ready to be used for data communication. The Command Instruction controls the actual operation of the selected format. Functions such as: Enable Transmit/Receive, Error Reset and Modem Controls are provided by the Command Instruction.

Once the Mode instruction has been written into the 8251 and Sync characters inserted, if necessary, then all further "control writes" ($C/\overline{D}$ = 1) will load the Command Instruction. A Reset operation (internal or external) will return the 8251 to the Mode Instruction Format.

Figure 3-12 illustrate the format of a Command Instruction control word.

Status Read Definition

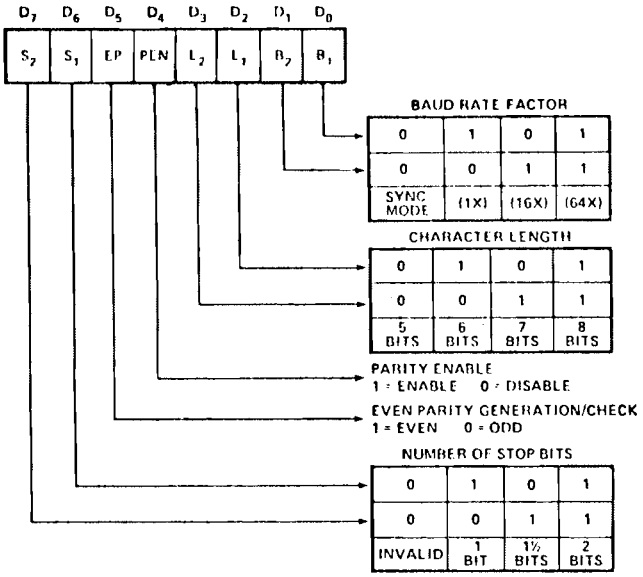
In data communication systems it is often necessary to examine the "status" of the active device to ascertain if errors have occurred or other conditions that require the processor's attention. The 8251 has facilities that allow the programmer to "read" that status of the device at any time during the functional operation.

A normal "read" command is issued by the CPU with the $C/\overline{D}$ input at one to accomplish this function.

Some of the bits in the Status Read Format have identical meanings to external output pins so that the 8251 can be used in a completely Polled environment or in an interrupt driven environment (refer to Figure 3-13).

8251 DATA TRANSFERS

Once programmed, the 8251 is ready to perform its communication functions. The TXRDY output is raised "high" to signal the CPU that the 8251 is ready to receive a character. This output (TXRDY) is reset automatically when the CPU writes a character into the 8251. On the other hand, the 8251 receives serial data from the MODEM or I/O device; upon receiving an entire character the RXRDY output is raised "high" to signal the CPU that the 8251 has a complete character ready for the CPU to fetch. RXRDY is reset automatically upon the CPU read operation.



Mode Instruction Format, Asynchronous Mode

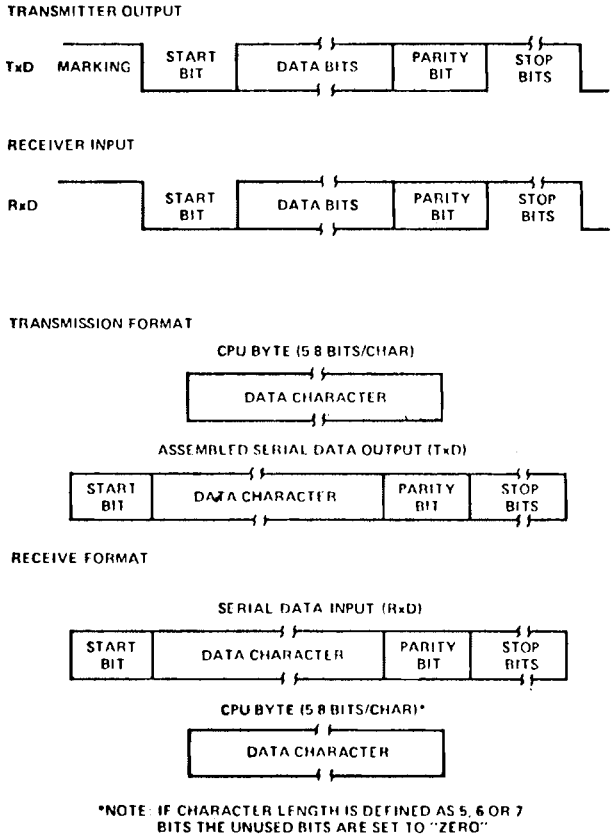
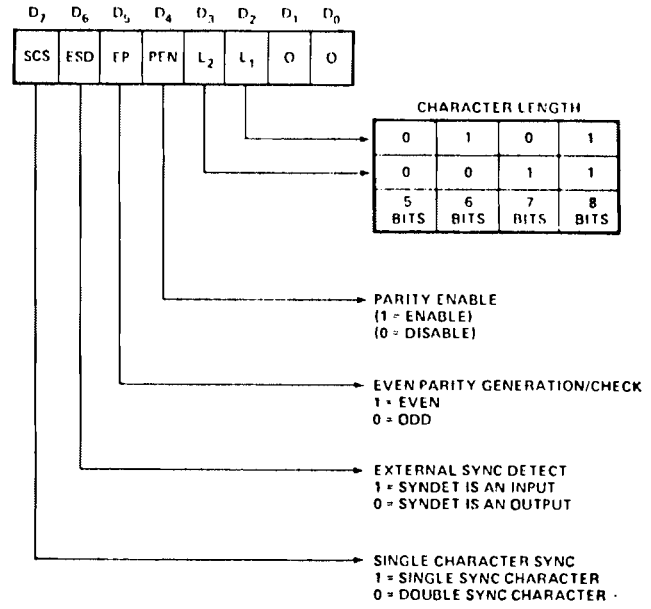


FIGURE 3-10 ASYNCHRONOUS MODE



Mode Instruction Format, Synchronous Mode

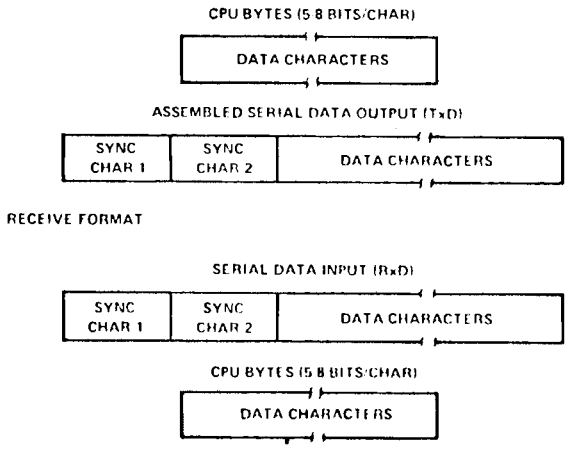


FIGURE 3-11 SYNCHRONOUS MODE

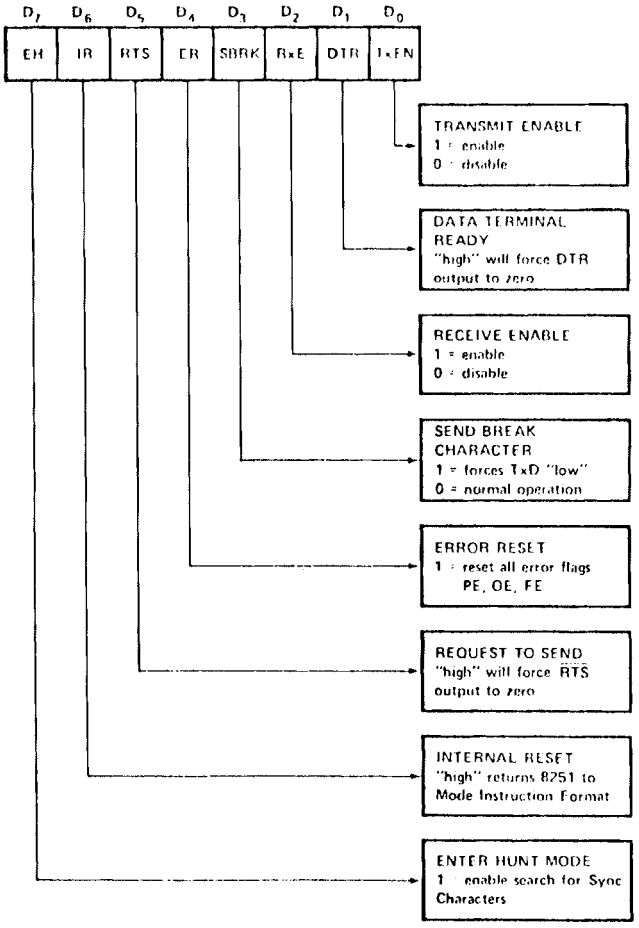


FIGURE 3-12 COMMAND INSTRUCTION FORMAT

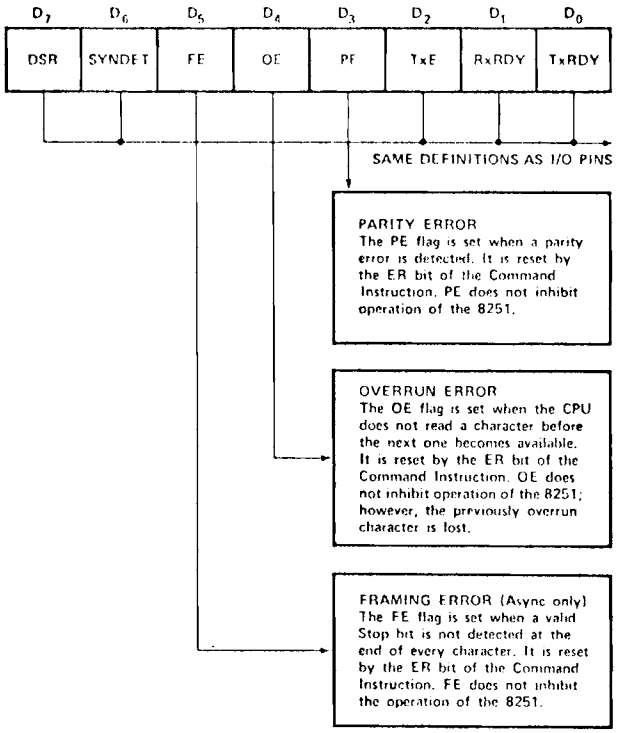


FIGURE 3-13 STATUS READ FORMAT

The 8251 cannot begin transmission until the TXEN (Transmitter Enable) bit is set in the Command Instruction and it has received a Clear To Send (CTS) input. The TXD output will be held in the marking state upon Reset.

Asynchronous Mode (Transmission):

Whenever a data character is sent by the CPU the 8251 automatically adds a Start bit (low level) and the programmed number of Stop bits to each character. Also, an even or odd Parity bit is inserted prior to the Stop bit(s), as defined by the Mode Instruction. The character is

then transmitted as a serial data stream on the TXD output. The serial data is shifted out on the falling edge of $\overline{TXC}$ at a rate equal to 1/16 or 1/64 that of the $\overline{TXC}$, as defined by the Mode Instruction. BREAK characters can be continuously sent to the TXD if commanded to do so.

When no data characters have been loaded into the 8251 the TXD output remains "high" (marking) unless a Break (continuously low) has been programmed.

Asynchronous Mode (Receive):

The RXD line is noramlly high. A falling edge on this line triggers the beginning of a START bit. The validity of this START bit is checked by again strobing this bit at its nominal center. If a low is detected again, it is a valid START bit, and the bit counter will start counting. The bit counter locates the center of the data bits, the parity bit (if it exists) and the stop bits. If parity error occurs, the parity error flag is set. Data and parity bits are sampled on the RXD pin with the rising edge of $\overline{RXC}$. If a low level is detected as the STOP bit, the Framing Error flag will be set. The STOP bit signals the end of a character. This character is then loaded into the parallel I/O buffer of the 8251. The RXRDY pin is raised to signal the CPU that a character is ready to be fetched. If a previous character has not been fetched by the CPU, the present character replaces it in the I/O buffer, and the OVERRUN flag is raised (thus the previous character is lost). All of the error flags can be reset by a command instruction. The occurrence of any of these errors will not stop the operation of the 8251.

Synchronous Mode (Transmission):

The TXD output is continuously high until the CPU sends its first character to the 8251 which usually is a SYNC character. When the $\overline{CTS}$ line goes low, the first character is serially transmitted out. All characters are shifted out on the falling edge of $\overline{TXC}$. Data is shifted out at the same rate as the $\overline{TXC}$.

Once transmission has started, the data stream at $\overline{TXD}$ output must continue at the $\overline{TXC}$ rate. If the CPU does not provide the 8251 with a

character before the 8251 becomes empty, the SYNC characters (or character if in single SYNC word mode) will be automatically inserted in the TXC data stream. In this case, the TXEMPTY pin will momentarily go high to signal that the 8251 is empty and SYNC characters are being sent out. The TXEMPTY pin is internally reset by the next character being written into the 8251.

Synchronous Mode (Receive):

In this mode, character synchronization can be internally or externally achieved. If the internal SYNC mode has been programmed, the receiver starts in a HUNT mode. Data on the RXD pin is then sampled in on the rising edge of RXC. The content of the RX buffer is continuously compared with the first SYNC character until a match occurs. If the 8251 has been programmed for two SYNC characters, the subsequent received character is also compared. When both SYNC characters have been detected, the USART ends the HUNT mode and is in character synchronization. The SYNDET pin is then set high, and is reset automatically by a STATUS READ.

In the external SYNC mode, synchronization is achieved by applying a high level on the SYNDET pin. The high level can be removed after one $\overline{\text{RXC}}$ cycle.

Parity error and overrun error are both checked in the same way as in the Asynchronous receive mode.

The CPU can command the receiver to enter the HUNT mode if synchronization is lost.

3.2.5.2 SERIAL I/O CONFIGURATIONS

The 8251 USART presents a parallel, eight-bit interface to the CPU set via the system data bus (DB0-DB7) and presents an EIA RS232C* or TTY current loop* interface to an external device (via edge connector J3). The 8251's interface with the CPU Set is enabled by a low level on its chip select (CS/) pin. CS/ is low when the I/O address on the system address bus is between EC and EF (hexadecimal). Address bits 2 through 7

* Electrical interfaces provided on SBC 80/10

are decoded (at A14) to produce the CS/ input. The least significant address bit, ADR0, is applied to the 8251s C/$\overline{D}$ input (pin 12) thus indicating a control (if set) or data (if reset) byte on the data bus.

I/O ADDRESS (BASE 16)	COMMAND	FUNCTION
ED OR EF	OUTPUT	CONTROL WORD
EC OR EE	OUTPUT	DATA
ED OR EF	INPUT	STATUS
EC OR EE	INPUT	DATA

TABLE 3-1 SERIAL COMMUNICATION (8251)
ADDRESS ASSIGNMENTS

An output instruction (IOW/ is true) to port ED or EF (CS/ is low and ADR0 is high) causes the 8251 to accept a control byte through its data bus pins. The control byte can be either a mode instruction or a command instruction, depending on the sequence in which it is sent. The various bits in the mode control word specify the baud rate multiplexer, character length, parity and the number of stop bits as described in Section 3.2.5.1. Note that the actual baud rate selected is dependent on the configuration of the baud rate jumper network (refer to Section 3.2.5.3). The various bits in the command control word instruct the USART to enable/disable the receiver and transmitter, to reset errors, to reset internal control and return to the mode control cycle, and to set/clear the Data Terminal Ready output.

An output instruction to port EC or EE (CS/ and ADR0 are low) causes the 8251 USART to accept a data byte through its data bus pins. Bit 0 is the least significant bit and bit 7 is the most significant bit. The 8251 will subsequently transmit the data byte (if the transmitter is enabled), in serial fashion, to the external device as described in Section 3.2.5.1.

An input instruction (IOR/ is true) to port ED or EF (CS/ is low and ADR0 is high) causes the 8251 USART to place a status byte onto the system bus. The status bits are the result of status and error checking functions performed within the USART (see Section 3.5.1).

An input instruction (IOR/ is true) to port EC or EE (CS/ and ADR0 are low) causes the USART to output a data byte (previously received from the external device) from its data bus pins. Bit 0 is the least significant bit and bit 7 is the most significant bit.

Timing for the USART's internal function is provided by the Ø2TTL signal (see Section 3.2.1.1). The USART is reset by the occurrence of a high level on the RESET line.

The 8251 USART transmits and receives serial data, synchronously or asynchronously, as described in Section 3.2.5.1. By jumper-connecting the 8251 pins to different external lines, the Serial I/O logic can present either a Teletype-compatible current loop interface or an EIA RS232C interface to an external device. If the TTY-compatible current loop interface is used, the connections listed in Table 3-5 are required (see Section 3.3).

If the EIA RS232C interface is used, the connections listed in Table 3-6 are required (see Section 3.3).

3.2.5.3 BAUD RATE CLOCK GENERATION

The baud rate clock network consists of a 93S16 'divide-by-15' counter, two 74161 'divide-by-16' counters and wire-wrap jumpers for baud rate clock selection. The 93S16 counter is driven by the oscillator output (OSC) from the CPU Set. The QD output from this counter, in turn, drives the two 74161 counters. The outputs from these counters, each providing a different clock frequency, are tied to jumper pins that

can be connected to the BAUD RATE CLK line. The available frequencies are listed in Table 3-9 (located in Section 3.3.2). Recall that the effective baud rate of the 8251 USART is also dependent on the state of the 8251's internal frequency divider and the mode of operation (refer to Section 3.2.5.1). The 8251 is capable of dividing the baud rate clock by 1, 16 or 64.

3.2.5.4 SERIAL I/O INTERRUPTS

The Serial I/O logic can be configured with different forms of an interrupt request mechanism. By connecting jumper pair 16-17 and disconnecting 15-16, the user can allow the 8251's Receiver Ready (RXRDY) output (pin 14) to generate an interrupt request INT51/) to the CPU Set. RXRDY goes high whenever the receiver enable bit of the command word has been set and the 8251 contains a character that is ready to be input to the CPU Set. The user can also choose to have the 8251's Transmitter Ready (TXRDY) or the Transmitter Empty (TXE) output activate the INT51/ interrupt request. If jumper pair 19-21 is connected, a high on TXRDY (pin 15) will activate INT51/. If jumper pair 18-19 is connected instead, an active TXE (pin 18) output will generate INT51/. TXE goes high when the 8251 has no characters to transmit. TXRDY is high when the 8251 is ready to accept a character from the CPU Set. Both TXE and TXRDY are enabled by setting the transmit enable bit of the command word Notice on the schematic that, if jumper pairs 19-20 and 15-16 are connected, Serial I/O interrupts are inhibited.

Upon receiving an interrupt, the program can determine the actual condition which is responsible for the interrupt (RXRDY, TXRDY or TXE) by reading the status of the 8251 device as described in Section 3.2.5.1. The interrupt request will be removed when the data is transferred to/ from the 8251, as required. Note that the TXE or TXRDY output will be high, and consequently maintain an interrupt request, during all idle periods, since the 8251's transmit buffer will remain empty. To disable the transmitter, and the resultant interrupt request, the program can issue a command instruction to the 8251 with the TXEN bit (bit 0) equal to zero (refer to Section 3.2.5.1). The transmitter should not be disabled

until TXE is high.

3.2.6 PARALLEL I/O INTERFACE

The Parallel I/O Interface logic on the SBC 80/10 provides forty-eight (48) signal lines for the transfer and control of data to or from peripheral devices. Eight lines have a bidirectional driver and termination network permanently installed. The remaining forty lines are uncommitted. Sockets are provided for the installation of active driver networks or passive termination networks. The optional drivers and terminators are installed in groups of four by insertion into the 14-pin sockets.

All forty-eight signal lines emanate from the I/O ports on two Intel 8255 Programmable Peripheral Interface devices, as shown on sheet 5 of the SBC 80/10 schematic (Appendix B). The two 8255 devices allow for a wide variety of I/O configurations. Before describing the possible configurations, however, we will summarize the general operational characteristics of the 8255 device.

3.2.6.1 INTEL®8255 OPERATIONAL SUMMARY

The 8255 contains three 8-bit ports (A, B and C). All can be configured in a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255.

Port A: One 8-bit data output latch/buffer and one 8-bit data input latch.

Port B: One 8-bit data input/output latch/buffer and one 8-bit data input buffer.

Port C: One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with Ports A and B.

The 8080 CPU dictates the operating characteristics of the ports by outputting two different types of control words to the 8255:

(1) mode definition control word (bit 7 = 1)

(2) port C bit set/reset control word (bit 7 = 0)

Bit 7 of each control word specifies its format, as shown in Figures 3-14 and 3-15, respectively.

Mode Selection

There are three basic modes of operation that can be selected by the system software:

Mode 0 - Basic Input/Output

Mode 1 - Strobed Input/Output

Mode 2 - Bidirectional Bus

When the RESET input goes "high" all ports will be set to the Input mode 0 (i.e., all 24 lines will be in the high impedance state). After the RESET is removed the 8255 can remain in the Input mode with no additional initialization required. During the execution of the system program, the other modes may be selected using a single OUTput instruction. This allows a single 8255 to service a variety of peripheral devices with a simple software maintenance routine.

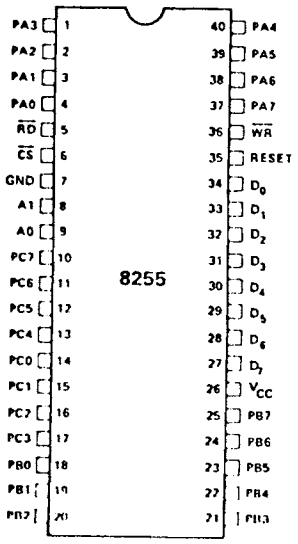
The modes for Port A and Port B can be separately defined, while Port C is divided into two portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed except for $\overline{OBF}$ in modes 1 and 2. Modes may be combined so that their functional definition can be "tailored" to almost any I/O structure. For instance; Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results; Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.

Single Bit Set/Reset Feature

Any of the eight bits of Port C can be Set or Reset using a single OUTput instruction (see Figure 3-16). This feature reduces software requirements in Control-based applications.

When Port C is being used as status/control for Port A or B, these

PIN CONFIGURATION



PIN NAMES

D_7-D_0	DATA BUS (BI DIRECTIONAL)
RESET	RESET INPUT
$\overline{CS}$	CHIP SELECT
$\overline{RD}$	READ INPUT
$\overline{WR}$	WRITE INPUT
A0, A1	PORT ADDRESS
PA7-PA0	PORT A (BIT)
PB7-PB0	PORT B (BIT)
PC7-PC0	PORT C (BIT)
V_{CC}	+5 VOLTS
GND	0 VOLTS

FIGURE 3-14 8255 PIN ASSIGNMENTS

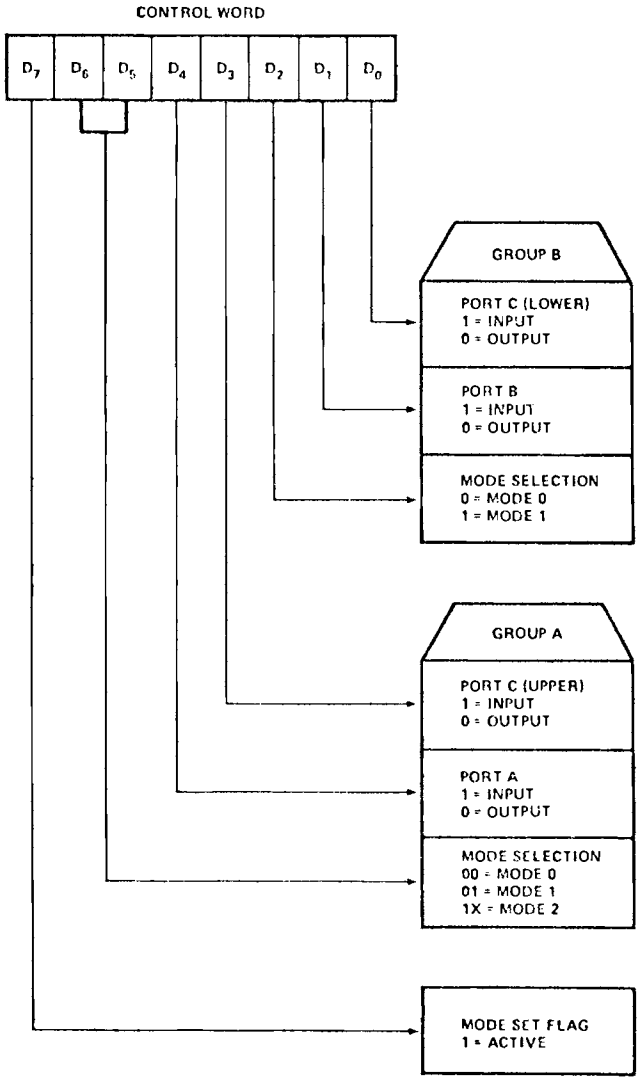


FIGURE 3-15 MODE DEFINITION CONTROL WORD FORMAT

bits can be set or reset by using the Bit Set/Reset operation just as if they were data output ports.

Interrupt Control Functions

When the 8255 is programmed to operate in Mode 1 or Mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from Port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the Bit set/reset function of Port C.

This function allows the Programmer to disallow or allow specific I/O devices to interrupt the CPU without effecting any other device in the interrupt structure.

INTE flip-flop definition:

(BIT-SET) - INTE is SET - Interrupt enable

(BIT-RESET) - INTE is RESET - Interrupt disable

Note: All Mask flip-flops are automatically reset during mode selection and device Reset.

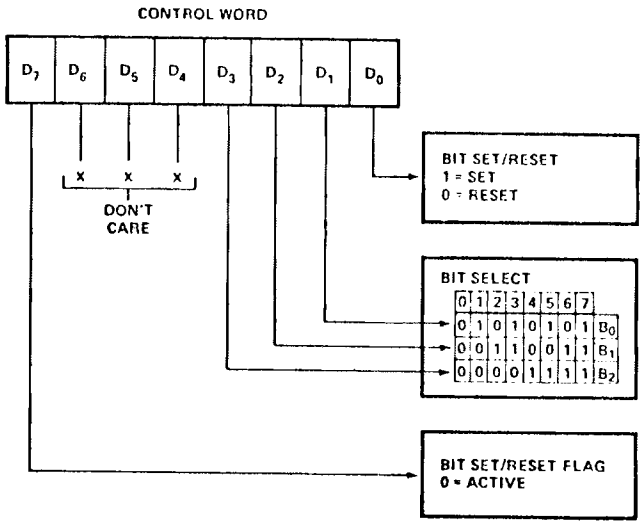


FIGURE 3-16 BIT SET/RESET CONTROL WORD FORMAT

Operating Modes

Mode 0 (Basic Input/Output):

This functional configuration provides simple Input and Output operations for each of the three ports. No "handshaking" is required, data is simply written to or read from a specified port. Mode 0 timing is illustrated in Figure 3-17.

Mode 0 Basic Functional Definitions:

- Two 8-bit ports and two 4-bit ports.
- Any port can be input or output.
- Outputs are latched.
- Inputs are not latched.

Sixteen different Input/Output configurations are possible in this Mode. Figure 3-18 shows two possible configurations.

Mode 1 (Strobed Input/Output):

This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "handshaking" signals. In Mode 1, Port A and Port B use the lines on Port C to generate or accept these "handshaking" signals.

Mode 1 Basic Functional Definitions:

- Two transfer ports (A and B).
- Each transfer port contains one 8-bit data port and 4 bits from one half of the control/data port (Port C).
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.

Input Control Signal Definition for Mode 1

$\overline{\text{STB}}$ (Strobe Input)

A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F)

A "high" on this output indicates that the data has been loaded into

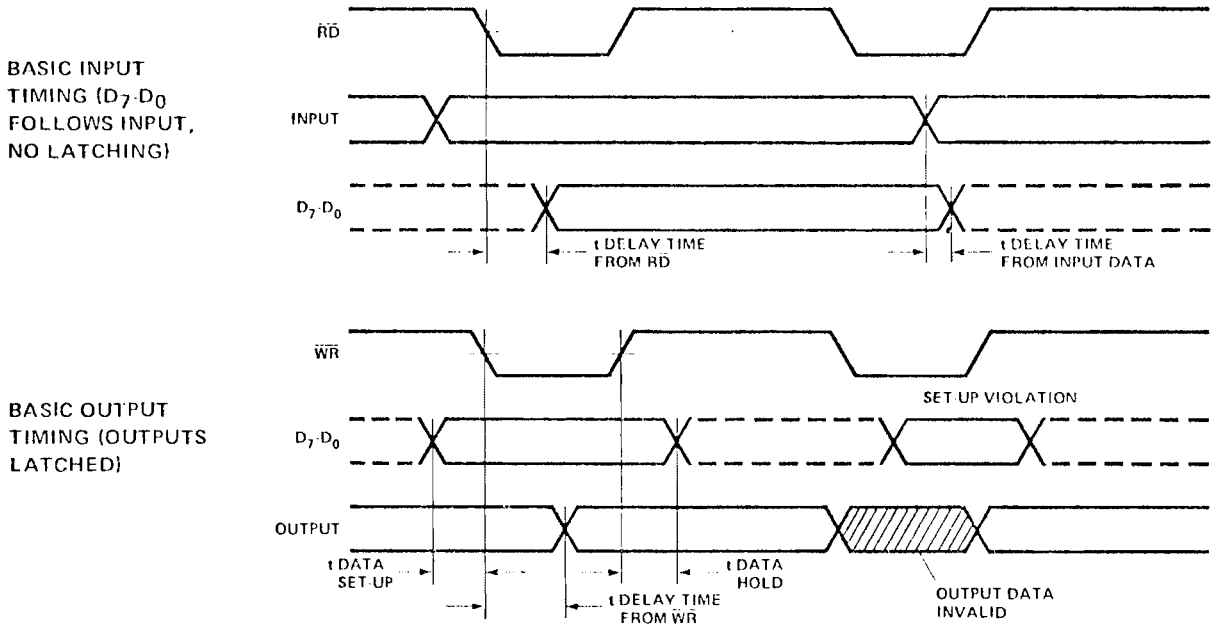


FIGURE 3-17 8255 MODE 0 TIMING

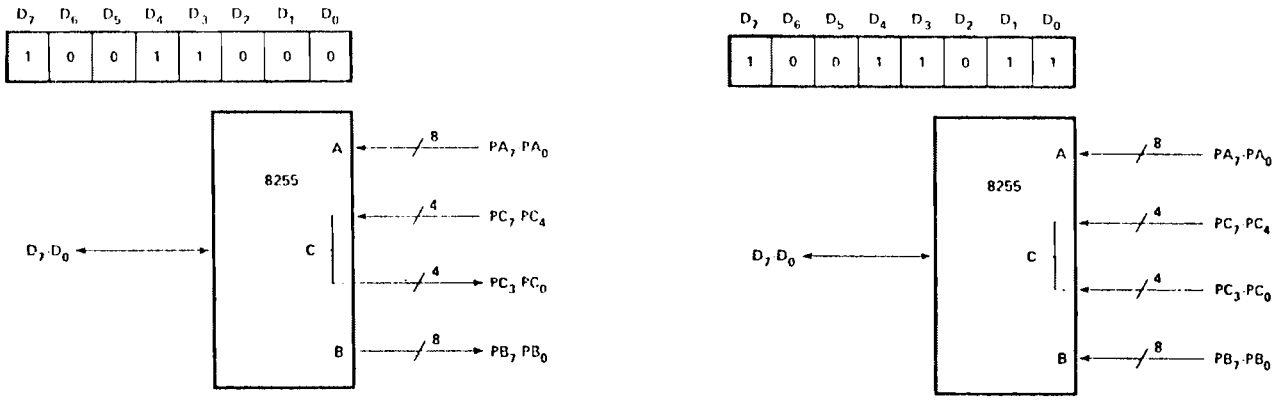


FIGURE 3-18 EXAMPLES OF MODE 0 CONFIGURATION

the input latch; in essence, an acknowledgment. IBF is set by the falling edge of the STB input and is reset by the rising edge of the $\overline{RD}$ input.

INTR (Interrupt Request)

A "high" on this output can be used to interrupt the CPU when an input device is requesting service. INTR is set by the rising edge of $\overline{STB}$ if IBF is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{RD}$. This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

INTE A

Controlled by bit set/reset of PC4.

INTE B

Controlled by bit/reset of PC 2.

Figure 3-19 illustrates the Mode 1 input configuration, while Figure 3-20 shows the basic timing for Mode 1 input.

Output Control Signal Definition for Mode 1

$\overline{OBF}$ (Output Buffer Full F/F)

The $\overline{OBF}$ output will go "low" to indicate that the CPU has written data out to the specified port. The OBF F/F will be set by the rising edge of the $\overline{WR}$ input and reset by the falling edge of the $\overline{ACK}$ input signal.

$\overline{ACK}$ (Acknowledge Input)

A "low" on this input informs the 8255 that the data from Port A or Port B has been accepted. In essence, a response from the peripheral device indicating that it has received the data output by the CPU.

INTR (Interrupt Request)

A "high" on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. INTR is set by the rising edge of $\overline{ACK}$ if $\overline{OBF}$ is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{WR}$.

INTE A

Controlled by bit/reset of PC6.

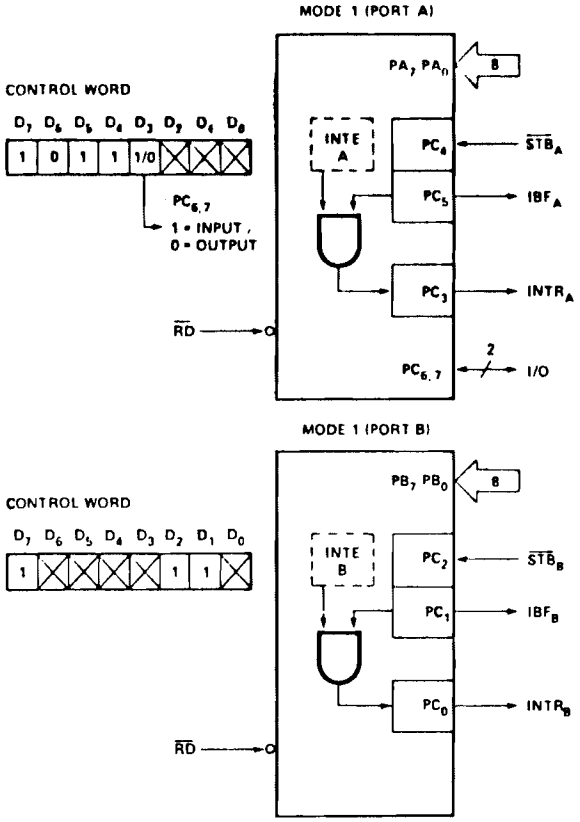


FIGURE 3-19 MODE 1 INPUT CONFIGURATION

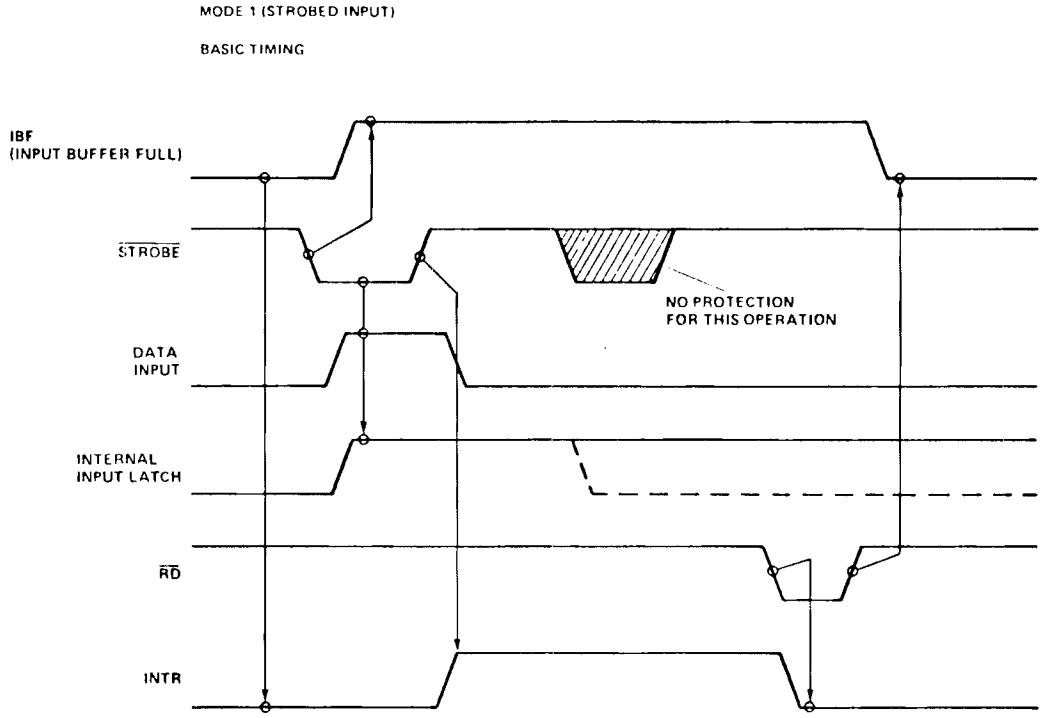


FIGURE 3-20 8255 MODE 1 INPUT TIMING

INTE B

Controlled by bit set/reset of PC2.

Figure 3-21 illustrates the Mode 1 output configuration, while Figure 3-22 shows basic Mode 1 output timing.

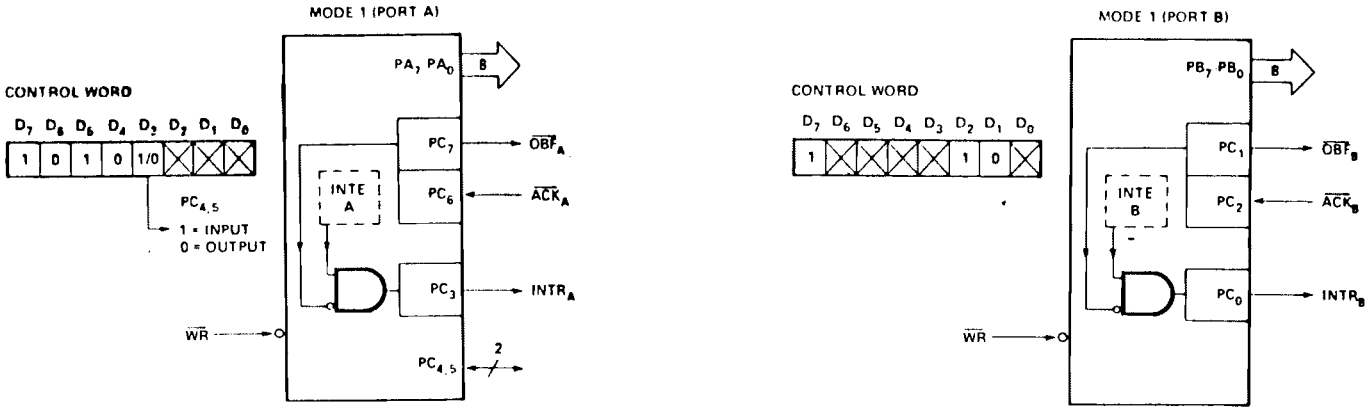


FIGURE 3-21 MODE 1 OUTPUT CONFIGURATION

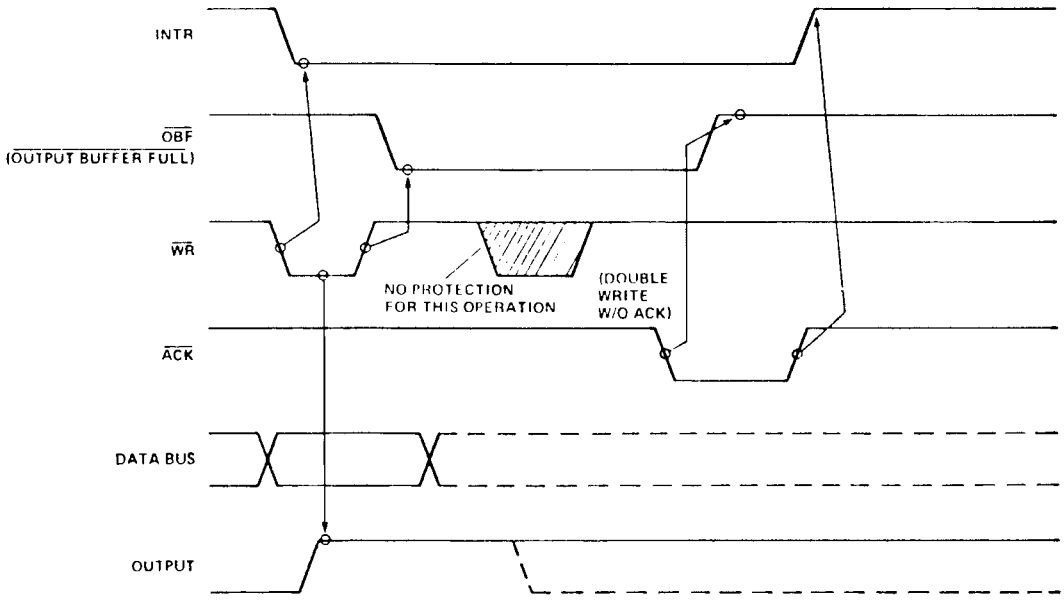


FIGURE 3-22 MODE 1 BASIC OUTPUT TIMING

Mode 2 (Strobed Bidirectional Bus I/O):

This functional configuration provides a means for communicating with a peripheral device or structure on a single 8-bit bus for both transmitting and receiving data (bidirectional bus I/O). "Handshaking" signals are provided to maintain proper bus flow discipline in a similar manner to Mode 1. Interrupt generation and enable/disable functions are also available.

Mode 2 Basic Functional Definitions:

- Used in Port A only.
- One 8-bit, bidirectional data Port (Port A) and a 5-bit control Port (Port C).
- Both inputs and outputs are latched.
- The 5-bit control port (Port C) is used for control and status for the 8-bit, bidirectional data port (Port A).

Bidirectional Bus I/O Control Signal Definition

INTR (Interrupt Request)

A high on this output can be used to interrupt the CPU for both input or output operations.

Output Operation Control Signals

$\overline{OBF}$ (Output Buffer Full)

The $\overline{OBF}$ output will go "low" to indicate that the CPU has written data out to Port A.

$\overline{ACK}$ (Acknowledge)

A "low" on this input enables the tri-state output buffer of Port A to send out the data. Otherwise, the output buffer will be in the high-impedance state.

INTR A and B (The INTE flip-flop associated with $\overline{OBF}$)

Controlled by bit set/reset of PC6 (INTE1)

Input Operation Control Signals

$\overline{STB}$ (Strobed Input)

A "low" on this output indicates that data has been loaded into the input latch.

INTE 2 (The INTE flip-flop associated with IBF)

Controlled by bit set/reset PC4 (INTE 2)

$INTR_A = PC6 \cdot OBF_A + PC4 \cdot INF_A$

Figure 3-23 illustrates the port configuration for Mode 2, Figure 3-24 shows Mode 2 timing, and Table 3-2 summarizes 8255 Mode definition.

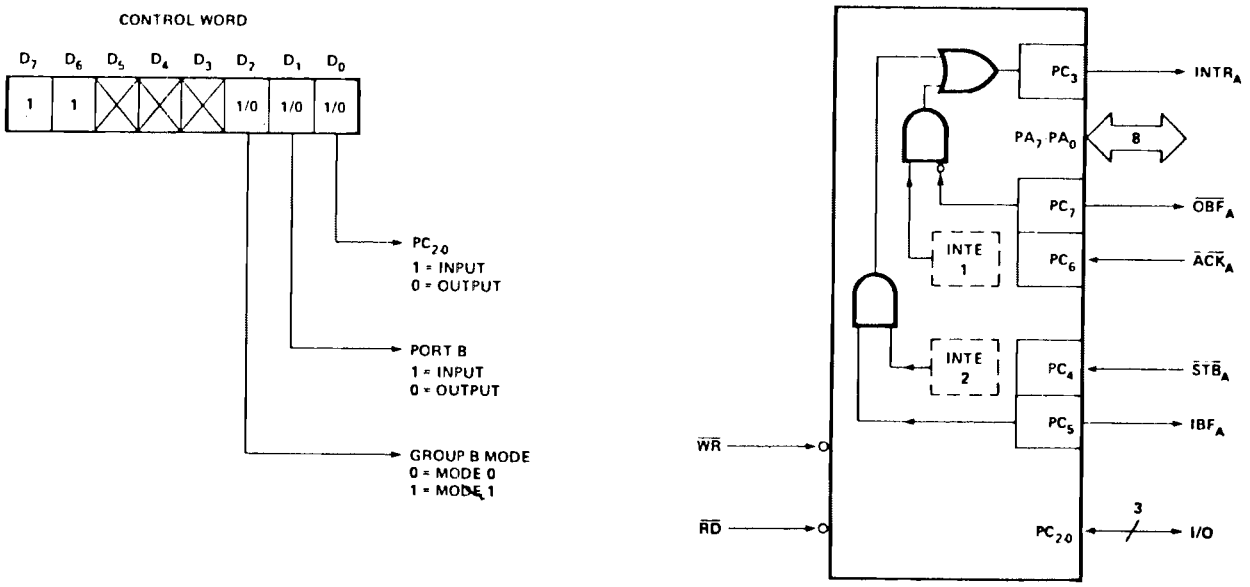


FIGURE 3-23 MODE 2 PORT CONFIGURATION

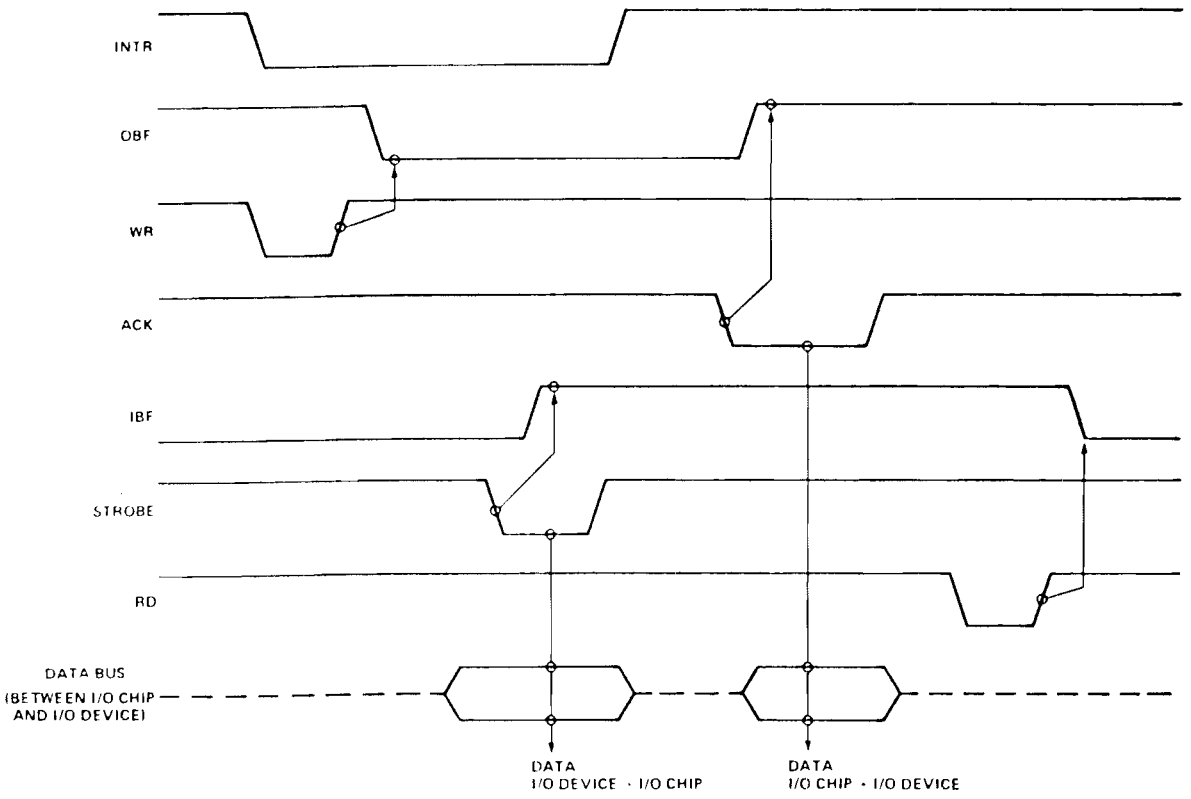


FIGURE 3-24 MODE 2 TIMING

MODE DEFINITION SUMMARY TABLE

	MODE 0		MODE 1		MODE 2	
	IN	OUT	IN	OUT	GROUP A ONLY	
PA_0	IN	OUT	IN	OUT	◄——►	
PA_1	IN	OUT	IN	OUT	◄——►	
PA_2	IN	OUT	IN	OUT	◄——►	
PA_3	IN	OUT	IN	OUT	◄——►	
PA_4	IN	OUT	IN	OUT	◄——►	
PA_5	IN	OUT	IN	OUT	◄——►	
PA_6	IN	OUT	IN	OUT	◄——►	
PA_7	IN	OUT	IN	OUT	◄——►	
PB_0	IN	OUT	IN	OUT	——	MODE 0 OR MODE 1 ONLY
PB_1	IN	OUT	IN	OUT	——	
PB_2	IN	OUT	IN	OUT	——	
PB_3	IN	OUT	IN	OUT	——	
PB_4	IN	OUT	IN	OUT	——	
PB_5	IN	OUT	IN	OUT	——	
PB_6	IN	OUT	IN	OUT	——	
PB_7	IN	OUT	IN	OUT	——	
PC_0	IN	OUT	$INTR_B$	$INTR_B$	I/O	
PC_1	IN	OUT	IBF_B	$\overline{OBF}_B$	I/O	
PC_2	IN	OUT	$\overline{STB}_B$	$\overline{ACK}_B$	I/O	
PC_3	IN	OUT	$INTR_A$	$INTR_A$	$INTR_A$	
PC_4	IN	OUT	$\overline{STB}_A$	I/O	$\overline{STB}_A$	
PC_5	IN	OUT	IBF_A	I/O	IBF_A	
PC_6	IN	OUT	I/O	$\overline{ACK}_A$	$\overline{ACK}_A$	
PC_7	IN	OUT	I/O	$\overline{OBF}_A$	$\overline{OBF}_A$	

TABLE 3-2 8255 MODE DEFINITION SUMMARY

3.2.6.2 PARALLEL I/O CONFIGURATIONS

Referring to sheet 5 of the schematic, we see that there are two 8255 devices, one located at A19, the other at A20. For convenience the following device designations will be used: The device at A19 is called the "group 1" device, while the device at A20 is referred to as the "group 2" device. Each device has three eight-bit ports. The "group 1" ports are designated Ports 1, 2 and 3 while the "group 2" ports are designated Ports 4, 5 and 6.

The group 1 and group 2 devices both communicate with the CPU Set using the same signal lines: the 8-bit data bus, DB0-DB7, and seven control/address lines; ADR0, ADR1, RESET, IOR/, IOW/, CS1/, and CS2/. The data lines bring control bytes or data bytes to an 8255 or deliver data from an 8255 to the CPU Set. The chip select control signals (CS1/ and CS2/) select the group 1 and group 2 devices, respectively, when the proper I/O address appears on the system address bus. CS1/ and CS2/ are the result of decoding address bits 1 through 7 (ADR2-ADR7), as shown on sheet 4 of the schematic (at A14). The two least significant address bits select the control register (when programming an 8255) or one of the three I/O ports (when reading or writing data). IOR/ (8255 → CPU Set) and IOW/ (CPU Set → 8255) indicate the direction of data flow, as summarized in Table 3-3. Specific I/O addresses for the six ports and two 8255 control registers on the SBC 80/10 are listed in Table 3-4.

A high on the RESET line clears all internal 8255 registers including the control register; all ports (A, B and C) are set for input.

Though both 8255's maintain the same interface (at different I/O addresses) with the CPU Set, the interface between the group 1 device and edge connector J1 is significantly different than the interface between the group 2 device and its associated edge connector (J2). This gives the user a great deal of flexibility when configuring the system's external parallel I/O devices. Because of those flexible "external" interfaces, however, not all ports are capable of operating in each 8255 mode, though all ports can be programmed as either input or output. The group 1 ports can fully utilize the 8255's multi-mode and external interrupt capabilities as described in Section 3.2.6.1. The group 2 ports, however, are limited

TABLE 3-3 8255 BASIC OPERATION

A1	A0	IOR/	IOW/	CS/	Input Operation (Read)
0	0	0	1	0	Port A → Data Bus
0	1	0	1	0	Port B → Data Bus
1	0	0	1	0	Port C → Data Bus
					Output Operation (Write)
0	0	1	0	0	Data Bus → Port A
0	1	1	0	0	Data Bus → Port B
1	0	1	0	0	Data Bus → Port C
1	1	1	0	0	Data Bus → Control
					Disable Function
x	x	x	x	1	Data Bus → High-Impedance
1	1	0	1	0	Illegal

TABLE 3-4 PARALLEL I/O PORT ADDRESSES

Port	8255 Device Location	*Eight-Bit Address (Hexadecimal)
1	8255 #1 Port (A)	E4
2	8255 #1 Port (B)	E5
3	8255 #1 Port (C)	E6
-	8255 #1 Control	E7 For I/O write only.
4	8255 #2 Port (A)	E8
5	8255 #2 Port (B)	E9
6	8255 #2 Port (C)	EA
-	8255 #2 Control	EB For I/O write only.

*Note: If address = 111001xx, CS1/ is activated.
If address = 111010xx, CS2/ is activated.

to a single mode of operation. The allowable port configurations for both groups are summarized below:

Port 1 (Group 1 Port A)

Mode 0 Input
Mode 0 Output (Latched)
Mode 1 Input (Strobed)
Mode 1 Output (Latched)
Mode 2 Bidirectional

Port 2 (Group 1 Port B)

Mode 0 Input
Mode 0 Output (Latched)
Mode 1 Input (Strobed)
Mode 1 Output (Latched)

Port 3 (Group 1 Port C)

Mode 0 8 Bit Input
Mode 0 8 Bit Output (Latched)

Note: Control mode dependent upon Port A and B mode.

Ports 4 and 5 (Group 2 Port A, B)

Mode 0 Input
Mode 0 Output (Latched)

Port 6 (Group 2 Port C)

Mode 0 8 Bit Input
Mode 0 8 Bit Output
Mode 0 4 Bit Input/4 Bit Output (Unlatched/latched)
Mode 0 4 Bit Output/4 Bit Input (Unlatched/latched)

Group 1

Port 1 is the most versatile of the six ports. It can be programmed to function in any one of the three 8255 operating modes. The first port is the only port that already includes a permanent bidirectional driver/termination network (two 8226 bus driver devices at A1 and A2).

Before Port 1 is programmed for input or output in any one of three operating modes (as described in Section 3.2.6.1), certain jumper connections must be made to allow the port to function properly in the chosen mode. The 40-41-42-43 jumper pad specifies the direction of data flow for the two 8226 bidirectional bus drivers. If input in mode 0 or mode 1 is

to be programmed for Port 1, jumper pair 41-42 should be connected. If output in mode 0 or mode 1 is to be used, jumper pair 40-41 should be connected. If Port 1 is to be programmed for bidirectional mode 2, then jumper pair 41-42 should be connected. This connection allows the output acknowledge, ACK/, that is input on bit 6 of Port 3 to dynamically dictate direction for the two 8226 devices.

Another jumper pad (48-49-50-51) enables interrupts for Port 1 when it is in mode 1 or mode 2. Jumper pair 49-50 should be connected to allow the INTR output (see Section 3.2.6.1) from bit 3 of Port 3 to activate an interrupt request (INT55/) from the 74LS02 gate at A45. In mode 0, during which there is no provision for interrupts, jumper pairs 48-49 and 50-51 must be connected to allow use of bit 3 of Port 3 and to inhibit Port 1 interrupts.

Because the 8226 bus drivers are inverting devices, all data input to or output from Port 1 is considered to be negative true with respect to the levels at the J1 edge connector.

Port 2 can be programmed for input or output in either mode 0 or mode 1 (see Section 3.2.6.1). If Port 2 is to be used for input (in either mode), terminator networks must be installed in the sockets at A5 and A6. Because these networks must be passive, data that is input to Port 2 will be positive true. If Port 2 is to be used for output (in either mode), driver networks must be installed in the sockets at A5 and A6. Assuming that the drivers are inverting devices, then the data being output will be negative true at the J1 edge connector.

When Port 2 is programmed for mode 1, interrupts can be enabled by connecting jumper pair 45-46. This connection allows the INTR output from bit 0 of Port 3 to activate the interrupt request (INT55/) to the CPU set. When Port 2 is in mode 0, jumper pairs 44-45 and 46-47 must be connected to allow use of bit 0 of Port 3 and to inhibit Port 2 interrupts.

As was described in Section 3.2.6.1, the use of Port 3 is dependent on the modes programmed for Ports 1 and 2. If Port 1 is in mode 1 or mode 2, bits 3, 4, 5, 6 and 7 of Port 3 can have dedicated control functions.

```
Port 3 bit 3 → INTR (interrupt request) - input or output
Port 3 bit 4 ← STB/ (input strobe)              } mode 1 input
Port 3 bit 5 → IBF  (input buffer full flag)   } or mode 2
Port 3 bit 6 ← ACK/ (output acknowledge)       } mode 1 output
Port 3 bit 7 → OBF/ (output buffer full flag)  } or mode 2
```

If Port 2 is in mode 1, bits 0, 1 and 2 of Port 3 have dedicated control functions:

```
Port 3 bit 0 → INTR (interrupt request) - input or output
Port 3 bit 1 → IBF  (input buffer full)  } input only
Port 3 bit 2 ← STB/ (input strobe)       }
Port 3 bit 1 → OBF/ (output buffer full) } output only
Port 3 bit 2 ← ACK/ (output acknowledge) }
```

While certain Port 3 bits are available if Port 1 is in mode 1 or if Port 2 is in mode 0, the use of Port 3 as an eight-bit data path is restricted to those configurations that have both Port 1 and Port 2 programmed for mode 0. In this case all 8 bits of Port 3 can be programmed for mode 0 input (termination networks must be installed in the sockets at A3 and A4) or output (driver networks must be installed at A3 and A4). Note: If Port 1 and 2 are not both in mode 0, then a driver network must be installed in the sockets at A3 and a termination network must be installed at A4, so that the Port 3 control lines can function properly.

Group 2

The three ports on the group 2 device can be programmed for input or output, but only in mode 0. If Port 4 is programmed for input, termination networks must be installed in the sockets at A7 and A8. The data being input will be in positive true form. If Port 4 is programmed for output, driver networks must be installed at A7 and A8. Assuming that inverting drivers are used, then the data will be considered negative true at the J2 edge connector.

If Port 5 is programmed for input, termination networks must be installed in the sockets at A21 and A11. If Port 5 is programmed for output, driver networks must be installed at A21 and A11.

All eight bits of Port 6 can be programmed for input or output, or four bits can be programmed for input while the other four bits are programmed for output (see Section 3.2.6.1). Driver termination networks must be

installed in the sockets at A9 and A10 as listed in Table 3-5.

TABLE 3-5. Port 6 I/O CONFIGURATIONS

	Sockets at A9	Sockets at A10
8-bit Input	Terminators*	Terminators*
8-bit Output	Drivers**	Drivers**
Upper 4-bits Input/ Lower 4-bits Output	Terminators*	Drivers**
Lower 4-bits Input/ Upper 4-bits Output	Drivers**	Terminators*

*Positive-true data.
**Negative-true data if inverting drivers.

In Section 3.3.2, all of the user options for configuring parallel I/O on the SBC 80/10 are summarized for convenient reference.

3.3 USER SELECTABLE OPTIONS

The SBC 80/10 provides the user with a powerful, but flexible I/O capability for both parallel and serial transfers. The serial I/O interface, using Intel's 8251 USART, provides a serial data communications channel that can be programmed to operate with most of the current serial data transmission protocols. Synchronous or asynchronous mode, character length, number of stop bits and even/odd parity are all program selectable. In addition, the user has the option, through jumper connections, of configuring the baud rate and the Serial I/O Interface as an ETA RS232C interface or as a Teletype-compatible current loop interface.

The Parallel I/O Interface, using two Intel 8255 Programmable Peripheral Interface devices, provides 48 signal lines for the transfer and control of data to or from peripheral devices. Eight lines already have a bidirectional driver and termination network permanently installed. The remaining 40 lines, however, are uncommitted. Sockets are provided for the installation of active driver networks or passive termination networks as required to meet the specific needs of the user system.

In this chapter, we will reiterate each of the options available to the user, and summarize, for easy reference, the specific information required to implement the user's tailored I/O configuration. Section 3.3.1 deals with the Serial I/O Interface, while Section 3.3.2 covers Parallel I/O options. Section 3.3.3 will describe general options not covered in the other two sections.

3.3.1 SERIAL I/O INTERFACE OPTIONS

There are three general areas of Serial I/O options:

(1) choice of interface type, RS232C or current loop,

(2) baud rate and program-selectable mode options,

(3) choice of an interrupt request mechanism.

The first two are covered in the following paragraphs; the third, choice of interrupt mechanism, is quite simple and is fully explained in Section 3.2.5.4.

3.3.1.1 INTERFACE TYPE

The user has the choice of configuring the Serial I/O logic to present either an EIA RS232C or a 20 mA current loop interface to an external device. If a Teletype-compatible current loop interface is used, the 8251 I/O pins should be connected to the external Teletype lines as listed in Table 3-6. The reader control logic is controlled by the output DSR/ from the 8251. If an EIA RS232C interface is used, the 8251 can assume the role of a "data set" (see Table 3-7) or a "data processing terminal" (see Table 3-8). Pin definitions for the 8251 USART are listed in Section 3.2.5.1.

3.3.1.2 BAUD RATE AND PROGRAM-SELECTABLE SERIAL I/O OPTIONS

Before beginning Serial I/O operations, the 8251 must be program-initialized to support the desired mode of operation. The CPU initializes the 8251 by outputting a set of control bytes to the USART device. These control words specify:

* synchronous or asynchronous operation,
* baud rate factor,
* character length,
* number of stop bits,
* even/odd parity,
* parity/no parity.

TABLE 3-6. 20 mA CURRENT LOOP SERIAL I/O INTERFACE

8251 PIN MNEMONIC	PIN NO.		CONNECTOR PIN NO.	JUMPER CONNECTIONS
TXD	19	TTY Tx	J3-25	1-2
DTR/	24	TTY RD CONTROL	J3-6	23-24
(1) RTS/	23	(CTS/)	-	27-29, 30-31
(1) CTS/	17	(RTS/)	-	27-29
(2) TXC	9	(Baud Rate Clk)	-	33-34 (8-4, 56-57)
(2) RXC	25	(Baud Rate Clk)	-	35-36 (8-4, 56-57)
TXD	3	TTY Rx	J3-22	38-39
-	-	TTY Rx RET	J3-23	-
-	-	TTY Tx RET	J3-24	-
-	-	TTY RD CTL RET	J3-16	-

Notes: (1) The 8251's RTS/ output is connected to the CTS/ input through jumper pair 27-28. The command instruction word for the 8251 must enable RTS/.

(2) TXC and RXC are connected to the Baud Rate Clk line via jumpers 33-34 and 35-36. The Baud Rate Clk should be configured for 110 baud by connecting jumpers 8-4 and 56-57 (see Table 3-9), and the 8251 should be programmed for a baud rate factor of 64 (see Section 3.3.2).

TABLE 3-7. RS232C INTERFACE, "DATA SET" ROLE

8251 PIN MNEMONIC	PIN NO.	LINE FUNCTION	CONNECTOR PIN No.	JUMPER CONNECTIONS
RXD	3	TRANSMITTED DATA	J3-3	37-38
TXD	19	RECEIVED DATA	J3-5	2-3
(1) CTS/	17	REQ TO SEND	J3-7	27-28
RTS/	23	CLEAR TO SEND	J3-9	29-30
DTR/	24	DATA SET READY	J3-11	22-23
(2) DSR/	22	DATA TERMINAL RDY	J3-14	25-26
-	-	PROTECTIVE GROUND	J3-1	-
-	-	SIGNAL GROUND	J3-13	-

TABLE 3-8. RS232C INTERFACE, "DATA PROCESSING TERMINAL" ROLE[1]

8251 PIN MNEMONIC	PIN NO.	LINE FUNCTION	CONNECTOR PIN NO.	JUMPER CONNECTIONS
TXD	19	TRANSMITTED DATA	J3-5	2-3
RXD	3	RECEIVED DATA	J3-3	37-38
RTS/	23	REQ TO SEND	J3-9	29-30
(1) CTS/	17	CLEAR TO SEND	J3-7	27-28
DTR/	24	DATA TERMINAL RDY	J3-11	22-23
(3) TXC	9	TRANSMIT CLOCK	J3-14	32-33
(2) DSP/	22	DATA SET RDY	J3-14	25-26
(3) RXC	25	RECEIVE CLOCK	J3-22	36-39
-	-	PROTECTIVE GROUND	J3-1	-
-	-	SIGNAL GROUND	J3-13	-

Notes:
(1) The CTS/ input pin on the 8251 must be "low" to enable the 8251 to transmit.
(2) When connector pin J3-14 is jumpered (25-26) to the DSR/ input, J3-14 cannot be used to supply an external transmit clock.
(3) In the asynchronous mode, TXC and RXC can be connected to externally supplied clocks via jumpers 32-33 and 36-39, or they can be connected to the internal Baud Rate Clk via jumpers 33-34 and 35-36, regardless of the mode.

[1]In this role, cable modifications must be made to conform with RS232 standards.

As explained in Section 3.2.5.1, there are two types of control words: (1) Mode instruction and (2) Command instruction. The Mode instruction initializes the 8251 USART. Because the USART supports either synchronous or asynchronous operation, the Mode instruction has one format for synchronous operation and another for asynchronous. The two least significant bits of the Mode instruction byte specify the format. If D0 and D1 both equal 0, synchronous operation is indicated; otherwise, it is asynchronous. The Mode instruction format for asynchronous operation is illustrated in Figure 3-6. The Mode instruction for synchronous operation is shown in Figure 3-26.

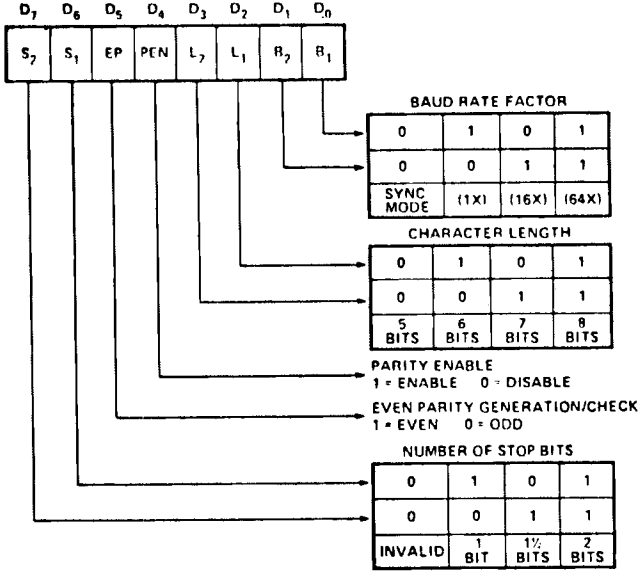
Notice in Figure 3-25 that the baud rate factor is specified by the two least significant bits of the instruction byte (labeled B1 and B2). During asynchronous communications, the Baud Rate Clock frequency supplied to the 8251's TXC and RXC input pins is divided by the baud rate factor to produce the effective baud rate (i.e., the frequency at which data bits are serially transmitted by the 8251 USART). Consequently, the Baud Rate Clock, as well as the program-selected baud rate factor, must be considered in implementing the desired effective baud rate. The Baud Rate Clock frequency is selected through various jumper connections as shown on sheet 4 of the SBC 80/10 schematic (Appendix B). The selection of an effective baud rate is summarized in Table 3-9.

Notice from the schematic that TXC and RXC inputs can be supplied by externally supplied clocks (via connector pins J3-14 and J3-22, respectively), instead of using the Baud Rate Clock, if jumpers 32-33 and 36-39 are connected and jumpers 33-34 and 35-36 are disconnected.

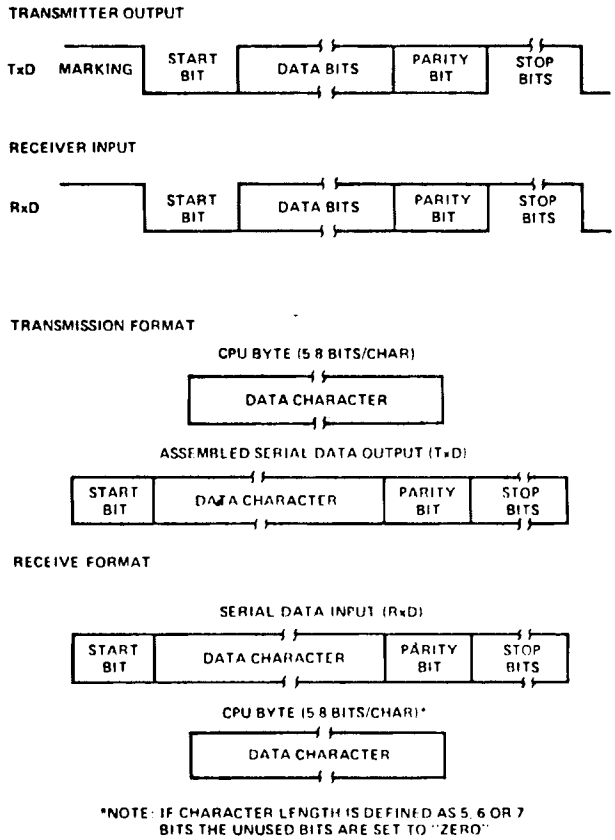
3.3.2 PARALLEL I/O OPTIONS

The Parallel I/O Interface consists of six 8-bit I/O ports implemented with two Intel 8255 Programmable Peripheral Interface devices. The primary user considerations in determining how to use each of the six I/O ports are:

1) Choice of operating mode (as defined in Section 3.2.6.1),
2) direction of data flow (input, output or bidirectional), and
3) choice of driver/termination networks for port's data path.

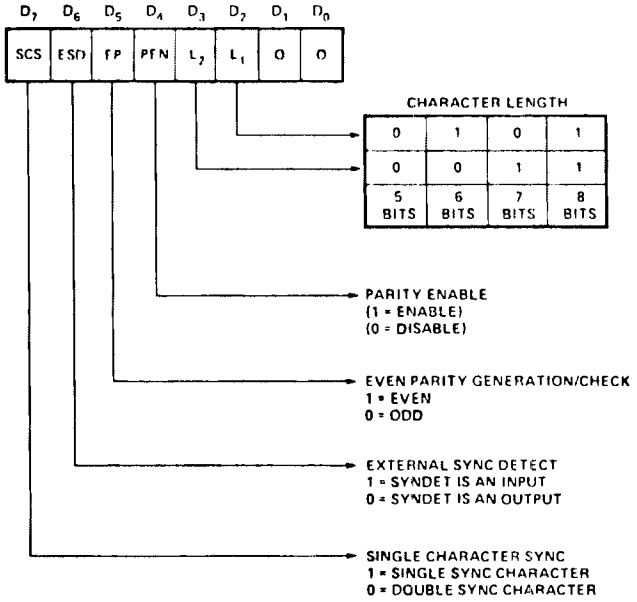


Mode Instruction Format, Asynchronous Mode

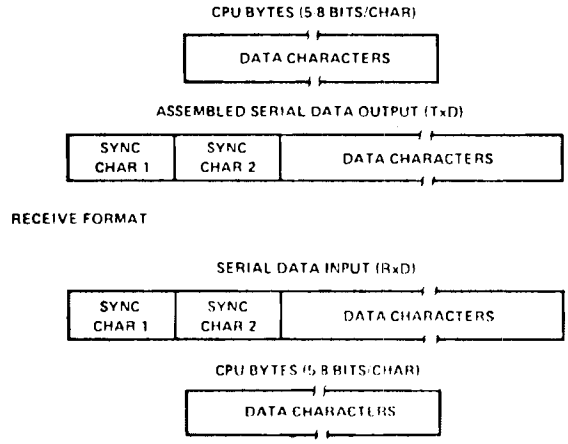


Asynchronous Mode

FIGURE 3-25. ASYNCHRONOUS OPERATION



Mode Instruction Format, Synchronous Mode



Synchronous Mode, Transmission Format

FIGURE 3-26. SYNCHRONOUS OPERATION

TABLE 3-9. BAUD RATE SELECTION

JUMPER CONNECTION	EFFECTIVE BAUD RATE (Hz)		
	SYNCHRONOUS MODE	ASYNCHRONOUS MODE	
		BAUD RATE FACTOR=16(2)	BAUD RATE FACTOR=64(2)
10-4	-	9600(3)	4800
11-4	-		2400
12-4	-	4800	1200
5-4	38,400	2400	600
6-4	19,200	1200	300
7-4	9600	600	150
(1) 8-4	4800	300	75
(1) 8-4, 56-57	6980	-	110 (TTY)

Note: (1) If jumper pair 56-57 is not connected, the frequency at jumper pole 8 is 4.8 KHZ. If jumper 56-57 is connected, however, the frequency at jumper pole 8 is 6.98 KHZ which, with a programmed baud rate factor of 64, provides an effective baud rate of approximately 110 baud for Teletype use.

(2) Baud rate factor is software selectable.

(3) Caution: Baud Rate Factor = 16

In the following paragraphs, we will define the capabilities of each port and summarize, in tables, that information which is necessary to use the port in each of its potential configurations. Each table will list the port I/O address, the control register address and the format for the control word which is output to the 8255 by the CPU Set and which specifies the particular configuration to be used. Each table will also summarize all of the relevant information concerning the choice and use of driver/termination networks, the data polarity, the connecting of jumpers and what they enable, and any restrictions on the use of the other two ports in each group. Examples of suitable driver/termination networks are listed in Section 8.1.1.

3.3.2.1 PORT 1 (GROUP 1 PORT A)

Port 1 is the only port that already includes a permanent bidirectional driver/termination network (two 8226 Bidirectional Bus Drivers). Port 1 is also the only port which can be programmed to function in any one of the

three 8255 operating modes, which were defined in Section 3.2.6.1. Before Port 1 is programmed for input or output in any one of the three modes, certain jumper connections must be made to allow the port to function properly in the chosen mode. Other jumper connections must be made to enable interrupts when Port 1 is in mode 1 or mode 2. In all, there are five potential configurations for Port 1. All of the necessary information for implementing each configuration has been summarized in the following tables:

TABLE 3-10. PORT 1 OPERATING MODES

PORT 1 CONFIGURATIONS		TABLE
Mode	Direction	
1. Mode 0	Input	Table 3-11
2. Mode 0	Output (Latched)	Table 3-12
3. Mode 1	Input (Strobed)	Table 3-13
4. Mode 1	Output (Latched)	Table 3-14
5. Mode 2	Bidirectional	Table 3-15

TABLE 3-11. PORT 1, MODE 0 INPUT CONFIGURATION

PORT 1 ADDRESS: E4, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	1	x	x	x	x

DRIVER/TERMINATION NETWORKS: Two Intel® 8226 Bidirectional Bus Drivers permanently installed at A1 and A2.

DATA POLARITY: Negative-true.

JUMPER CONNECTIONS: 41-42 to enable input at 8226's.

PORT 2 RESTRICTIONS: None; port 2 can be programmed for mode 0 or mode 1, input or output (see Section 3.3.2.2).

PORT 3 RESTRICTIONS: None; port 3 can be programmed for mode 0, 8-bit input or output, unless port 2 is in mode 1. (see Section 3.3.2.3).

TABLE 3-12. PORT 1, MODE 0 LATCHED OUTPUT CONFIGURATION

PORT 1 ADDRESS: E4, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	0	x	x	x	x

DRIVER/TERMINATION NETWORKS: Two Intel® 8226 Bidirectional Bus Drivers permanently installed at A1 and A2.

DATA POLARITY: Negative-true.

JUMPER CONNECTIONS: 40-41 to enable output at 8226's.

PORT 2 RESTRICTIONS: None; port 2 can be programmed for mode 0 or mode 1, input or output (see Section 3.3.2.2).

PORT 3 RESTRICTIONS: None; port 3 can be programmed for mode 0, input or output, unless port 2 is in mode 1 (see Section 3.3.2.3).

TABLE 3-13. PORT 1, MODE 1 STROBED INPUT CONFIGURATION

PORT 1 ADDRESS: E4, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	1	1	x	x	x	x

DRIVER/TERMINATION NETWORKS: Two Intel® 8226 Bidirectional Bus Drivers permanently installed at A1 and A2. A driver network must be installed at A3 and a termination network must be installed at A4.

DATA POLARITY: Negative-true. The polarity of Port 3 control outputs is dependent on the type of driver installed at A3.

JUMPER CONNECTIONS: 41-42 to enable input at 8226's; connect 49-50 to enable interrupt request via INT55/.

PORT 2 RESTRICTIONS: None; port 2 can be programmed for mode 0 or mode 1, input or output (see Section 3.3.2.2).

PORT 3 RESTRICTIONS: Port 3 bits perform the following dedicated functions:

*Bits 0, 1 and 2 - dedicated to control of port 2 if port 2 is in mode 1 (see Tables 3-17 to 3-20).
*Bit 3 - INTR (interrupt request) output for port 1.
*Bit 4 - STB/ (strobe) input for port 1.
*Bit 5 - IBF (input buffer full) output for port 1.
*Bit 6 - can be used for input only. Bit 3 of control word = 1
*Bit 7 - cannot be used.

TABLE 3-14. PORT 1, MODE 1 LATCHED OUTPUT CONFIGURATION

PORT 1 ADDRESS: E4, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	1	0	x	x	x	x

DRIVER/TERMINATION NETWORKS: Two Intel® 8226 Bidirectional Bus Drivers permanently installed at A1 and A2. A driver network must be installed at A3 and a termination network must be installed at A4.

DATA POLARITY: Negative-true. The polarity of Port C control outputs is dependent on the type of driver installed at A3.

JUMPER CONNECTIONS: 41-40 to enable output at 8226's; connect 49-50 to enable interrupt request via INT55/.

PORT 2 RESTRICTIONS: None; port 2 can be programmed for mode 0 or mode 1, input or output (see Section 3.3.2.2).

PORT 3 RESTRICTIONS: Port 3 bits perform the following dedicated functions:

*Bits 0, 1 and 2 - dedicated to the control of port 2 if port 2 is in mode 1 (see Tables 3-19 and 3-20).
*Bit 3 - INTR (interrupt request) output for port 1.
*Bit 4 - can be used for input if bit 3 of control word = 1
*Bit 5 - cannot be used if PC4 is used; can be used for output if control word bit 3 = 0 (PC4 cannot be used then).
*Bit 6 - ACK/ (acknowledge) input for port 1.
*Bit 7 - OBF/ (output buffer full) output for port 1.

TABLE 3-15. PORT 1, MODE 2 BIDIRECTIONAL CONFIGURATION

PORT 1 ADDRESS: E4, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	1	x	x	x	x	x	x

DRIVER/TERMINATION NETWORKS: Two Intel®8226 Bidirectional Bus Drivers permanently installed at A1 and A2. A driver network must be installed at A3 and a termination network must be installed at A4.

DATA POLARITY: Negative-true. The polarity of Port C control outputs is dependent on the type of driver installed at A3.

JUMPER CONNECTIONS: 41-43 to allow ACK/ input on PC6 to dynamically change data direction at 8226's (input when ACK/ = 1 and output when ACK/ = 0); connect 49-50 to enable interrupt request via INT55/.

PORT 2 RESTRICTIONS: None.

PORT 3 RESTRICTIONS: Port 3 bits perform the following dedicated functions:

*Bits 0 and 1 - can be used for output if bit 3 of control word = 0
*Bit 2 - cannot be used if PC0 and PC1 are used; can be used for input if control word bit 3 = 1 (PC0 and PC1 cannot be used then).
*Bit 3 - INTR (interrupt request) output for port 1.
*Bit 4 - STB/ (strobe input for port 1.
*Bit 5 - IBF (input buffer full) output for port 1.
*Bit 6 - ACK/ (acknowledge) input for port 1.
*Bit 7 - OBF/ (output buffer full) output for port 1.

3.3.2.2 PORT 2 (GROUP 2 PORT B)

Port 2 can be programmed for input or output in either mode 0 or mode 1. If Port 1 is in mode 2, however, Port 2 must be programmed for mode 0. If Port 2 is to be used for input, in either mode, terminator networks must be installed in the sockets at A5 and A6. If Port 2 is to be used for output, in either mode, driver networks must be installed in the sockets at A5 and A6. When Port 2 is programmed for mode 1, interrupts can be enabled by connecting jumper pair 45-46. The four potential configurations for Port 2 are summarized in the following tables:

TABLE 3-16. PORT 2 OPERATING MODES

PORT 2 CONFIGURATIONS		TABLE
Mode	Direction	
1. Mode 0	Input	Table 3-17
2. Mode 0	Output (Latched)	Table 3-18
3. Mode 1	Input (Strobed)	Table 3-19
4. Mode 1	Output (Latched)	Table 3-20

TABLE 3-17. PORT 2, MODE 0 INPUT CONFIGURATION

PORT 2 ADDRESS: E5, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	x	x	x	x	0	1	x

DRIVER/TERMINATION NETWORKS: Termination networks must be installed at A5 and A6.

DATA POLARITY: Positive-true.

JUMPER CONNECTION: None.

PORT 1 RESTRICTIONS: None (see Section 3.3.2.1).

PORT 3 RESTRICTIONS: None, port 3 can be programmed for mode 0, input or output, unless port 1 is in mode 1 or mode 2 (see Section 3.3.2.3).

TABLE 3-18. PORT 2, MODE 0 LATCHED OUTPUT CONFIGURATION

PORT 2 ADDRESS: E5, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	x	x	x	x	0	0	x

DRIVER/TERMINATION NETWORKS: Driver networks must be installed at A5 and A6.

DATA POLARITY: Negative-true, assuming that inverting drivers are at A5 and A6.

JUMPER CONNECTIONS: None.

PORT 1 RESTRICTIONS: None (see Section 3.3.2.1).

PORT 3 RESTRICTIONS: None, port 3 can be programmed for mode 0 or mode 1, 8-bit input or output, unless port 1 is in mode 1 or mode 2 (see Section 3.3.2.3).

TABLE 3-19. PORT 2, MODE 1 STROBED INPUT CONFIGURATION

PORT 2 ADDRESS: E5, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	x	x	x	1	1	x

DRIVER/TERMINATION NETWORKS: Termination networks must be installed at A5 and A6. A driver network must be installed at A3 and a termination network must be installed at A4.

DATA POLARITY: Positive-true. The polarity of Port C control outputs is dependent on the type of driver installed at A3.

JUMPER CONNECTIONS: 45-46 to enable interrupt request via INT55/.

PORT 1 RESTRICTIONS: None.

PORT 3 RESTRICTIONS: Port 3 bits perform the following dedicated functions:

*Bit 0 - INTR (interrupt request) output for port 2.
*Bit 1 - IBF (input buffer full) output for port 2.
*Bit 2 - STB/ (strobe) input for port 2.
*Bit 3 to Bit 7 - dedicated to control of port 1 if port 1 is in mode 1 (see Tables 3-11 to 3-14).

TABLE 3-20. PORT 2, MODE 1 LATCHED OUTPUT CONFIGURATION

PORT 2 ADDRESS: E5, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	x	x	x	1	0	x

DRIVER/TERMINATION NETWORKS: Driver networks must be installed at A5 and A6. A driver network must be installed at A3 and a termination network must be installed at A4.

DATA POLARITY: Negative-true, assuming that inverting drivers are at A5 and A6. The polarity of Port C control outputs is dependent on the type of driver installed at A3.

JUMPER CONNECTIONS: 45-46 to enable interrupt request via INT55/

PORT 1 RESTRICTIONS: None.

PORT 3 RESTRICTIONS: Port 3 bits perform the following dedicated functions:

*Bit 0 - INTR (interrupt request) output for port 2.
*Bit 1 - OBF/ (output buffer full) output for port 2.
*Bit 2 - ACK/ (acknowledge) input for port 2.
*Bit 3 - P3-7 - dedicated to control of port 1 if port 1 is in mode 1 (see Tables 3-11 to 3-14).

3.3.2.3 PORT 3 (GROUP 1 PORT C)

The use of Port 3 is dependent on the modes programmed for Ports 1 and 2 (refer to Tables 3-11 to 3-20). While certain Port 3 bits are available if Port 1 is in mode 1 or if Port 2 is in mode 0, the use of Port 3 as an 8-bit data path is restricted to those configurations that have both Port 1 and Port 2 programmed for mode 0. In this case, all eight bits of Port 3 can be programmed for mode 0 input (see Table 3-22) or output (see Table 3-23). A 4-bit input/4-bit output configuration is never possible for group 1 Port 3.

Note: If Ports 1 and 2 are not both in mode 0, then a driver network must be installed in the sockets at A3 and a termination network must be installed at A4, so that the Port 3 control lines can function properly.

3.3.2.4 PORT 4 AND 5 (GROUP 2 PORTS A AND B)

Ports 4 and 5 can be programmed for input or output but only in mode 0. The two potential configurations for each port are summarized in the following tables:

TABLE 3-21. PORT 4 AND 5 OPERATING MODES

CONFIGURATIONS			TABLE
PORT	MODE	DIRECTION	
1. Port 4	Mode 0	Input	Table 3-24
2. Port 4	Mode 0	Output (Latched)	Table 3-25
1. Port 5	Mode 0	Input	Table 3-26
2. Port 5	Mode 0	Output (Latched)	Table 3-27

TABLE 3-22. PORT 3, MODE 0, 8-BIT INPUT CONFIGURATION

PORT 3 ADDRESS: E6, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	x	1	0	x	1

DRIVER/TERMINATION NETWORKS: Termination networks must be installed at A3 and A4.

DATA POLARITY: Positive-true.

JUMPER CONNECTIONS: 46-47 and 44-45 to disable port 2 interrupts and enable P3-0; connect 48-49 and 50-51 to disable port 1 interrupts and enable P3-3.

PORT 1 AND 2 RESTRICTIONS: Both ports 1 and 2 must be in mode 0.

TABLE 3-23. PORT 3, MODE 0, 8-BIT LATCHED OUTPUT CONFIGURATION

PORT 3 ADDRESS: E6, CONTROL REGISTER ADDRESS: E7

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	x	0	0	x	0

DRIVER/TERMINATION NETWORKS: Driver networks must be installed at A3 and A4.

DATA POLARITY: Negative-true, assuming that inverting drivers are installed at A3 and A4.

JUMPER CONNECTIONS: 46-47, and 44-45 to disable port 2 interrupts and enable P3-0; connect 48-49 and 50-51 to disable port 1 interrupts and enable P3-3.

PORT 1 AND 2 RESTRICTIONS: Both ports 1 and 2 must be in mode 0.

TABLE 3-24. PORT 4, MODE 0, INPUT CONFIGURATION

PORT 4 ADDRESS: E8, CONTROL REGISTER ADDRESS: EB

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	1	x	0	x	x

DRIVER/TERMINATION NETWORKS: Termination networks must be installed at A7 and A8.

DATA POLARITY: Positive-true.

JUMPER CONNECTIONS: None.

PORT 5 AND 6 RESTRICTIONS: None; ports 5 and 6 can be programmed for mode 0, input or output (also see Section 3.3.2.5).

TABLE 3-25. PORT 4, MODE 0 LATCHED OUTPUT CONFIGURATION

PORT 4 ADDRESS: E8, CONTROL REGISTER ADDRESS: EB

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	0	x	0	x	x

DRIVER/TERMINATION NETWORKS: Driver networks must be installed at A7 and A8.

DATA POLARITY: Negative-true, assuming that inverting drivers are installed at A7 and A8.

JUMPER CONNECTIONS: None.

PORT 5 AND 6 RESTRICTIONS: None; ports 5 and 6 can be programmed for mode 0, input or output (also see Section 3.3.2.5).

TABLE 3-26. PORT 5, MODE 0 INPUT CONFIGURATION

PORT 5 ADDRESS: E9, CONTROL REGISTER ADDRESS: EB

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	x	x	0	x	x

DRIVER/TERMINATION NETWORKS: Termination networks must be installed at A11 and A21.

DATA POLARITY: Positive-true.

JUMPER CONNECTIONS: None.

PORT 4 AND 6 RESTRICTIONS: None; ports 4 and 6 can be programmed for mode 0, input or output (also see Section 3.3.2.5).

TABLE 3-27 PORT 5, MODE 0 LATCHED OUTPUT CONFIGURATION

PORT 5 ADDRESS: E9, CONTROL REGISTER ADDRESS: EB

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	x	0	0	x	x

DRIVER/TERMINATION NETWORKS: Driver networks must be installed at A11 and A21.

DATA POLARITY: Negative-true, assuming that inverting drivers are installed at A11 and A21.

JUMPER CONNECTIONS: None.

PORT 4 AND 6 RESTRICTIONS: None; ports 4 and 6 can be programmed for mode 0, input or output (also Section 3.3.2.5).

3.3.2.5 PORT 6 (GROUP 2 PORT C)

All eight bits of Port 6 can be programmed for mode 0 input or output, or four bits can be programmed for mode 0 input while the other four bits are programmed for mode 0 output. The four potential configurations for Port 6 are summarized in the following tables:

TABLE 3-28. PORT 6 OPERATING MODES

PORT 6 CONFIGURATIONS	TABLE
1. MODE 0 8-BIT INPUT	Table 3-29
2. MODE 0 8-BIT OUTPUT (LATCHED)	Table 3-30
3. MODE 0 UPPER 4-BIT INPUT/LOWER 4-BIT OUTPUT	Table 3-31
4. MODE 0 UPPER 4-BIT OUTPUT/LOWER 4-BIT INPUT	Table 3-32

TABLE 3-29. PORT 6, MODE 0, 8-BIT INPUT CONFIGURATION

PORT 6 ADDRESS: EA, CONTROL REGISTER ADDRESS: EB

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	x	1	0	x	1

DRIVER/TERMINATION NETWORKS: Termination networks must be installed at A9 and A10.

DATA POLARITY: Positive-true.

JUMPER CONNECTIONS: None.

PORT 4 AND 5 RESTRICTIONS: None (see Section 3.3.2.4).

TABLE 3-30. PORT 6, MODE 0, 8-BIT LATCHED OUTPUT CONFIGURATION

PORT 6 ADDRESS: EA, CONTROL REGISTER ADDRESS: EB

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	x	0	0	x	0

DRIVER/TERMINATION NETWORKS: Driver networks must be installed at A9 and A10.

DATA POLARITY: Negative-true, assuming that inverting drivers are installed at A9 and A10.

JUMPER CONNECTIONS: None.

PORT 4 AND 5 RESTRICTIONS: None (see Section 3.3.2.4).

TABLE 3-31. PORT 6, MODE 0 UPPER 4-BIT INPUT/LOWER 4-BIT LATCHED OUTPUT CONFIGURATION

PORT 6 ADDRESS: EA, CONTROL REGISTER ADDRESS: EB

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	x	1	0	x	0

DRIVER/TERMINATION NETWORKS: A termination network must be installed at A9 and a driver network must be installed at A10.

DATA POLARITY: The upper 4-bits will be in positive-true form; however, the lower four bits will be in negative-true form if an inverting driver is installed at A10.

JUMPER CONNECTIONS: None.

PORT 4 AND 5 RESTRICTIONS: None (see Section 3.3.2.4).

TABLE 3-32. PORT 6, MODE 0 UPPER 4-BIT LATCHED OUTPUT/LOWER 4-BIT INPUT CONFIGURATION

PORT 6 ADDRESS: EA, CONTROL REGISTER ADDRESS: EB

CONTROL WORD FORMAT:

7	6	5	4	3	2	1	0
1	0	0	x	0	0	x	1

DRIVER/TERMINATION NETWORKS: A driver network must be installed at A9 and a termination network must be installed at A10.

DATA POLARITY: The lower 4-bits will be in positive-true form; however, the upper 4-bits will be in negative-true form if an inverting driver is installed at A9.

JUMPER CONNECTIONS: None.

PORT 4 AND 5 RESTRICTIONS: None (see Section 3.3.2.4).

TABLE 3-33. PARALLEL I/O ADDRESS AND SOCKET ASSIGNMENTS

PORT	I/O ADDRESS	SOCKET NUMBERS
1	E4	BI-DIRECTIONAL DRIVER/ TERMINATOR AT A1, A2
2	E5	A5, A6
3	E6	A3, A4*
4	E8	A7, A8
5	E9	A11, A21
6	EA	A9, A10**

*Note requirements specified in Tables 3-11 through 3-23.

**Note requirements specified in Tables 3-24 through 3-32.

3.3.3 GENERAL OPTIONS

There are several other options that may be useful. Details are provided in the following paragraphs.

3.3.3.1 SYSTEM RESET OUTPUT

The user can enable a SYSTEM RESET output from the SBC 80/10 by connecting jumper pair 54-55. This allows the reset signal which is generated on the SBC 80/10 during power-up sequences (see Section 3.2.1.5) to be made available to other modules in the system via connector P1-14. Notice on the schematic that a SYSTEM RESET input is accepted by the SBC 80/10 and P1-14 and applied to the 8080 regardless of jumper connections.

3.3.3.2 DISABLE BUS CLOCK SIGNALS

The bus clock BCLK/ (connector pin P1-13) or the constant clock CCLK/ (P-31) outputs can be disabled (if more drive, or a different frequency is needed) by disconnecting jumper pair 61-63 or 62-64, respectively. When connected, both BCLK/ and CCLK/ provide a 9.216 MHz timing reference to other modules.

3.3.3.3 ADVANCED ACKNOWLEDGE INPUT

Certain OEM mocules generate an advanced acknowledge, AACK/, in response to a memory read command, that allows the memory to complete the access without requiring the CPU to wait. When such modules are used with the SBC 80/10, jumper pair 52-53 should be connected to allow AACK/ to be accepted (at P1-25) and gated to the RDYIN pin on the 8224 Clock generator.

3.3.4. DEFAULT OPTIONS

Table 3-33 lists the default options jumpered on the SBC 80/10. These options permit the SBC 80/10 to communicate to a TTY; they also provide power-up reset, bus clock, and the communication clock to the system bus.

TABLE 3-34. DEFAULT OPTION ON THE SBC 80/10

DEFAULT JUMPERS	REFERENCE	DESCRIPTION
1 - 2	3.3.1.1	Connect 8251 T_xD to 20 mA Current Loop Driver
23 - 24		Connect 8251 DTR/ to TTY Reader Control Circuit
39 - 38		Connect 8251 R_xD to 20 mA current Loop Receiver
4 - 8	3.3.1.2	Generates 6.98K Baud Rate Clock
57 - 56		Generates 6.98K Baud Rate Clock
34 - 33		Connect 8251 T_x Clock to Baud Rate Clock
35 - 36		Connect 8251 R_x Clock to Baud Rate Clock
27 - 29		Connect 8251 RST/ to 8251 CTS/
19 - 20	3.2.5.4	Disable T_xRDY Interrupt from 8251
16 - 15	3.2.5.4	Disable R_xRDY Interrupt from 8251
26 - 25	3.3.1.2	Connect DTR/ Receiver to 8251 DSR/ Input
30 - 31	3.3.1.2	Connect Set Clear to Send Driver to +12V
40 - 41	3.3.2.1	Enable Port 1 Bi-directional Drivers to Input
54 - 55	3.3.3.1	Connect Power-Up Reset to System Bus
62 - 64	3.3.3.2	Connect 9.216 MHz Clock to Communication Clock Line
61 - 63	3.3.3.2	Connect 9.216 MHz Clock to Bus Clock Line
*65 - 66	3.3.5	
*68 - 69	3.3.5	Configures SBC 80/10A for 4K ROM/PROM
*73 - 74	3.3.5	
*76 - 78	3.3.5	

*Used with SBC 80/10A only.

3.3.5 JUMPER CONFIGURATION FOR ROM/PROM INSTALLATION

The System 80/10 using SBC 80/10A has jumpers which allow installation of up to 4K or up to 8K bytes of read only memory. Up to 4K bytes can be installed using Intel's 8708 Erasable and Electrically Reprogrammable ROMs (EPROM), Intel's 8308 Metal Masked ROMs, or Intel's 2758 Erasable and electrically Reprogrammable ROMs (EPROM). Up to 8K bytes can be installed using Intel's 2716 Erasable and Electrically Reprogrammable ROMs (EPROM) or Intel's 2316E Metal Masked ROMs. Table 3-35 list the jumper configurations for 4K and 8K

bytes of read only memory. Table 3-36 lists the addresses for each PROM socket in 4K and 8K configurations.

TABLE 3-35. PROM JUMPER CONFIGURATION

	JUMPER			
4K	65-66	68-69	73-74	76-78
*4K	66-67	69-71	73-74	76-78
8K	66-67	69-70	74-75	77-78

*Using Intel's 2758 Erasable and Electrically Reprogrammable ROMs (EPROM).

TABLE 3-36. PROM ADDRESSES

	CHIP ADDRESS			
	A23	A24	A25	A26
4K	0-3FF	400-7FF	800-BFF	C00-FFF
*4K	0-3FF	400-7FF	800-BFF	C00-FFF
8K	0-7FF	1000-17FF	800-FFF	1800-1FFF

*Using Intel's 2758 Erasable and Electrically Reprogrammable ROM's (EPROM)

CHAPTER 4
FRONT PANEL

The System 80/10 Front Panel allows the user to reset the entire system and provides a visual indication of AC power on.

The System 80/10 Front Panel consists of two switches:

(1) AC Power switch

(2) System Reset switch

4.1 AC POWER SWITCH

The AC power switch is a double-pole-double-throw latching switch. The switch is illuminated by a lamp. The lamp is a midget flanged base lamp, size T1 3/4, and operates at 28 volts. It can be replaced by simply pulling the plastic push button away from the switch housing. Front panel removable is not necessary.

4.2 SYSTEM RESET SWITCH

The System reset switch is a double-pole-double-throw momentary switch. One half of the DPDT switch is debounced by an RS flip-flop and an inverter driver; the other half of the DPDT switch is not debounced. The inverter driver is an open collector device (48 mA sink current) and it is connected directly to the System Bus (INIT/). See Figure 4-1.

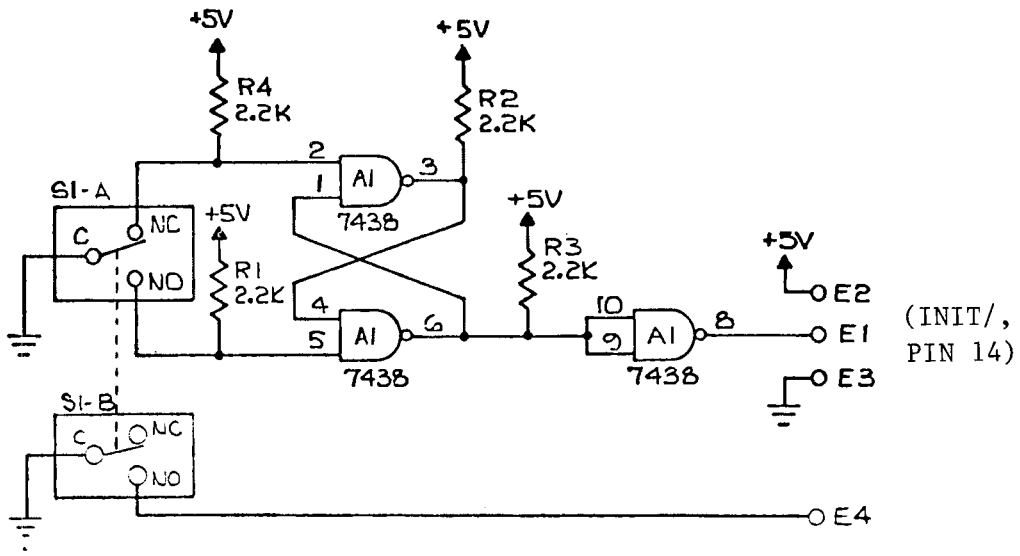


FIGURE 4-1. SYSTEM RESET SWITCH

CHAPTER 5

CARDCAGE AND BACKPLANE ASSEMBLY

5.1 FUNCTIONAL ORGANIZATION OF THE CARDCAGE AND BACKPLANE ASSEMBLY

The System 80/10 modular cardcage and backplane assembly consists of two functional blocks:

(1) Structural foam cardcage

(2) Termination backplane

Structural Foam Cardcage

The structural foam cardcage is an injection molded structure consisting of three separate pieces, the cardcage body and two card guides. The two card guides are bonded to the cardcage body by an ultrasonic bonding process. The cardcage assembly has a very high flexural, compressive and tensile strength. The resin used in the injection molded process allows for very close tolerance control of all dimensions.

Termination Backplane

The termination backplane consists of a motherboard, four 43/86 pin PCB connectors, termination resistors and two 7-pin power connectors. Most of the bus signals on the backplane assembly are terminated. The backplane accepts four SBC modules and each module has access to the bus. The two power connectors are used to supply power to the backplane.

5.2 CARDCAGE AND BACKPLANE ASSEMBLY UTILIZATION

The cardcage and backplane assembly houses the SBC 80/10 module and three additional expansion boards if needed. Paragraph 8.1.2 is a discussion of all signals on the backplane.

Four additional 30/60 pin auxiliary connectors can be added to the backplane assembly to form an auxiliary bus structure. The auxiliary bus structure can be used to implement battery back-up or an inter-module bus.

CHAPTER 6
POWER SUPPLY

6.1 FUNCTIONAL ORGANIZATION OF THE POWER SUPPLY

The System 80/10 Power Supply provides regulated DC output power at +12, +5, -5, and -12 volt levels. The current capabilities of each of these output levels have been chosen to provide power over the System 80/10 temperature range with the Single Board Computer fully loaded with I/O line terminators and drivers, and four 8708 EPROMs plus residual capability for most combinations of up to three SBC memory, I/O, or combination expansion boards within the System 80/10.

Current limiting and over-voltage protection is provided on all outputs. Access for AC input is provided via a standard 4-pin keyed connector. DC output power levels are provided on cables with keyed connectors which are directly compatible with the Modular Backplane/Cardcage assembly. The Power Supply includes logic which senses a system AC power failure and generates a TTL signal for clean system power-down control.

6.2 THEORY OF OPERATION

6.2.1 5V, 14 A OUTPUT

Operation is as follows:

(Ref. Schematic 2000952, Appendix B.)

6.2.1.1 VOLTAGE REGULATION

If we start at the moment of AC on, C1, 17 and 8 will begin charging up from the current through CR1, 6 and 7. U1 will be "off" until ≈8 volts is present at pins 11 and 12. At that point, the reference voltage of 7.15V will be present at pin 6. This reference voltage is divided down to 5V at pin 5, which is the non-inverting input of the internal differential amplifier. Pin 4 is the inverting input; since no output is present the differential amplifier is unbalanced and U1 will drive current out pin 10.

When pin 10 goes positive, Q1 turns on and drives the bases of Q2 and 3 on. Collector current through Q2 and 3 will charge the output cap C3 and 17 and deliver power to the load. When the voltage at C3 is 5.0 VDC, the differential amplifier will be balanced; pin 10 will reduce its output, and voltage regulation will occur. The output voltage may be adjusted by varying the resistance of R8.

6.2.1.2 CURRENT LIMIT/FOLDBACK

U1 has an internal current limit transistor that, when turned on, will reduce the output of pin 10.

The base-emitter of this transistor is brought out to pins 2 and 3. respectively. Pin 3 is connected to the output, pin 2 is connected to a current-sensitive point (in this case, the base of pass element Q2,3).

In normal, full load operation, R2 and 4 will drop ≃.5V because of the base current at Q2,3. R3 and 5 are shut-off resistors for the pass elements. (R7 is used to bias the current limit transistor off during normal conditions.)

Under full load operating conditions, the voltage at Q1-E will be approximately 6.5 VDC.

R6 is set to bring pin 2 to a threshold condition (≃5,5 VDC) when 120% to 150% of rated current is being delivered.

If additional current is drawn, the current limit transistor in U1 will reduce the output of pin 10. This causes a reduction in output voltage which reduces the bias current in R7 and increases the current to the current limit transistor. At the maximum load (shorted output), very little current flows through R7, which means that it takes only a slight voltage at Q1-E to activate the current limit transistor. The current limit/foldback circuit described above has a fairly low gain, and it is reasonably normal to deliver 20-50% of rated current into a short. This is desirable because most logic circuits have start-up conditions requiring greater current than linear V/I curves would indicate.

6.2.1.3 OVP OPERATION

Zener CR2 senses the output voltage. When this voltage is sufficient to cause CR7 to zener and charge C18, SCR1 will fire which reduces the output to ≃1 VDC. OVP can be reset by cycling the AC input.

6.2.2 +12V OPERATION (Ref. Schematic 2000952, Appendix B.)

6.2.2.1 REGULATION

When the secondary of T1 charges C8 through CR6 and 7, U3 will drive current out of pin 10 until the differential amplifier is balanced (see Section 6.2.1.1).

6.2.2.2 CURRENT LIMIT/FOLDBACK

U3 has an internal current limit transistor that, when turned on, will reduce the output of pin 10.

The base-emitter of this transistor is brought out to pins 2 and 3, respectively. Pin 3 is connected to the output, and pin 2 is connected to a current-sensitive point (in this case, the base of Q6).

In normal operation, at full load, R27 will have ≃.44 volts dropped across it. R25 and R26 are calculated (and tested) to bias pin 2-3 at ≃+.4V at full load. When excess current (≃120% of rated) is drawn, the increased drop across R3 will force the current limit transistor into conduction. When the load is further increased the current limit transistor takes more and more of the drive from U3-pin 10, and the output voltage will decrease. When the output voltage is decreased, the offset bias through R27 is less and further reduces the available current from U3-pin 10. At a short circuit, the current through R27 is minimal and output current will be 20-30% of rated.

6.2.2.3 OVP OPERATION (See section 6.2.1.2.)

6.2.3 -12V OPERATION

6.2.3.1 REGULATION

Regulation is accomplished in a similar manner to the +12V regulator (see Section 6.2.2.1). Q7 is used to maintain U4 at an operational bias level.

6.2.3.2 CURRENT LIMIT

The current limit circuit operates in an identical manner to the circuit of Section 6.2.2.2, except that the circuit does not have foldback characteristics. Current can be drawn until there is ≃.55 volts across R36. Since Pins 2 and 3 are across R36, this maximum current is also the short circuit current.

6.2.3.3 OVP (See Section 6.2.1.2)

6.2.4 -5V OPERATION

6.2.4.1 REGULATION

Regulation is accomplished in a similar manner to the +12V regulator (see Section 6.2.2.1). Q4 is used to maintain U2 at an operational bias level.

6.2.4.2 CURRENT LIMIT

The current limit circuit operates in an identical manner to the circuit of Section 6.2.2.2, except that the circuit does not have foldback characteristics. Current can be drawn until there is ≃.55 volts across R16 and R55. Since pins 2 and 3 are across R16 and R55, this maximum current is also the short circuit current.

6.2.4.3 OVP (See Section 6.2.1.2.)

6.2.5 POWER FAIL OPERATION

At turn-on C15 charges through CR13, 14, U5 is a 723 set up as a comparator. Pin 6 is the reference voltage. R44 adjusts the reference voltage

on pin 4 (the inverting input). When the voltage on pin 5 is higher than on pin 4 the output goes high and Q9 saturates. R48, 46, 51 determines the hysteresis and compensates for the ripple on C15. R49 limits the drive to Q9. Operational power is obtained from C8 (raw 12V).

6.3 DC POWER OUTPUTS

Power supply provides ±5V and ±12V through power supply connectors P6 and P8. The P8 harness is 16 inches in length and P6 harness is 24 inches in length from mid-point of surface A (see outline drawings in Appendix C). See the diagram below for pin out and voltage assignments.

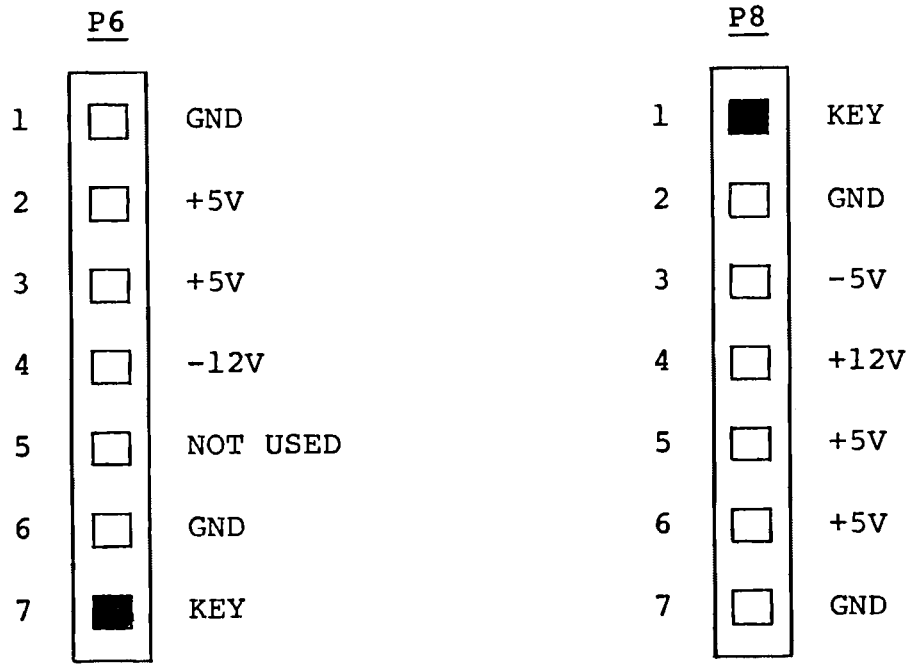


FIGURE 6-1

OUTPUT POWER CONNECTIONS

6.3.1 DC OUTPUT VOLTAGE ADJUSTMENT

Adjustment range for all voltages is ±5% of nominal voltage. To adjust output voltage, locate appropriate voltage adjustment trimmer (see outline drawings in Appendix C), then turn trimmer, with non-metallic screwdriver, clockwise to increase output voltage or counterclockwise to decrease output voltage. If over-voltage-protection circuit is tripped during voltage adjustment see OVP reset procedure below.

6.3.2 OVER-VOLTAGE-PROTECTION CIRCUIT RESET PROCEDURE

After OVP is tripped, input power must be turned off for at least 3 seconds. Reduce output voltage of the tripped OVP by rotating the voltage adjustment trimmer about a half turn counterclockwise. Turn power on, normal operation should be restored, if not, repeat OVP reset procedure. Re-adjust output voltage according to DC Output Voltage Adjustment above.

6.4 AC LOW DETECTION CIRCUIT CONNECTIONS*

An active high TTL level signal indicating AC power failure is provided to the user through connector J3 pin 5 (CAUTION: Other voltages are present in connector J3, see pin out diagram below.) Use Molex connector (P/N 09-50-7071), pin (P/N 08-50-0106) and polarizing key (P/N 15-04-0219) for mating parts to J3 connector.

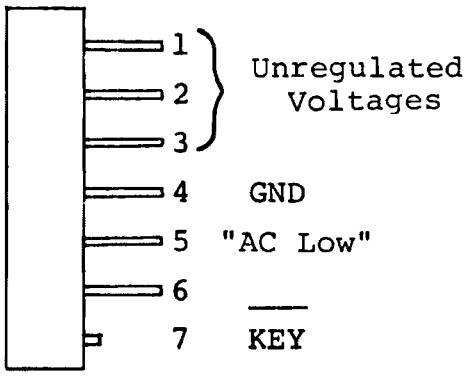


FIGURE 6-2

AC LOW DETECTION CIRCUIT CONNECTIONS

*For 115V/230V operation only.

(NOTE: The location of the AC low detection connector can vary within the indicated volume shown on the outline drawings. (See Appendix C.)

Application of AC Power Failure Detection Circuit

The "AC Low" output from the J3 connector is an active high TTL compatible signal which indicates that the AC input line voltage is below

103/206 VAC (RMS). "AC Low" output should be pulled up by the battery backup logic. In case of a power failure, all DC voltage is guaranteed to stay within regulation for a minimum of 7.5 msec at any frequency within the operating frequency range (47-63 hz) after the "AC Low" line goes high.

In a battery backup system, the "AC Low" signal is used to generate a "Power Fail Interrupt/" to the CPU and enable "Memory Protect" logic to disable any further Read/Write operations to the memory. A "Power Failure Detect/" signal may also be generated by the system to indicate that a power failure has occurred. Typical system timing during a power down and recovery is described in Figure 6-3.

When AC power recovers, the "AC Low" signal will return to the low state, the memory protect logic is then disarmed and a system "RESET" must be generated. Once reset, the system executes a start up routine which will sample the "Power Failure Detect/" line. If it is active, it will restore the CPU to its original line. If it is active, it will restore the CPU to its original state before power failed, and continue operation. If it is inactive, a "Cold Start" initialization routine will be performed.

6.5 MODIFICATION PROCEDURE FOR 230V OPERATION

The System 80/10 is wired for 115v/230v operation. The System is set to operate at 115v as shipped. For 230v operation follow the procedure as listed below:

(1) Turn off System and disconnect power cord from unit.

-- CAUTION --

Be sure power to System is OFF and power cord is removed from unit and power outlet. Damage to the System and personnel might result if power is NOT TURNED OFF!

(2) Remove 3 Amp fuse from fuse holder. Replace with 1½ Amp fuse that is shipped with the System.

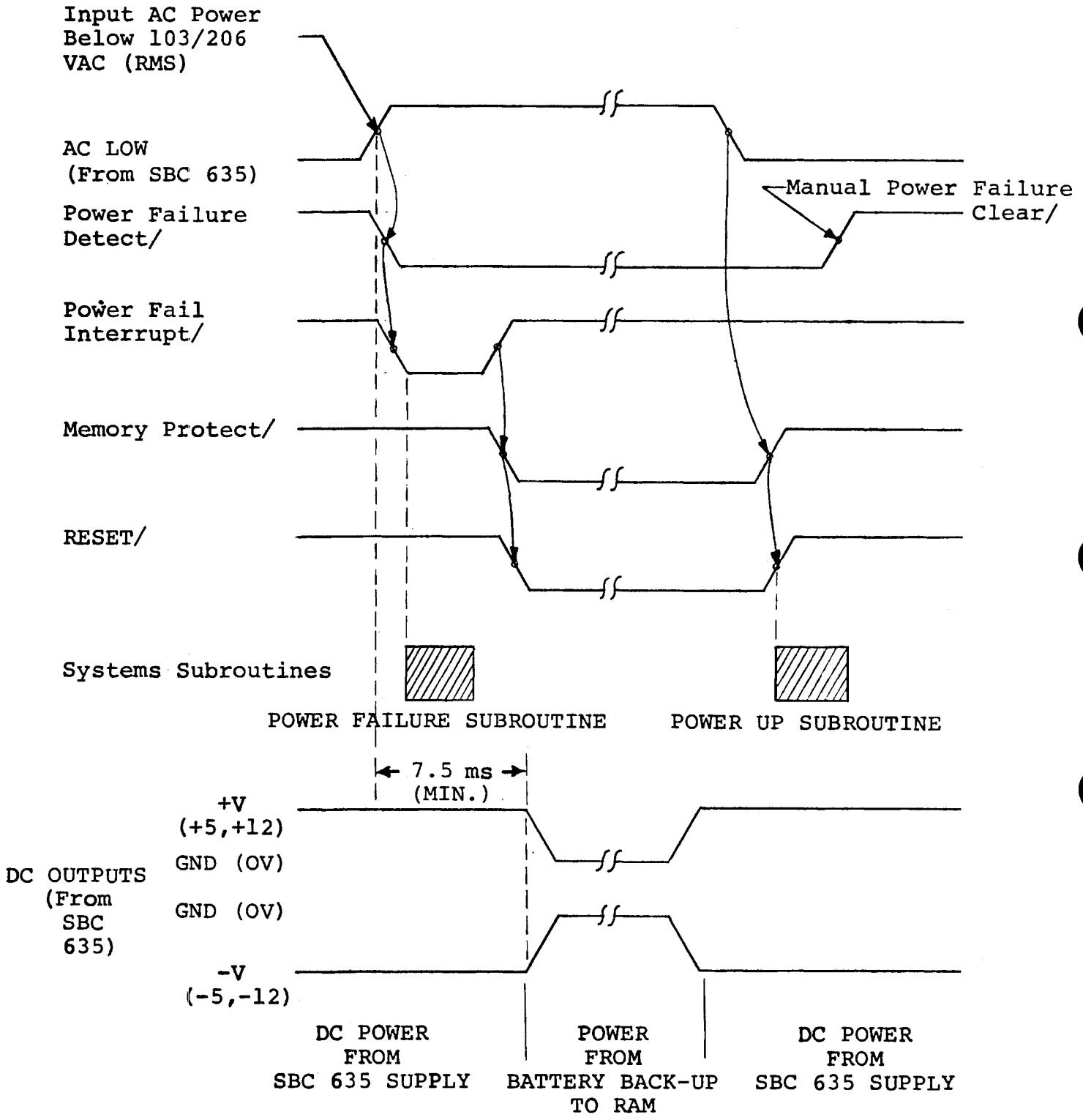


FIGURE 6-3

TYPICAL POWER FAIL SEQUENCE

(3) Remove top cover of System 80/10 and locate 115v/230v switch mounting bracket. Remove SBC 80/10 module.

(4) Loosen top two screws and remove switch locking plate from mounting bracket. Reverse orientation of locking plate by flipping locking plate over sideways. Locking plate is stamped with "230".

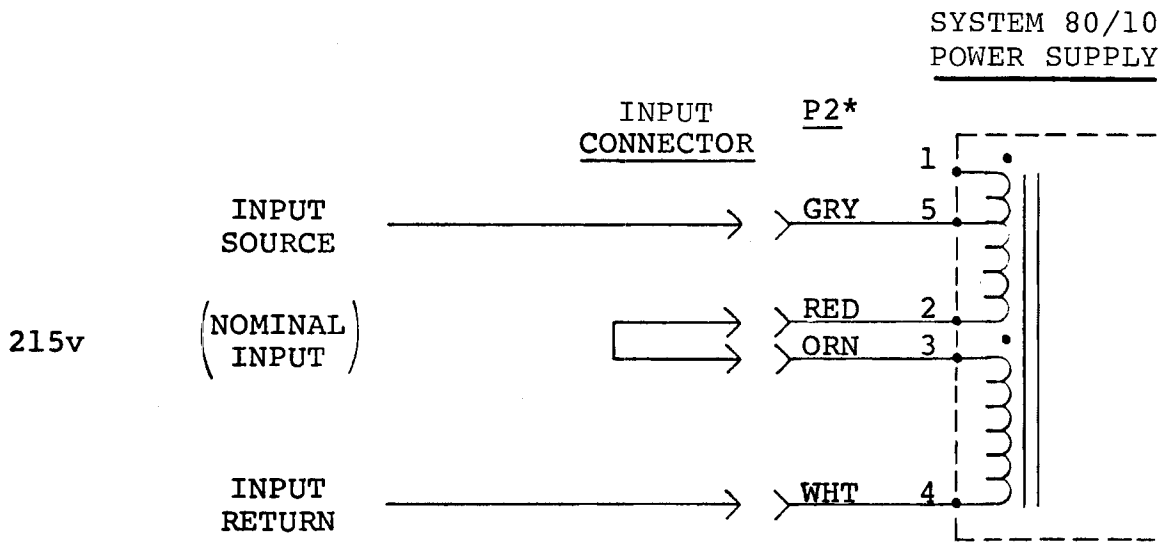
(5) Slide voltage selection switch to the left, the side marked with "230" on it, and re-install switch locking plate. Tighten top two screws.

(6) Re-connect power cord to unit. Turn unit on and verify all voltages on backplane assembly. Turn off power and re-install SBC 80/10 module. Unit is now ready for 230v operation.

Modifications must be made to the transformer for other AC input voltages. See Table 6-1 and Figures 6-4 and 6-5.

TABLE 6-1

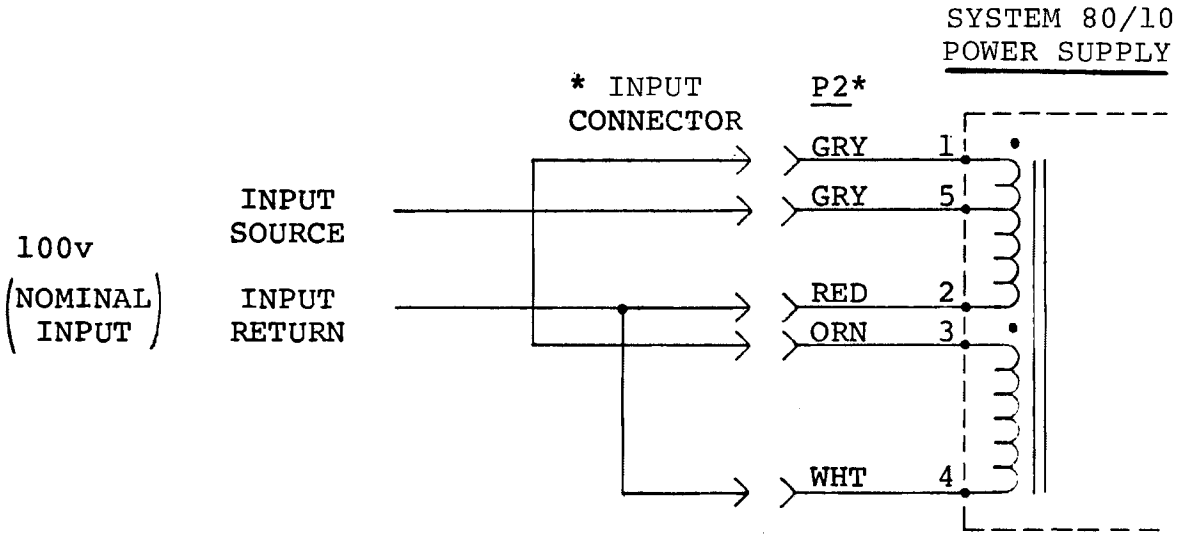
OPTIONAL TRANSFORMER CONNECTIONS				
INPUT RANGE	NOMINAL INPUT	INPUT SOURCE (PIN)	INPUT RETURN (PIN)	TRANSFORMER JUMPERS REQUIRED (PIN)
103.5-126.5	115	1	2	1-3, 2-4
207-253	230	1	4	2-3
192.5-236.5	215	5	4	2-3
90-110	100	5	2	1-3, 2-4



*Wire from transformer T1, Pin 1 is moved to T1, Pin 5.

FIGURE 6-4

215v CONNECTION



*5 pin connectors must be used. Use Molex P/N 03-09-2052 for P2 transformer connector and use Molex P/N 03-09-1052 as mating connector.

FIGURE 6-5

100v CONNECTION

CHAPTER 7
SYSTEM MONITOR

7.1 MONITOR FUNCTIONAL SPECIFICATION

7.1.1 GENERAL CHARACTERISTICS AND SCOPE

The monitor is a program written in Intel®8080 macro assembly language. The monitor resides in two programmed ROMs and is located in the memory address space of the System 80/10 microcomputer between 0 and 0560H (H=hexadecimal). The non-volatile nature of the program's storage media means that the monitor is available for use immediately after power-on or reset.

7.1.2 DESCRIPTION OF ALL MAJOR FUNCTIONS PERFORMED

7.1.2.1 CONSOLE COMMANDS

The monitor communicates with the operator via an interactive console, normally a teletypewriter. The dialogue between the operator and the monitor consists of commands in the monitor's command language and the monitor's responses. After the cold start procedure, the monitor begins the dialogue by typing a sign-on message on the console and then requests a command by presenting a prompt character, ".". Commands are in the form of a single alphabetic character specifying the command, followed by a list of numeric or alphabetic parameters. Numeric parameters are entered as hexadecimal numbers. The monitor recognizes the characters 0 through 9 and A through F as legal hexadecimal digits. The valid range of numbers is from 1 to 4 hex digits. Longer numbers may be entered, but such numbers will be evaluated modulo 2^{16} so that they will fall into the range specified above.

The only command requiring an alphabetic parameter is the "X" command. The nature of such parameters will be discussed in the section explaining the command.

7.1.2.2 USE OF THE MONITOR FOR PROGRAMMING AND CHECKOUT

The monitor allows the user to enter, check out, and execute programs. The monitor contains facilities for memory modification, 8080 CPU register display and modification, program input and output from the console device, program initiation, and the recognition of an "RST 7" instruction as an unconditional branch to RAM address 3C3DH. By inserting RST 7 instructions in a program under test, or by using the hardware generated RST 7 instruction (if available), the user can cause execution of a program to transfer to a dedicated location (3C3DH), for whatever purposes he desires.

When the user wishes to re-enter the monitor (also see Appendix E), he should use an RST 1 instruction coded into his program. When entered in this manner, the monitor will print an "#" followed by the contents of the user program counter (byte address of RST 1 instruction plus one). The monitor will also automatically save the state of the 8080: specifically, it will save all registers (A, B, C, D, E, H, L), the CPU flags (F), the user's Program Counter (PC), and the user's Stack Pointer (SP). These may be examined with the X command. When the operator enters a G command, these values will be restored.

7.1.2.3 I/O SYSTEM

The I/O system provides four routines. Console character in and console character out, which the user may call upon to read and write, respectively, characters from and to the console device. The other two routines allow the user to read and punch paper tapes from the teletype.

7.1.3 APPLICABLE STANDARDS

Throughout this specification, the numbering convention for bits in a word is that bit 0 is the least significant, or rightmost bit.

The internal code set used by the monitor is 7 bit (no parity) ASCII.

7.2 MONITOR INTERFACE SPECIFICATIONS

7.2.1 COMMAND STRUCTURE

In the following paragraphs the monitor command language is discussed. Each command is described, and examples of its use are included for clarity. Error conditions that may be encountered while operating the monitor are described in Section 7.3.2.

The monitor requires each command to be terminated by a carriage return. With the exception of the "S" and "X" commands, the command is not acted upon until the carriage return is sensed. Therefore, the user can abort any command, before he enters the carriage return by typing any illegal character (such as RUBOUT or any alphabetic character with the exception of A through F).

Except where indicated otherwise, a single space is synonymous with the comma for use as a delimiter. Consecutive spaces or commas, or a space or comma immediately following the command letter, will be interpreted as null parameters. Null parameters are illegal in all commands except the "X" command (see below).

Items enclosed in square brackets "[" and "]" are optional. The consequences of including or omitting them are discussed in the text.

7.2.1.1 DISPLAY MEMORY COMMAND, D

D<low address>,<high address>

Selected areas of addressable memory may be accessed and displayed by the D command. The D command produces a formatted listing of the memory area between <low address> and <high address>, inclusive, on the console device. Each line of the listing begins with the address of the first memory location displayed on that line, represented as 4 hexadecimal digits, followed by up to 16 memory locations, each one represented by 2 hexadecimal digits.

The D command may be aborted during execution by typing an Escape (ESC) on the console. The command will be terminated immediately, and a new prompt issued.

Example

D9,2A

0009 00 11 22 33 44 55 66

0010 77 88 99 AA BB CC DD EE FF 10 20 30 40 50 60 70

0020 80 90 A0 B0 C0 D0 E0 F0 01 02 03

Note: If the <low addresss> parameter is greater than the <high address> parameter, only the first location defined by <low address> is printed.

7.2.1.2 PROGRAM EXECUTE COMMAND, G

G[<entry point>]

Control of the CPU is transferred from the monitor to the user program by means of the program execute command, G. The <entry point> should be an address which contains an instruction in the user's program. If no entry point is specified, the monitor uses, as an address, the value of the P register saved during a previous "G" command or saved as a result of a RST 1 instruction coded into the user's program.

Example

G1400

Control is passed to location 1400H.

Note: When entering a user program for the first time, the user should initialize the Stack Pointer to his stack area. Also, after a reset to the SBC 80/10 the user Stack Pointer must be reinitialized by the user during subsequent entry into the user's program or by the 'XS' command.

7.2.1.3 INSERT INSTRUCTIONS INTO MEMORY, I

I<address>

Single or multiple instructions are entered into memory with the I command. After sensing the carriage return terminating the command line, the monitor waits for the user to enter a string of hexadecimal digits (0-9,A-F). Each digit in the string is converted into its binary value,

and then loaded into memory, beginning at the starting address specified and continuing into sequential locations. Two hexadecimal digits are loaded into each byte of memory.

Separators between digits (spaces, commas, carriage returns) are ignored; illegal characters will terminate the command. The escape characters will terminate the command. The escape character, ESC (echoed as "$") terminates the digit string. If an odd number of hex digits have been entered, a Ø will be appended to the string.

Example

I3E10

1122334455667788 99$

This command puts the following pattern into RAM:

3E10 11 22 33 44 55 66 77 88 99

I3E40

123456789$

This command puts the following pattern into RAM:

3E40 12 34 56 78 90

Note that, since an odd number of hexadecimal digits were entered initially, a 0 was appended to the digit string.

7.2.1.4 MOVE MEMORY COMMAND, M

M<low address>,<high address>,<destination>

The M command moves the contents of memory <low address> through <high address>, inclusive, to the area of RAM beginning at <destination>. The contents of the source field remain undisturbed, unless the receiving field overlaps the source field.

The move operation is performed on a byte-by-byte basis, beginning at <low address>. Care should be taken if <destination> is between <low address> and <high address>. For example, if location 3E10 contains 1AH, the command

M3E10,3E1F,3E11

will result in locations 3E10 to 3E20 containing

"1A1A1A...".

The monitor will continue to move data until the source field is exhausted, or until it reaches address OFFFFH. If the monitor reaches address OFFFFH without exhausting the source field, it will move data into this location, then stop.

Example

M3E00,3E0F,3F00

16 bytes of memory are moved from 3E00-3E0F to 3F00-3F0F by this command.

Note: If the <low address> parameter is greater than the <high address> parameter, only the first destination address is altered.

7.2.1.5 READ HEXADECIMAL FILE, R

R

The R command reads a hexadecimal tape from the teletypewriter and loads the data into the locations specified by the address fields in the hexadecimal records. (See Section 7.3.3 for hexadecimal tape format definition.)

A typical R command will appear as follows:

.R (User should turn on the tape reader before executing this command.)

7.2.1.6 SUBSTITUTE MEMORY COMMAND, S

S<address>

The S command allows the user to examine and optionally modify memory locations individually. The command functions as follows:

(1) Type an S, followed by the hexadecimal address of the first memory location you wish to examine, followed by a space or comma.

(2) The contents of the location is displayed, followed by a dash (-).

(3) To modify the contents of the location displayed, type in the new data, followed by a space, comma, or carriage return. If you do not wish to modify the location, type only the space, comma, or carriage return.

(4) If a space or comma was typed in Step (3), the next memory location

will be displayed as in Step (2). If a carriage return as typed, the S command will be terminated.

Example

S3D50 AA- BB-CC 01-13 23-24

Location 3D50, which contains AA is unchanged, but location 3D51 (which used to contain BB) now contains CC, 3D52 (which used to contain 01) now contains 13, and 3D53 (which used to contain 23) now contains 24.

7.2.1.7 WRITE HEXADECIMAL FILE, W

W<low address>,<high address>

The W command outputs portions of memory to punched paper tape on the teletypewriter. Data is in hexadecimal format. A leader tape consisting of 60 null characters is punched followed by the memory data specified by the low/high address parameters. An end of file record is punched automatically to terminate the tape. Following the end of file record, a trailer tape is punched consisting of 60 null characters.

An example of the Write Hexadecimal file operation is as follows:
W3D00,3DAF (User should turn on tape punch before executing this command.)
This command punches out the contents of memory locations 3D00H through 3DAFH.

7.2.1.8 EXAMINE AND MODIFY CPU REGISTERS COMMAND, X

X<register identifier>

Display and modification of the CPU registers is accomplished via the X command. The X command uses <register identifier> to select the particular register to be displayed. A register identifier is a single alphabetic character denoting a register, defined as follows:

A - 8080 CPU register A

B - 8080 CPU register B

C - 8080 CPU register C

D - 8080 CPU register D

E - 8080 CPU register E

F - 8080 CPU flags byte, displayed in the form as it is stored by the "PUSH PSW" (hex code F5) instruction

H - 8080 CPU register H

L - 8080 CPU register L

M - 8080 CPU register H and L combined

P - 8080 Program Counter

S - 8080 Stack Pointer

The command operates as follows:

(1) Type an X followed by a register identifier or a carriage return.

(2) The contents of the register are displayed (two hexadecimal digits for A, B, C, D, E, F, H, and L, four hexadecimal digits for M, P, and S), followed by a dash (-).

(3) The register may be modified at this time by typing the new value followed by a space, comma, or carriage return. If no modification is desired, type only the space, comma, or carriage return.

(4) If a space or comma was typed in Step (3), the next register in sequence (alphabetical order) will be displayed as in Step (2) (unless S was just displayed in which case the command is terminated). If a carriage return was entered in Step (3), the X command is terminated.

(5) If a carriage return was typed in Step (1) above, an annotated list of all registers and their contents are displayed.

7.2.2 DEVICE DRIVERS

The monitor has device drivers for the console device including the tape reader and punch (if console device is a teletypewriter). The drivers interface through synchronous/asynchronous receiver/transmitter (USART) which is

described in Section 4 of this Hardware Reference Manual. The monitor configures the USART during a power on or reset condition to the following state:

Mode:

2 stop bits
Parity disabled
8 bit character length
Baud rate factor of 64X

Command:

No hunt mode
Request to send high
Receiver enabled
Data terminal ready low
Transmitter enabled

Care should be exercised by the user in modifying the USART mode and command since the monitor depends on the configuration defined above for device driver operation.

7.2.3 USING THE I/O SYSTEM

The user may access the four monitor I/O system routines from his program by calling the routine desired. The following paragraphs describe the routines available and their respective functions.

CI - Console Input

This routine returns an 8 bit character received from the console device to the caller in the A-register. The A-register and the CPU condition codes are affected by this operation. The entry point of this routine is 3FDH.

Example

```
CI  EQU  3FDH

    ...
    CALL CI
    STA  DATA
    ...
```

CO - Console Output

This routine transmits an 8 bit character, passed from the caller in the C-register, to the console device. The A and C registers, and the CPU condition codes, are affected by this operation. The entry point of this routine is 3FAH.

Example

```
CO    EQU   3FAH

      ...
      MVI   C,"."
      CALL  CO
```

RI - Reader Input

RI returns an 8 bit character read from the teletype reader in the A-register. If no character was read from the device (i.e., end of file), the CARRY condition code is set equal to 1, and the A-register is zeroed. If data is ready, the CARRY bit is zeroed. The reader driver contains a timer so that if no character is received from the teletype reader within a pre-established time (250 milliseconds), an end of file may be simulated and control returned to the calling program. The entry point of this routine is 400H.

Example

```
RI    EQU   400H

      ...
      CALL  RI
      JC    EOF       : END OF FILE SENSED
      STA   DATA
      ...
```

PO - Punch Output

PO transmits an 8 bit character from the calling program to the teletypewriter. PO is identical in format to CO, the only difference being the entry point address, '403H'.

7.3 MONITOR OPERATING SPECIFICATIONS

7.3.1 PRODUCT ACTIVATION INSTRUCTIONS

7.3.1.1 COLD START PROCEDURE

After power-on or reset, the monitor will begin execution at location 0 in ROM. The monitor will perform an initialization sequence, and then display a sign-on message "SBC 80P Monitor" on the console. When the monitor is ready for a command, it will prompt with a period, ".".

7.3.1.2 USE OF RAM STORAGE IN THE MONITOR

The monitor dynamically assigns its RAM stack in the first 64 bytes of RAM. The top 3 bytes in this block of RAM are reserved for a jump instruction, supplied by the user, which is used as a destination pointer for RST 7 instructions (or the optional hardwired instruction). Several additional bytes are used, below the stack, for temporary storage. Except for RAM addresses 3C00H to 3C3FH, all other RAM is available for the user.

7.3.2 ERROR CONDITIONS

7.3.2.1 INVALID CHARACTERS

The monitor checks the validity of each character as it is entered from the console. As soon as the monitor determines that the last character entered is illegal in its context, the monitor aborts the command and issues an "#" to indicate the error.

Example

```
D1400,145G#
```

The character G was encountered in a parameter list where only hexadecimal digits and delimiters are valid.

```
Y#
```

Y is not a valid command.

7.3.2.2 ADDRESS VALUE ERRORS

Some commands require an address pair of the form <low address>, <high address>. If, on these commands, the value of <low address> is greater than or equal to the value of <high address>, the action indicated by the command will be performed on the data at <low address> only.

Addresses are evaluated modulo 2^{16}. Thus, if a hexadecimal address greater than FFFF is entered, only the last 4 hex digits will be used.

Another type of address error may occur when the operator specifies a part of memory in a command which does not exist in his particular configuration. In general, if a nonexistent portion of memory is specified as the source field for an instruction, the data fetched will be unpredictable. If a nonexistent portion of memory is given as the destination field in a command, the command has no effect.

7.3.2.3 PERIPHERAL DEVICE ERRORS

Peripheral devices selected by the operator which are not ready or are nonexistent will cause undefined execution of the monitor (usually an indefinite wait for ready status in an I/O loop). This situation may be rectified by readying the device and by executing the Cold Start sequence (Section 7.3.1.1) to reinitialize the system.

7.3.3 HEXADECIMAL OBJECT FILE FORMAT FOR PAPER TAPE

Hexadecimal object file format is a way of representing a binary object file in ASCII. The ASCII character set is defined by the "American National Standard Institute, Code for Information Interchange, X3.4-1968".

The hexadecimal representation of binary is coded in ASCII. For example, the 8-bit binary value 0011 1111 is 3F in hexadecimal. To code this in ASCII one 8-bit byte containing the ASCII code for 3 and one 8-bit byte containing the ASCII code for F are required. This representation (ASCII hexadecimal) requires twice as many bytes as the binary.

A hexadecimal object file can contain either 8-bit or 4-bit data but not both. Two ASCII hexadecimal digits are used to represent both 8-bit and 4-bit data. In the case of 4-bit data, only one of the digits is

meaningful. Whether it is the high-order or the low-order digit must be known by the program reading the file and must be consistent throughout the file.

Since ASCII characters need only 7-bits for their representation, the highest-order bit of each 8-bit byte can be used as a parity bit by a program that generates the hexadecimal format object file. However, when such a file is loaded by an Intel product, the highest order bit is masked as the ASCII is converted to binary. Also, Intel software does not generate parity bits when creating object files.

The format described below is for paper tape and does not define the format for other media, which may use record separators such as the ASCII code for carriage return. On paper tape, one ASCII character requires one frame. The record format is described here according to the fields in the record.

Record Mark Field: Frame 0

The ASCII code for a colon (:) is used to signal the start of a record.

Record Length Field: Frames 1 and 2

The number of data bytes in the record is represented by two ASCII hexadecimal digits in this field. The high-order digit is in frame 1. The maximum number of data bytes in a record is 255 (FF in hexadecimal). An end-of-file record contains two ASCII zeros in this field.

Load Address Field: Frames 3 - 6

The four ASCII hexadecimal digits in frames 3-6 give the address at which the data is loaded. The high-order digit is in frame 3, the low-order digit in frame 6. The first data byte is stored in the location indicated by the load address; successive bytes are stored in successive memory locations. This field in an end-of-file record can contain the starting address of the program.

Record Type Field: Frames 7 and 8

The two ASCII hexadecimal digits in this field specify the record type. The high-order digit is in frame 7. All data records are type 0; end-of-file records are type 1. Other possible values for this field are reserved for future expansion.

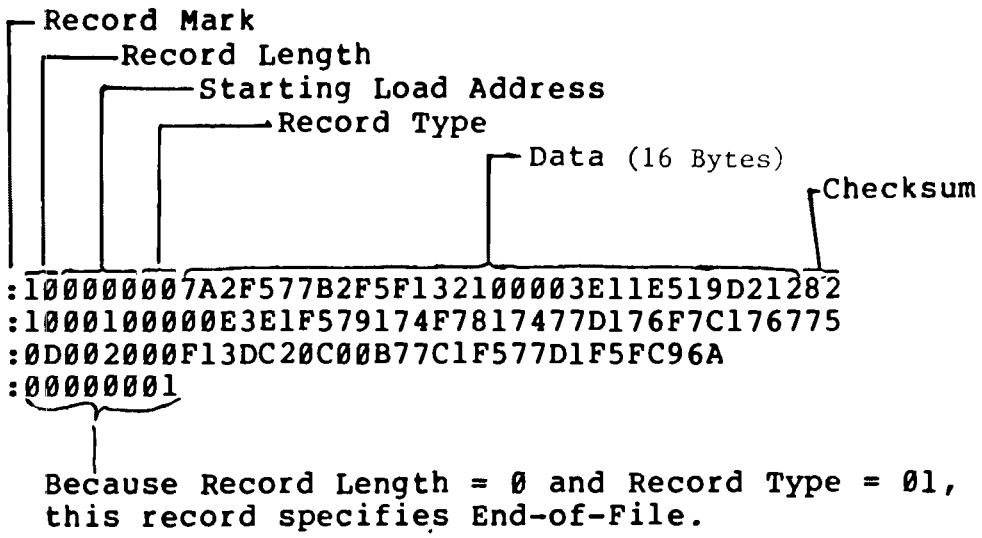
Data Field: Frames 9 to 9+2*(record length)-1

A data byte is represented by two frames containing the ASCII characters 0-9 or A-F, which represent a hexadecimal value between 0 and FF (0 and 255 decimal). The high-order digit is in the first frame of each pair. If the data is 4-bit, then either the high or low-order digit represents the data and the other digit of the pair may be any ASCII hexadecimal digit. There are no data bytes in an end-of-file record.

Checksum Field: Frames 9+2*(record length) to 9+2*(record length)+1

The checksum field contains the ASCII hexadecimal representation of the twos complement of the 8-bit sum of the 8-bit bytes that result from converting each pair of ASCII hexadecimal digits to one byte of binary, from the record length field to and including the last byte of the data field. Therefore, the sum of all the ASCII pairs in a record after converting to binary, from the record length field to and including the checksum field, is zero.

Sample Hexadecimal Tape Format:



7.4 HARDWARE GENERATED BREAKPOINTS

This section is meant to describe a method whereby the user may interrupt the operation of his program, return control to the monitor program which will save the state of the 8080 registers and finally, allow the user to restart his program from the point of interruption.

The first requirement in order to produce an interrupt, would be to attach a switch to ground or a low logic level signal from a user circuit to P1-42 or J1-49 (EXTernal INTerrupt Request 1, EXTernal INTerrupt Request 2 respectively) of the SBC 80/10 connnectors.

The second requirement would be to enter the following instructions in RAM location 3C3DH and 3C3EH.

```
3C3D   CF   RST   1      ; THIS WILL CAUSE A BREAKPOINT
3C3E   C9   RET          ; THIS WILL CAUSE THE USER PROGRAM TO
                           BE RE-ENTERED
```

An example of how this is used follows:

(1) The user will start execution of his program by entering the monitor command listed below.

.G<start of user program>

(2) Now the user may use a switch or a low logic level to force an interrupt to which the SBC 80/10 will respond with a RESTART 7.

(3) The monitor will break, save the state of the 8080 and print the following message .#3C3E.

(4) The user may now examine the 8080 registers or memory or his own circuits interfaced to the 80/10.

(5) To resume operation, the user enters the following monitor command.

.G

Note: In order to use this method of breakpointing, the user must have the interrupts enabled.

CHAPTER 8
SYSTEM UTILIZATION

8.1 SYSTEM I/O INTERFACING

The SBC 80/10, with its memory and I/O ports, is a complete computer on a single board. However, the SBC 80/10 can also serve as a primary master module within an expanded System 80/10, communicating with numerous memory and I/O modules. In this chapter we identify each of the SBC 80/10's external connections and define all signals on the external system bus.

8.1.1 ELECTRICAL CONNECTIONS

The SBC 80/10 comes on a 12.00 x 6.75 inch printed circuit board, 0.50 inch thick and weighing 12 oz. The SBC 80/10 requires DC power at the following levels:

	Without EPROM[1]	With 8708 EPROM[2]	With 2716 or 2758 EPROM[3]
V_{CC} +5V ±5%	I_{CC} = 2.9A	4.0A	4.36A
V_{CC} +12V ±5%	I_{DD} = 150mA	400mA	150mA
V_{BB} -5V ±5%	I_{BB} = 2mA	200mA	2mA
V_{AA} -12V ±5%[4]	I_{AA} = 175mA	175mA	175mA

1. Does not include power required for optional ROM/EPROM, I/O drivers or I/O terminators.
2. With four Intel 8708 EPROMs and 220Ω/330Ω terminators installed for 48 input ports; all terminator inputs low.
3. With four Intel 2716 or 2758 EPROMs and 220Ω/330Ω terminators installed for 48 input ports; all terminator inputs low.
4. Required for RS232C drivers.

The SBC 80/10 has five edge connectors, as shown in Figure 8-2. Edge connectors at the top of the module are designed for compatibility with both flat cable and round cable hardware. All parallel I/O functions are paired with an independent signal ground pin. This allows flat cable imple-

mentation to utilize an alternate signal/ground scheme for reduction of cross talk. Round cables may easily be implemented as twisted pair with an individual ground pin for every return wire. The serial connection hardware has similar flexibility but ground return lines are not extensive. The connector is wired for RS232C compatibility, thus, only one signal ground is provided.

NOTE

All pin numbers listed in the following tables refer to numbers printed on the board, not to mating connector pin positions. When specifying pin numbers for cable harnesses, use caution since pin numbering is not necessarily the same as the connector pin numbering scheme.

The Parallel I/O Interface communicates with external I/O devices via two 50-pin double-sided PC edge connectors (J1 and J2), 0.1 inch centers. External devices can be attached to J1 or J2 using any of the following mating connectors:

J1 and J2 Mating Connectors

Connector Type	Vendor	Part No.
Flat Cable	3M AMP	3415-0001 2-86792-3
Soldered	AMP VIKING TI	2-583715-3 3VH25/1JV-5 H312125
Wire-wrap	TI VIKING CDC ITT	H 311125 3VH25/1JND-5 VPB01B25D00A1 EC4A050A1A
Crimp	AMP	1-583717-1

Tables 8-1 and 8-2 provide pin lists for the J1 and J2 connectors, respectively. The following TTL line drivers and Intel terminators are all compatible with the I/O driver sockets in the Parallel I/O Interface:

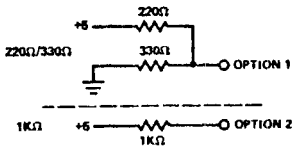
Driver	Characteristic	Sink Current (ma)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

Note: I = inverting; N.I. = non-inverting
OC = open collector

I/O Terminators:

Terminators: 220Ω/330Ω divider or 1Ωk pull up

FIGURE 8-1. TERMINATION PACK SCHEMATICS



SBC-901

SBC-902

See Appendix B for schematics

TABLE 8-1. PIN ASSIGNMENTS FOR CONNECTOR J1
(Parallel I/O Interface - Group 1)

PIN	SIGNAL	PIN	SIGNAL
1	PORT 2 - BIT 3	2	GND
3	PORT 2 - BIT 2	4	
5	PORT 2 - BIT 1	6	
7	PORT 2 - BIT 0	8	
9	PORT 2 - BIT 4	10	
11	PORT 2 - BIT 5	12	
13	PORT 2 - BIT 6	14	
15	PORT 2 - BIT 7	16	
17	PORT 3 - BIT 3	18	
19	PORT 3 - BIT 2	20	
21	PORT 3 - BIT 4	22	
23	PORT 3 - BIT 6	24	
25	PORT 3 - BIT 0	26	
27	PORT 3 - BIT 5	28	
29	PORT 3 - BIT 1	30	
31	PORT 3 - BIT 7	32	
33	PORT 1 - BIT 7	34	
35	PORT 1 - BIT 6	36	
37	PORT 1 - BIT 5	38	
39	PORT 1 - BIT 4	40	
41	PORT 1 - BIT 1	42	
43	PORT 1 - BIT 0	44	
45	PORT 1 - BIT 2	46	
47	PORT 1 - BIT 3	48	
49	EXT INTR 1/	50	GND

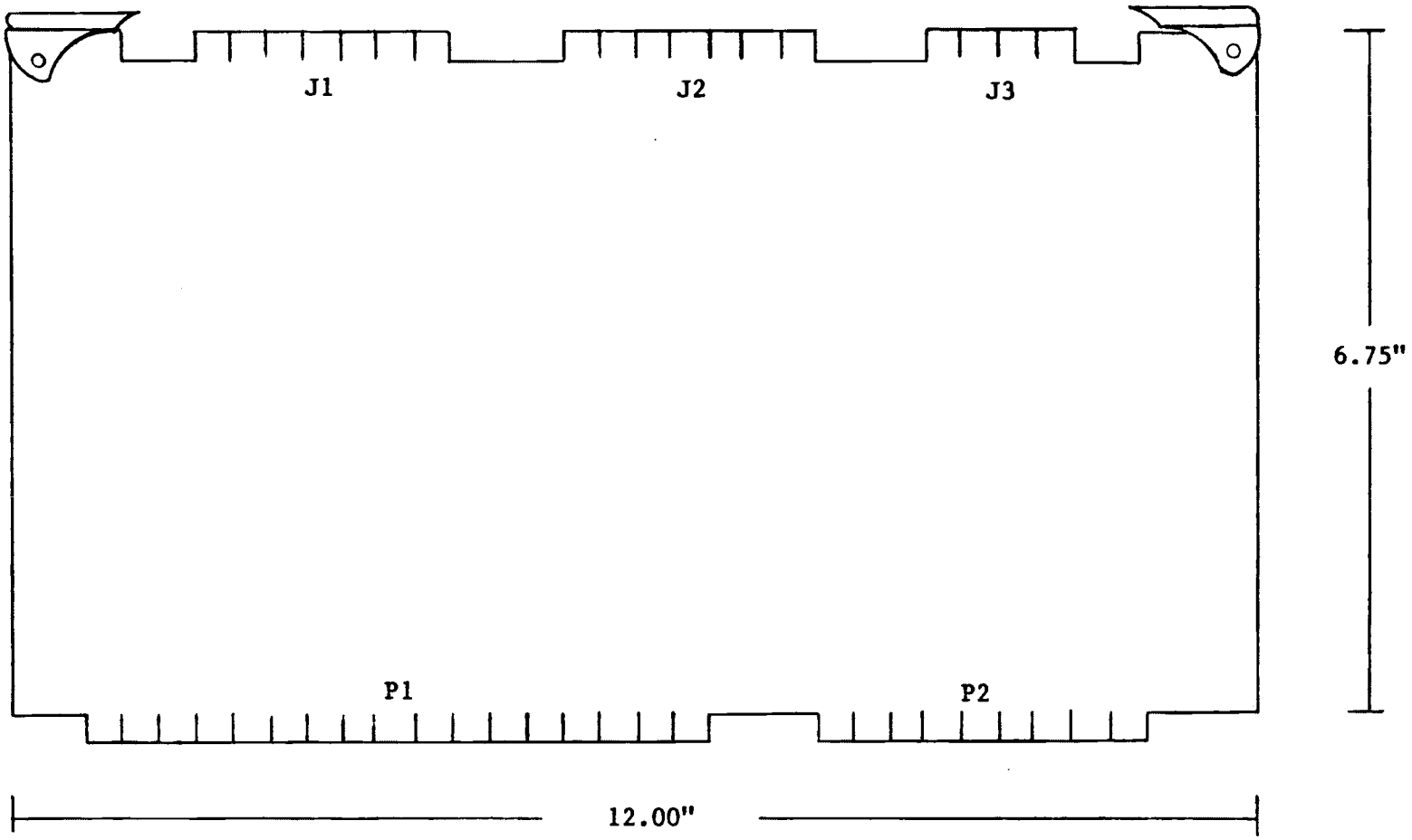


FIGURE 8-2. SBC-80/10 EDGE CONNECTORS

TABLE 8-2. PIN ASSIGNMENTS FOR CONNECTOR J2
(Parallel I/O Interface - Group 2)

PIN	SIGNAL	PIN	SIGNAL
1	GND	2	GND
3	PORT 5 - BIT 3	4	
5	PORT 5 - BIT 0	6	
7	PORT 5 - BIT 1	8	
9	PORT 5 - BIT 2	10	
11	PORT 5 - BIT 4	12	
13	PORT 5 - BIT 5	14	
15	PORT 5 - BIT 6	16	
17	PORT 5 - BIT 7	18	
19	PORT 6 - BIT 3	20	
21	PORT 6 - BIT 2	22	
23	PORT 6 - BIT 1	24	
25	PORT 6 - BIT 0	26	
27	PORT 6 - BIT 4	28	
29	PORT 6 - BIT 5	30	
31	PORT 6 - BIT 6	32	
33	PORT 6 - BIT 7	34	
35	PORT 4 - BIT 7	36	
37	PORT 4 - BIT 6	38	
39	PORT 4 - BIT 5	40	
41	PORT 4 - BIT 4	42	
43	PORT 4 - BIT 0	44	
45	PORT 4 - BIT 1	46	
47	PORT 4 - BIT 2	48	
49	PORT 4 - BIT 3	50	GND

The Serial I/O Interface communicates with an external I/O device via a 26-pin double-sides PC edge connector (J3), 0.1 inch centers. An external device can be connected to J3 using a 3M 3462-0001 flat cable connector or one of the following soldered connectors; TI H312113 or AMP 1-583715-1. Table 8-3 provides a pin list for connector J3.

The SBC 80/10 communicates with other system modules via an 86-pin double-sided edge connector (P1), 0.156 inch centers. Section 8.1.2 defines each of the external system bus signals and includes a pin list for P1 (Table 8-5).

TABLE 8-3. PIN ASSIGNMENTS FOR CONNECTOR J3
(Serial I/O Interface)

PIN	SIGNAL NAME	PIN	SIGNAL NAME
1	CHASSIS GND	2	
3	TRANSMITTED DATA	4	
5	RECEIVED DATA	6	TTY RD CONTROL
7	REQ TO SEND	8	
9	CLEAR TO SEND	10	
11	DATA SET READY	12	
13	GND	14	Tx CLK/DATA TERMINAL RDY
15	DATA CARRIER RETURN	16	TTY RD CONTROL RETURN
17		18	
19		20	
21		22	RECEIVE CLK/TTY Rx DATA RETURN
23	TTY Rx DATA		
25	TTY Tx DATA	24	TTY Tx DATA RETURN
		26	GND

The 60-pin double-sided edge connector labeled P2 in Figure 8-1 allows access to various test points on the SBC 80/10 (see Table 8-4). The following wire-wrap connectors will attach to P2:

CDC VPB01B30A00A2,
TI H311130 and
AMP PE5-14559

TABLE 8-4. PIN ASSIGNMENTS FOR CONNECTOR P2
(Auxilliary Connector)

SIGNAL NAME	PIN ASSIGNMENT	COMMENT
OSC	P2 - 28	TEST POINT
RAM 3C00 ENABLE/	P2 - 30	
RAM 3D00 ENABLE/	P2 - 32	
RAM 3E00 ENABLE/	P2 - 34	
RAM 3F00 ENABLE/	P2 - 36	
OSC INH/	P2 - 40	
DATA BUS INH/	P3 - 42	
BAUD RATE CLK TTY	P2 - 44	
COUNT 1 ENABLE 1	P2 - 46	
BAUD RATE CLK	P2 - 50	
COUNT 2 ENABLE/	P2 - 52	
TIME OUT ENABLE/	P2 - 54	
B & C CLK SET/	P2 - 55	
STATUS STROBE	P2 - 57	
RDY IN INH/	P2 - 57	
BAUD RATE CLEAR/	P2 - 58	
OSC/2	P2 - 60	TEST POINT

8.1.2 EXTERNAL SYSTEM BUS SUMMARY

A significant measure of the System 80/10's power and flexibility can be attributed to its system bus. In expanded systems, the bus structure allows for master-slave relationships between the various system modules. The bus includes its own clock (BCLK/) which is derived independently from the processor clock. Actual transfers via the bus, however, proceed asynchronously with respect to the bus clock. Thus, the transfer speed is dependent on the transmitting and receiving devices only. Once a module has gained control of the bus by activating the HOLD/ input to the SBC 80/10, single or multiple read/write transfers can proceed at a maximum rate of 5 million data bytes per second. The 16 system address lines allow the

SBC 80/10 to support up to 65,536 bytes of storage. The signal lines on the system bus are defined as follows:

BCLK/ — Bus clock; used to synchronize bus control circuits on all master modules. BCLK/ has a period of 101.725 nanoseconds (9.8304 MHz frequency), 30%-70% duty cycle. BCLK/ may be slowed, stopped or single stepped, if desired.

INIT/ — Initialization signal; resets the entire system to a known internal state.

BPRN/ — Bus priority input signal; indicates to the SBC 80/10 that a higher priority master module is requesting use of the system bus. BPRN/ suspends the processing activity and drivers of the SBC 80/10.

BUSY/ — Bus busy signal; indicates that the bus is currently in use. BUSY/ prevents all other master modules from gaining control of the bus. BUSY/ is driven by the HLDA/ output from the SBC 80/10 in response to a HOLD/ input. It indicates that the bus is available.

MRDC/ — Memory read command; indicates that the address of a memory location has been placed on the system address lines and specifies that the contents of the addressed location are to be read and placed on the system data bus.

MWTC/ — Memory write command; indicates that the address of a memory location has been placed on the system address lines and that a data word has been placed on the system data bus. MWTC/ specifies that the data word is to be written into the addressed memory location.

IORC/ — I/O read command; indicates that the address of an input port has been placed on the system address bus and that the data at that input port is to be read and placed on the system data bus.

IOWC/ — I/O write command; indicates that the address of an output port has been placed on the system address bus and that the contents of the system data bus are to be output to the addressed port.

XACK/ — Transfer acknowledge signal; the required response of an external memory location or I/O port which indicates that the specified read/write operation has been completed. That is, data has been placed on, or accepted from, the system data bus lines.

AACK/ — Advance acknowledge signal; used with 8080 CPU-based systems. 8080/ is an advance acknowledge, in response to a memory read command, that allows the memory to complete the access without requiring the CPU to wait.

CCLK/ — Constant clock; provides a clock signal of constant frequency (9.8304 MHz) for use by option memory and I/O expansion boards. CCLK/ coincides with BCLK/ and has a period of 101.725 nanoseconds, 30%-70% duty cycle.

EXT INTR/ — Externally generated interrupt request.

ADRO/-ADRF/ — 16 Address lines: used to transmit the address of the memory location or I/O port to be accessed. ADRF/ is the most significant bit.

DATO/-DAT7/ — Bidirectional data lines; used to transmit/receive information to/from a memory location or I/O port. DAT7/ is the most significant bit.

8.1.3 RS232C CABLING

When the Serial I/O Interface is configured as an RS232C interface, the J3 edge connector is presented to the user's terminal or modem. A 26-pin mating connector, 3M 3462-0001, should be attached to the J3 edge connector on the SBC 80/10 and to a 25-wire flat cable, 3M 3349/25. The flat cable is, in turn, attached to the RS232C pin-compatible connector,

TABLE 8-5. PIN ASSIGNMENTS FOR CONNECTOR P1
(System Bus)

	(COMPONENT SIDE)			(CIRCUIT SIDE)		
	PIN	MNEMONIC	DESCRIPTION	PIN	MNEMONIC	DESCRIPTION
POWER SUPPLIES	1	GND	Signal GND	2	GND	Signal GND
	3	VCC	+ 5VDC	4	VCC	+ 5VDC
	5	VCC	+ 5VDC	6	VCC	+ 5VDC
	7	VDD	+12VDC	8	VDD	+12VDC
	9	VBB	- 5VDC	10	VBB	- 5VDC
	11	GND	Signal GND	12	GND	Signal GND
BUS CONTROLS	13	BCLK/	Bus Clock	14	INIT/	Initialize
	15	BPRN	Bus Pri. In	16	1	
	17	BUSY/	Bus Busy	18	1	
	19	MRDC/	Mem Read Cmd	20	MWTC/	Mem Write Cmd
	21	IORC/	I/O Read Cmd	22	IOWC/	I/O Write Cmd
	23	XACK/	XFER Acknow	24	1	
SPARES	25	AACK/	Special	26		
	27			28		
	29			30		
	31	CCLK/	Constant Clock	32		
	33	1		34		
INTERRUPTS	35	1		36	1	
	37	1		38	1	
	39	1		40	1	
	41	1		42	INTR1/	Interrupt request
ADDRESS	43	ADRD/	Address Bus	44	ADRF/	Address Bus
	45	ADRC/		46	ADRD/	
	47	ADRA/		48	ADRB/	
	49	ADR8/		50	ADR9/	
	51	ADR6/		52	ADR7/	
	53	ADR4/		54	ADR5/	
	55	ADR2/		56	ADR3/	
	57	ADR0/		58	ADR1/	
DATA	59	1	Data Bus	60	1	Data Bus
	61	1		62	1	
	63	1		64	1	
	65	1		66	1	
	67	DAT6/		68	DAT7/	
	69	DAT4/		70	DAT5/	
	71	DAT2/		72	DAT3/	
	73	DAT0/		74	DAT1/	
POWER SUPPLIES	75	GND	Signal GND	76	GND	Signal GND
	77	VBB 1	-10VDC	78	VBB 1	-10VDC
	79	VAA	-12VDC	80	VAA	-12VDC
	81	VCC	+ 5VDC	82	VCC	+ 5VDC
	83	VCC	+ 5VDC	84	VCC	+ 5VDC
	85	GND	Signal GND	86	GND	Signal GND

1 Used by Intellec® MDS Bus.

3M 3483-1000. Table 8-6 equates the J3 edge connector pins with the associated RS232-compatible pins on the 3M 3483-1000 connector.

TABLE 8-6. J3/RS232C CONNECTOR PIN CORRESPONDENCE

J3 CONNECTOR PIN NO.	RS232C CONNECTOR PIN NO.
1	1
2	14
3	2
4	15
5	3
6	16
7	4
8	17
9	5
10	12
11	6
12	19
13	7
14	20
15	8
16	21
17	9
18	22
19	10
20	23
21	11
22	24
23	12
24	25
25	13

8.2 TELETYPEWRITER MODIFICATIONS

The ASR-33 Teletypewriter must be modified for use with the System 80/10. Appendix H is a procedure for modifying the ASR-33 Teletypewriter.

CHAPTER 9

INTERFACING TO MULTIBUS MASTER

The System 80/10's bus structure permits interfacing to one other multibus-compatible master module. This interface is accomplished using the serial priority scheme as shown in Figure 9-1, using the Intel SBC 604 cardcage/backplane. The System 80/10 does not provide the Bus Priority Request Out (BPRO/) signal and therefore, the System 80/10 can only be used with one other multibus master. For these configurations, the SBC 80/10 or SBC 80/10A must always have lower priority than the other Multibus Master and a wire must be added from Master's BREQ/ (pin 12) to the SBC 80/10 BPRN (pin 15). In the configuration shown in Figure 9-1 the SBC 80/10 acquires control of the multibus whenever BREQ/ generated by the Diskette Controller is in the high state. This occurs whenever the Diskette Controller is not using the multibus. Similarly BREQ/ is driven to the low state when the Diskette Controller acquires control of the Multibus disabling the SBC 80/10 from accessing the multibus.

For a detailed description of Multibus interfacing refer to the Intel Multibus Interfacing Application Note (AP-28).

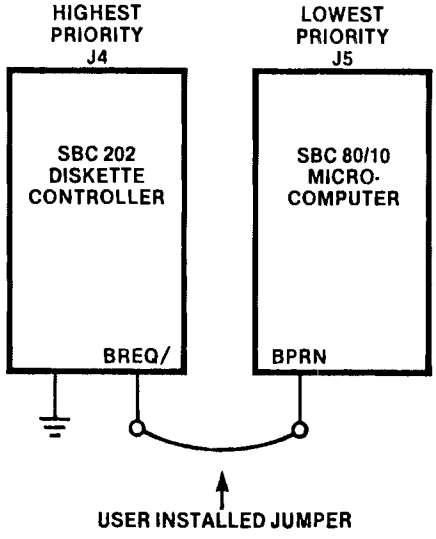


Figure 9-1. Serial Priority Configuration with SBC 80/10 and another Multibus Master

APPENDIX A

SYSTEM SPECIFICATIONS

A.1 GENERAL SYSTEM SPECIFICATIONS

WORD SIZE

Instruction: 8, 16, or 24 bits

Data: 8 bits

CYCLE TIME

Basic Instruction Cycle: 1.95 µsec

Note: Basic instruction cycle is defined as the fastest instruction (i.e., four clock cycles).

MEMORY ADDRESSING

On-Board ROM/PROM: 0-1FFF

On-Board RAM: 3C00-3FFF

MEMORY CAPACITY

On-Board ROM/PROM: 4K bytes (sockets only)

On-Board ROM/PROM: 8K bytes (sockets only) (SBC 80/10A only)

On-Board RAM: 1K bytes

Off-Board Expansion: Up to 48K bytes using optional RAM, ROM, and PROM expansion boards.

Note: ROM/PROM may be added in 1K byte increments for the SBC 80/10 and SBC 80/10A, or in 2K bytes on the SBC 80/10A.

I/O ADDRESSING

On-Board Programmable I/O (see Table A-1).

Port	8255 No. 1			8255 No. 2			8255 No. 1 Control	8255 No. 1 Control	USART Data	USART Control
	1	2	3	4	5	6				
Address	E4	E5	E6	E8	E9	EA	E7	EB	EC	ED

I/O CAPACITY

Parallel: 48 programmable lines (see Table A-1).

Note: Expandable with optional I/O boards.

SERIAL BAUD RATES

Frequency (kHz) (Jumper Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous (Program Selectable)	
		÷ 16	÷ 64
307.2	—	19200	4800
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
6.98	6980	—	110

SERIAL COMMUNICATIONS CHARACTERISTICS

Synchronous:

5-8 bit characters

Internal or external character synchronization

Automatic Sync Insertion

Asynchronous:

5-8 bit characters

Break character generation

1, 1½, or 2 stop bits

False start bit detectors

INTERRUPTS

Single-level with on-board logic that automatically vectors processor to location 38_{16} using RESTART 7 instruction. Interrupt requests may originate from user-specified I/O (2), the programmable peripheral interface (2), or USART (2).

TABLE A-1

INPUT/OUTPUT PORT MODES OF OPERATION

PORT	NO. OF LINES	MODE OF OPERATION					
		UNIDIRECTIONAL				BIDIRECTIONAL	CONTROL
		INPUT		OUTPUT			
		UNLATCHED	LATCHED & STROBED	LATCHED	LATCHED & STROBED		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	8	X		X			X[1]
4	8	X		X			
5	8	X		X			
6	4	X		X			
	4	X		X			

1. Note: Port 3 must be used as a control port when either Port 1 or Port 2 are used as a latched and strobed input or a latched and strobed output or Port 1 is used as a bidirectional port.

INTERFACES

Bus: All signals TTL compatible

Parallel I/O: All signals TTL compatible

Serial I/O: RS232C or a 20 mA current loop TTY interface (jumper-selectable)

Interrupt Requests: All TTL compatible (active-low)

SYSTEM CLOCK

2.048 MHz $\pm$0.1%

CONNECTORS

Interface	No. of Double-Sided Pins	Centers (in.)	Mating Connectors
Parallel I/O (2)	50	0.1	3M 3415-000 Flat
Serial I/O	26	0.1	3M 3462-0001 Flat or AMP 88106-1 Flat

PHYSICAL CHARACTERISTICS

Height: 8.90 cm (3.5 in.)

Width:

At Front Panel: 48.3 cm (19 in.)

Behind Front Panel: 43.2 cm (17 in.)

Depth: 50.8 cm (20 in. with all protrusions)

Weight: 13.6 kg. (30 lbs.)

ELECTRICAL CHARACTERISTICS

Input Power:

Frequency: 47-63 Hz

Voltage:

Standard: 115 VAC ±10%

Option: 230 VAC ±10%

AIR CIRCULATION

2 x 37 CFM, 74 CFM total

OUTPUT POWER AVAILABLE FOR EXPANSION BOARDS:

Voltage	Supply Current	Power Available without PROM & Termination Packs Installed	Power Available with PROM & Termination Packs Installed*	Over-Voltage Protection
+12	2A	1.86A	1.6A	+14 to +16 volts
+5	14A	11.1A	10A	5.8 to 6.6 volts
-5	0.9A	0.898A	0.7A	5.8 to -6.6 volts
-12	0.8A	0.625A	0.625A	-14 to -16 volts

*PROMs are 4 each of 8708's; Termination Packs are 10 each of 220Ω/330Ω.

LINE DRIVERS AND TERMINATORS

I/O Drivers:

The following line drivers and terminators are all compatible with the I/O driver sockets on the SBC 80/10.

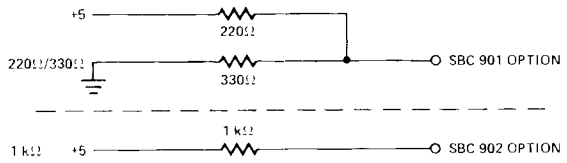
Driver	Characteristic	Sink Current (mA)	Driver	Characteristic	Sink Current (mA)
7438	I,OC	48	7409	NI,OC	16
7437	I	48	7408	NI	16
7432	NI	16	7403	I,OC	16
7426	I,OC	16	7400	I	16

Note: I = inverting; N.I. - non-inverting; O.C. = open collector.

Port 1 has 25 mA totem-pole drivers and 1 kΩ pull-up.

I/O Terminators:

Terminators: 220Ω/330Ω divider or 1 kΩ pull-up.



Bus Drivers:

Function	Characteristics	Sink Current (mA)
Data	Tri-state	25
Address	Tri-state	25
Commands	Tri-State	25

ENVIRONMENTAL

Operating Temperature: 0°C to 50°C

Non-operating Temperature: -40°C to 85°C

SYSTEM MONITOR

Addresses:

$0000\text{-}0560_H$(ROM); $3C00_H\text{-}3C3F_H$(RAM)

Commands:

Display Memory (D)

Program Execute (G)

Insert Instructions into Memory (I)

Move Memory (M)

Read Hexadecimal File (R)

Substitute Memory (S)

Write Hexadecimal File (W)

Examine and Modify CPU Registers (X)

Drivers:

Console Input

Console Output

Reader Input

Punch Output

Breakpoints:

A hardware breakpoint capability may be implemented by an interrupt service routine beginning at RAM location $3C3D_H$. Typically, a 2-byte call is used. Interrupt generation at the breakpoint may be accomplished by user hardware or by insertion of single byte calls at instruction boundaries.

ENVIRONMENTAL TEST SPECIFICATION

(1) Line Voltage and Frequency Variation

a) 115v ± 10%, 60hz ± 10%
b) 230v ± 10%, 50hz ± 5%

(2) AC Leakage (Personnel Hazard) (L or N to ground)

a) Less than 1.46 ma rms at 115v, 60hz
b) Less than 5 ma pk at 66hz and 253 volts

(3) Insulation Resistance and High Potential (L & N to Ground)

a) Less than 210 ua of leakage current at 2,100 volts DC
b) No arcing or breakdown as indicated by current flunctuation.
c) Minimum insulation of 10 MΩ.

(4) Altitude

Non-operating: 25,000 ft. (10.8 in. hg.), test for 30 minutes minimum after stabilization

Operating: 15,000 ft. (16.8 in. hg.), test for 30 minutes at 25°C after stabilization - the relative humidity is uncontrolled

(5) Temperature

a) Non-operating: -40°C to +185°C, test for 2 hours minimum after stabilization

b) Operating: 0°C to 50°C, test for 16 hours minimum after stabilization

c) Exposure: -20°C to +65°C, test for 1 hour minimum after stabilization

(6) Humidity

a) Five 24 hour cycles from 50% to 95% RH and from +25°C to +40°C, product operating at all times.
b) Condensation at +40°C and 95% RH with product operating at all times.

(7) Vibration

a) Test at all three mutually perpendicular planes with product operating at all times.
b) 15 minutes of cycling (15 one minute cycles) from 10-55hz and with a .010" pk-pk excursion.
c) 3 minutes at .010" pk-pk excursion at major resonant points.

(8) Shock

a) 3 shocks on each of six sides for a total of 18 shocks.
b) Level 30G; duration 11 ms; shape ½ sinewave

(9) Transportation Environment (Packaging)

a) Drop Test: 12 free-fall, oriented drops (4 corner, 5 flat, and 3 edge) on a concrete floor or slab.

1) from 30 inches high
2) from 20G to 50G shock range

b) Vibration: 0.5 inch amplitude, low frequency circular synchronous vibration at an acceleration of 1 ± 0.1G

1) on all positions used by common carriers in transporting the package.
2) total vibration time is 2 hours.

(10) Finished Product Test

a) Unit is tested after final assembly for 48 hours at ambient conditions.
b) System test consists of 10 sub-tests. They are as follows:

1) Programmable Parallel I/O Port #1 Test
2) Programmable Parallel I/O Port #2 Test
3) RAM Memory Test 3D00-3FFF
4) RAM Memory Test 3C00-3CFF
5) CPU Instruction Test
6) Off Board I/O Test
7) Halt Test
8) Ready Time-out (non-existent memory and I/O) Test
9) Off-board Memory Access Test
10) External Interrupt Test

A.2 SBC 80/10 AND SBC 80/10A SPECIFICATIONS

TABLE A-2. DC POWER REQUIREMENTS

	Without EPROM[1]	With 8708 EPROM[2]	With 2716 or 2758 EPROM[3]
V_{CC} +5V ±5%	I_{CC} = 2.9A	4.0A	4.36A
V_{CC} +12V ±5%	I_{CC} = 150mA	400mA	150mA
V_{BB} -5V ±5%	I_{BB} = 2mA	200mA	2mA
V_{AA} -12V ±5%[4]	I_{AA} = 175mA	175mA	175mA

1. Does not include power required for optional ROM/EPROM, I/O drivers or I/O terminators.
2. With four Intel 8708 EPROMs and 220Ω/330Ω terminators installed for 48 input ports; all terminator inputs low.
3. With four Intel 2716 or 2758 EPROMs and 220Ω/330Ω terminators installed for 48 input ports; all terminator inputs low.
4. Required for RS232C drivers.

TABLE A-3. AC CHARACTERISTICS (WITH BUS EXCHANGE)

PARAMETER	OVERALL		WITH BUS EXCHANGE				DESCRIPTION	REMARKS
			READ		MEMORY WRITE			
	MIN (ns)	MAX (ns)	MIN (ns)	MAX (ns)	MIN (ns)	MAX (ns)		
t_{AS}	82		82		658		Address Setup Time to Command	
t_{AH}	61		0		61		Address Hold Time	
t_{DS}	140		—		140		Data Setup Time to Command	
t_{DH}	61		0		61		Data Hold Time	
$t_{ACK\emptyset}$			68	191			First ACK Sampling Point of Current Cycle	Generates 0 Wait States
t_{ACK1}			551	684	-60	132	Second ACK Sampling Point of Current Cycle	Generates 1 Wait State
t_{ACK2}			1034	1174	423	625	Third ACK Sampling Point of Current Cycle	Generates 2 Wait States
t_{CY}	483	493					ACK & BPRN Sample Cycle Time	
t_{WC}			596	796	1412	1516	Command Width	Read, 0 Wait States Write, 2 Wait States
t_{ACC}				344			Read Access Time	▷
t_{8KD}				68		-60	Advanced ACK Response Time for Minimum Delay	▷
t_{8KO}	0	100	0	100	0	100	Advanced ACK Turn Off Delay	▷
t_{XKD}	0		0				XACK Delay From Valid Data or Write	
t_{XKO}	0	100	0	100	0	100	XACK Turn Off Delay	
t_{DBS}		3500					Bus Sample to Exchange Initiation	▷ Assume HOLD/ becomes active prior to DAD instruction
t_{BS}	0	493					BPRN Sampling Point Delay	
t_{DBY}	358	700					Bus Busy Turn On Delay	

▷ Memory and I/O access occurs with no wait states.

TABLE A-4. AC CHARACTERISTICS (WITH CONTINUOUS BUS CONTROL)

PARAMETER	OVERALL		CONTINUOUS BUS CONTROL				DESCRIPTION	REMARKS
			READ		MEMORY WRITE			
	MIN (ns)	MAX (ns)	MIN (ns)	MAX (ns)	MIN (ns)	MAX (ns)		
t_{AS}	82		82		658		Address Setup Time to Command	
t_{AH}	79		0		79		Address Hold Time	
t_{DS}	140		–		140		Data Setup Time to Command	
t_{DH}	79		0		79		Data Hold Time	
$t_{ACK\emptyset}$			68	191			First ACK Sampling Point of Current Cycle	Generates 0 Wait States
t_{ACK1}			551	684	-60	132	Second ACK Sampling Point of Current Cycle	Generates 1 Wait State
t_{ACK2}			1034	1177	423	625	Third ACK Sampling Point of Current Cycle	Generates 2 Wait States
t_{CY}	483	493					ACK & BPRN Sample Cycle Time	
t_{SEP}	259		613		259	▷	Command Separation	
t_{WC}			596	796	1412	1516	Command Width	Read, 0 Wait States Write, 2 Wait States
t_{ACC}	344			344			Read Access Time	◁
t_{8KD}				68		-60	Advanced ACK Response Time for Minimum Delay	◁
t_{8KO}	0	100	0	100	0	100	Advanced ACK Turn Off Delay	
t_{XKD}	0		0				XACK Delay From Valid Data or Write	
t_{XKO}	0	100	0	100	0	100	XACK Turn Off Delay	
t_{BCY}	I07	110					Bus Clock Cycle Time	80/10 Generator
t_{BW}	25	85					Bus Clock Low or High Periods	80/10 Generator
t_{INT}	3000						Initialization Width	After all voltages have stabilized

◁ MAX assumes no acknowledge delays.

▷ Write Command to next Read Command separation.

TABLE A-5. DC CHARACTERISTICS

SIGNALS	SYMBOL	PARAMETER DESCRIPTION	TEST CONDITIONS	MIN	MAX	UNITS
ADRØ/-ADRF/	V_{OL}	Output Low Voltage	I_{OL} = 50 mA		0.6	V
ADDRESS	V_{OH}	Output High Voltage	I_{OH} = -10 mA	2.4		V
	V_{IL}	Input Low Voltage			0.95	V
	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.45		-0.25	mA
	I_{IH}	Input Current at High V	V_{IN} = 5.25V		10	µA
	*C_L	Capacitive Load			18	pF
MROC/,MWTC/	V_{OL}	Output Low Voltage	I_{OL} = 32 mA		0.4	V
IORC/,IOWC/	V_{OH}	Output High Voltage	I_{OH} = -5.2 mA	2.4		V
	I_{LH}	Output Leakage High	V_O = 2.4		40	µA
	I_{LL}	Output Leakage Low	V_O = 0.4		-40	µA
	*C_L	Capacitive Load			15	pF
DATØ/-DAT7/	V_{OL}	Output Low Voltage	I_{OL} = 50 mA		0.6	V
	V_{OH}	Output High Voltage	I_{OH} = -10 mA	2.4		V
	V_{IL}	Input Low Voltage			0.95	V
	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.45		-0.25	mA
	I_{LH}	Output Leakage High	V_O = 5.25		100	µA
	I_{LL}	Output Leakage Low	V_O = 0.45		100	µA
	*C_L	Capacitive Load			18	pF
INT1/	V_{IL}	Input Low Voltage			0.8	V
	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.4V		-2.2	mA
	I_{IH}	Input Current at High V	V_{IN} = 5.5V		1	mA
	*C_L	Capacitive Load			18	pF
BPRN/,XACK	V_{IL}	Input Low Voltage			0.8	V
AACK	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.5		-2.6	mA
	I_{IH}	Input Current at High V	V_{IN} = 2.7V		0.30	mA
	*C_L	Capacitive Load			18	pF
BUSY/	V_{OL}	Output Low Voltage	I_{OL} = 25 mA		0.4	V
OPEN COLLECTOR	*C_L	Capacitive Load			20	pF
INT	V_{OL}	Output Low Voltage	I_{OL} = 32 mA		0.6	V
(SYSTEM RESET)	V_{OH}	Output High Voltage	OPEN COLLECTOR			
	V_{IL}	Input Low Voltage			0.7	V
	V_{IH}	Input High Voltage		2.0		V
	I_{IH}	Input Current at High V	V_{IN} = 5.5		0.2	mA
	I_{IL}	Input Current at Low V	V_{IN} = 0.3		-0.9	mA
	*C_L	Capacitive Load			38	pF
BCLK + CCLK	V_{OL}	Output Low Voltage	I_{OL} = 20 mA		0.5	V
	V_{OH}	Output High Voltage	I_{OH} = -1 mA	2.7		V
	*C_L	Capacitive Load			18	pF

***Capacitive values are approximations only.**

TABLE A-5. DC CHARACTERISTICS (Continued)

SIGNALS	SYMBOL	PARAMETER DESCRIPTION	TEST CONDITIONS	MIN	MAX	UNITS
EXT INTRØ/	V_{IL}	Input Low Voltage			0.8	V
	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.4V	6.8		mA
	I_{IH}	Input Current at High V	V_{IN} = 5.5V		2	mA
	*C_L	Capacitive Load			18	pF
PORT E4	V_{OL}	Output Low Voltage	I_{OL} = 20 mA		.45	V
BIDIRECTIONAL	V_{OH}	Output High Voltage	I_{OH} = -12 mA	2.4		V
DRIVERS	V_{IL}	Input Low Voltage			.95	V
	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.45		-5.25	mA
	I_{LH}	Output Leakage High	V_O = 5.25		.30	mA
	*C_L	Capacitive Load			18	pF
8255	V_{OL}	Output Low Voltage	I_{OL} = 1.7 mA		.45	V
DRIVER/ RECEIVER	V_{OH}	Output High Voltage	I_{OH} = -50 µA	2.4		V
	V_{IL}	Input Low Voltage			.8	V
	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.45		10	µA
	I_{IH}	Input Current at High V	V_{IN} = 5.0		10	µA
	*C_L	Capacitive Load			18	pF

***Capacitive values are approximations only.**

TABLE A-6. SBC BOARDS COMPATIBLE CONNECTOR HARDWARE

FUNCTION	# OF PINS	CENTERS (inches)	CONNECTOR TYPE	VENDOR	VENDOR PART #	INTEL PART #
PARALLEL I/O	25/50	0.1	FLAT CRIMP ↓	3M 3M AMP ANSLEY SAE	3415-0000 WITH EARS 3415-0001 W/O EARS 88083-1 609-5015 SD6750 SERIES	SBC-955 (CABLE ASSY.)
SERIAL I/O	13/26	0.1	FLAT CRIMP ↓	3M AMP ANSLEY SAE	3462-0001 CRIMP 88106-1 609-2615 SD6726 SERIES	SBC-956 (CABLE ASSY.)
PARALLEL I/O	25/50	0.1	SOLDERED ↓	AMP VIKING TI	2-583485-6 3VH25/1JV5 H312125	N/A
SERIAL I/O	13/26	0.1	SOLDERED ↓	TI AMP	H312113 1-583485-5	N/A
AUXILIARY ▷	30/60	0.1	SOLDERED ↓	VIKING TI	3VH30/1JN5 H312130	N/A
BUS ▷	43/86	0.156	SOLDERED ↓	CDC MICRO PLASTICS ARCO VIKING	VPB01E43D00A1 ▷▷ MP-0156-43-BW-4 AE443WP1 LESS EARS 2VH43/1AV5	N/A
PARALLEL I/O ▷	25/50	0.1	WIREWRAP ↓	TI VIKING CDC ITT CANNON	H311125 3VH25/1JND5 VPB01B25D00A1 ▷▷ EC4A050A1A	N/A
SERIAL I/O ▷	13/26	0.1	WIREWRAP	TI	H311113	N/A
AUXILIARY ▷ ▷	30/60	0.1	WIREWRAP ↓	CDC TI	VPB01B30A00A2 ▷▷ H311130	MDS-980
BUS ▷ ▷	43/86	0.156	WIREWRAP ↓	CDC CDC VIKING	VFB01E43D00A1 or ▷▷ VPB01E43A00A1 2VH43/1AND5	MDS-985
SBC 201 SBC 501 SBC 508 SBC 905, etc.	50/100	0.1	SOLDER TAIL	VIKING	3VH50/1JN5	MDS-990
			SOLDER PAK (RAYCHEM)	CDC	VPB04B50E00A1E ▷▷	N/A

▷ Connector heights are not guaranteed to conform to OEM packaging equipment. Intel OEM and Intellec® Development System motherboards offer complete mechanical compatibility.

▷ Wirewrap pin lengths are not guaranteed to conform to OEM packaging equipment. Intel connectors and OEM and Intellec® Development System motherboards offer complete mechanical compatibility.

▷▷ CDC VPB01 ..., VPB02 ..., VPB04 ..., etc. are identical connectors with different electroplating thicknesses or metal surfaces.

NOTE: See next page for vendor addresses, telephone numbers and TWX numbers.

VENDORS ADDRESSES

The following information is for our customer's convenience only. Intel does not represent these vendors, guarantee availability nor continued quality of their products.

CDC CONNECTOR DIVISION
31829 W. LaTienda Drive
Westlake Village, CA 91361 USA

213-889-3535
TWX 910-494-1224

VIKING INDUSTRIES, INC.
21001 Nordhoff Street
Chatsworth, CA 91311 USA

213-341-4330
TWX 910-494-2094

Connector Systems
TEXAS INSTRUMENTS, INC.
34 Forest Street
Attleboro, MA 02703 USA

617-222-2800

AMP Incorporated
P.O. Box 3608
Harrisburg, PA 17105 USA

717-564-0100
TWX 510-657-4110

T & B/ANSLEY
Subsidiary of Thomas & Betts Corporation
3208 Humbolt Street
Los Angeles, CA 90031 USA

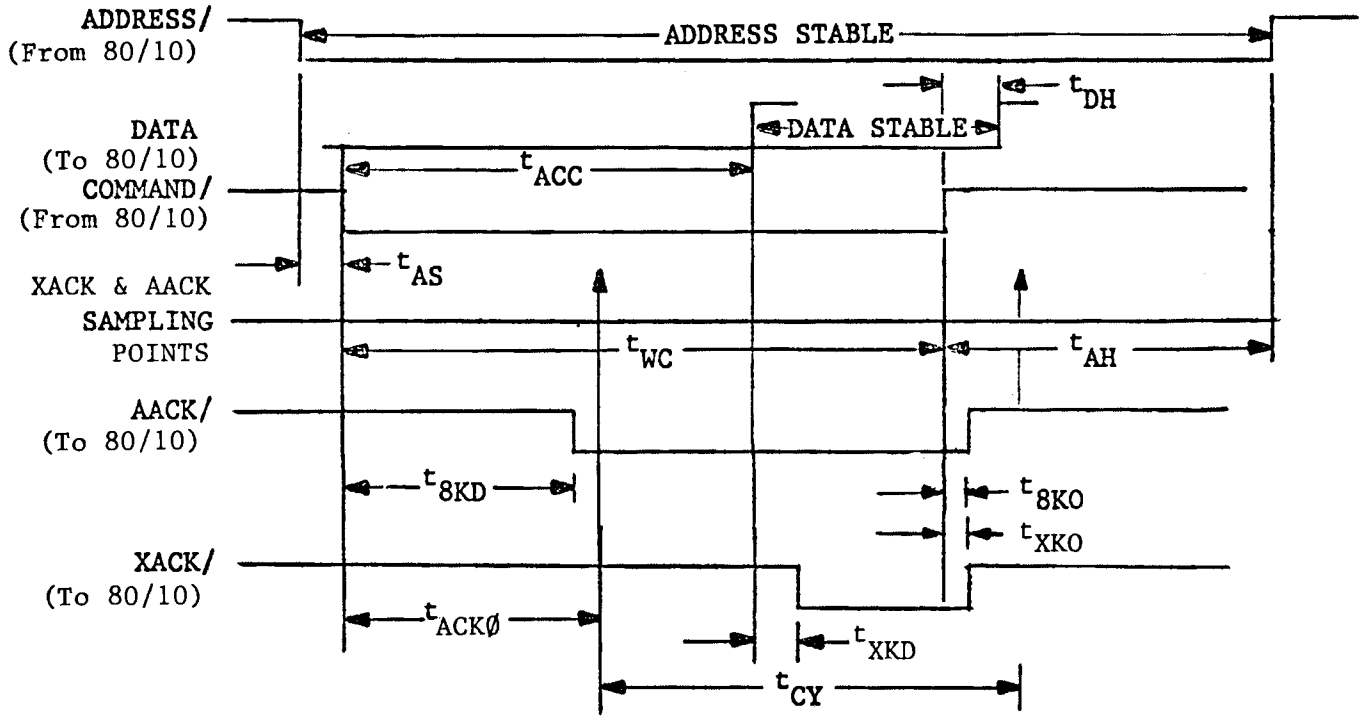
213-223-2331
TWX 910-321-3938

STANDFORD APPLIED ENGINEERING, INC. (SAE)
340 Martin Avenue
Santa Clara, CA 95059 USA

408=243-9200
TWX 910-338-0132

3M Connectors
Electronic Products Division, Bldg. 223-4E
3M COMPANY
3M Center
St. Paul, MN 55101 USA

612-733-1110



* FIGURE A-1. MEMORY AND I/O READ TIMING (CONTINUOUS BUS CONTROL)

*NOT DRAWN TO SCALE.

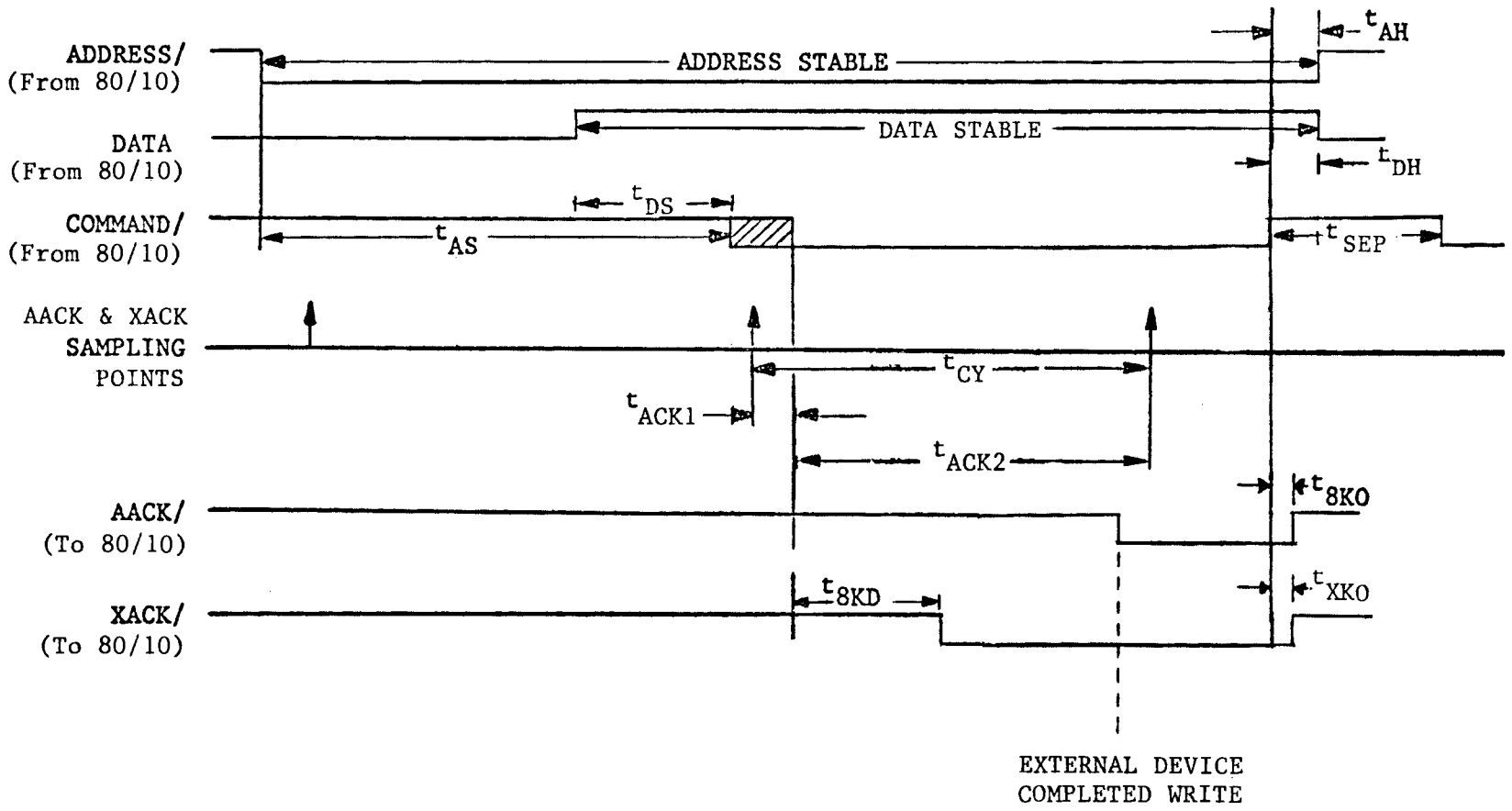


*FIGURE A-2. MEMORY AND I/O WRITE TIMING (CONTINUOUS BUS CONTROL)

*NOT DRAWN TO SCALE.

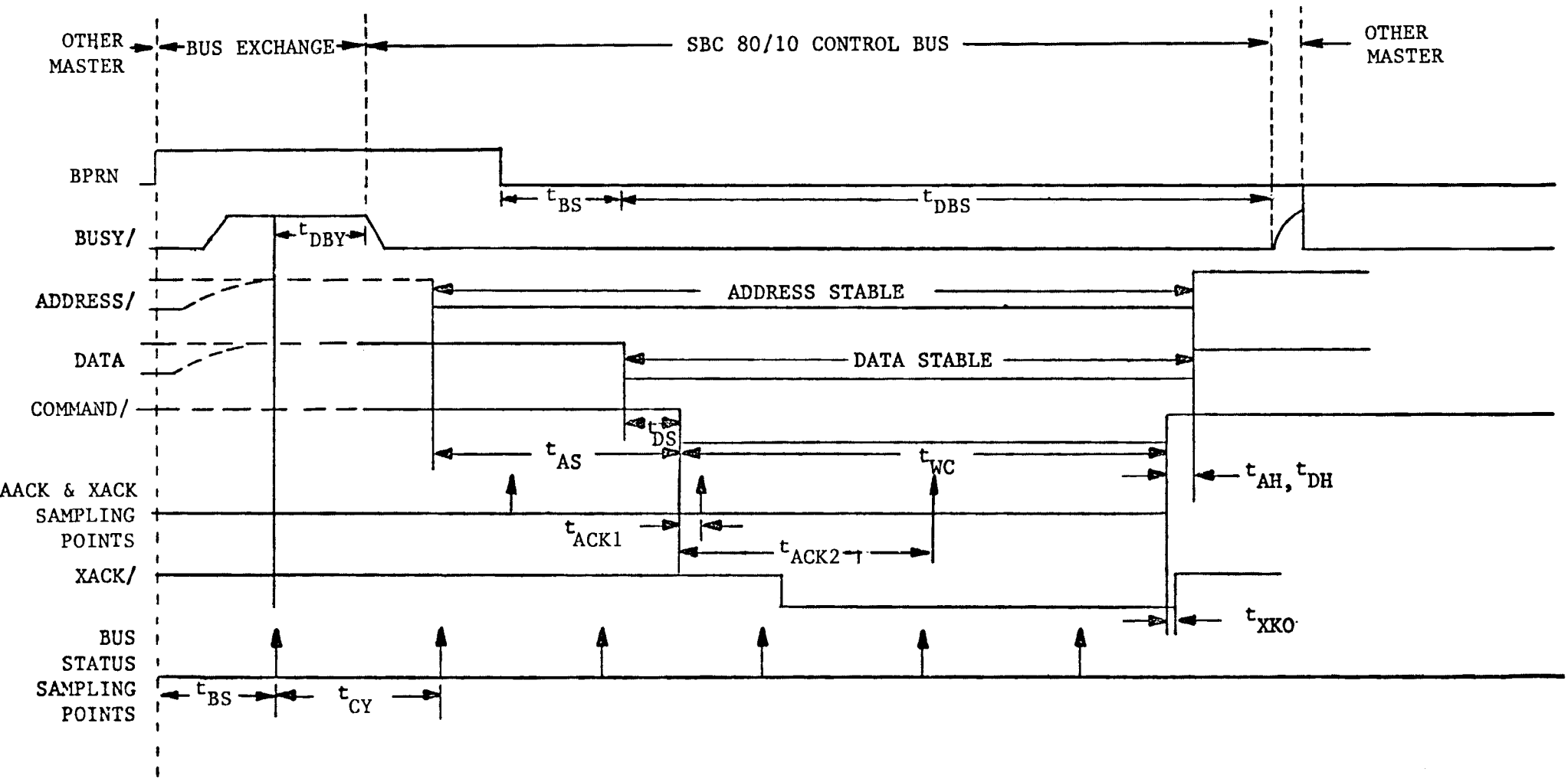


*FIGURE A-3. BUS EXCHANGE (WRITE)

*NOT DRAWN TO SCALE.

A.3 POWER SUPPLY SPECIFICATIONS

ELECTRICAL SPECIFICATIONS

Input: 100/115/215/230 VAC ±10%, 47-63 hz, single phase
(Input voltage is selectable by transformer jumpers.)

Output:

TABLE A-7

NOMINAL DC VOLTAGE				
Nominal DC Voltage	+5V	-5V	+12V	-12V
Current (Amp)	14.0	0.9	2.0	0.8

Protection Circuits:

TABLE A-8

CURRENT LIMIT AND OVP PROTECTION					
Nominal DC Voltage	+5V	-5V	+12V	-12V	Comment
Current Limit (Amp)	16.8	1.1	2.4	1.0	≈ 120% of Rated I
OVP Range (Volt)	5.8 to 6.6	-5.8 to -6.6	14 to 16	-14 to -16	

DC output voltage adjustment range: ±5% of nominal voltage (all outputs)

Line Regulation: ±0.1% for 10% line change (all outputs)

Load Regulation: ±0.1% for 50% load change (all outputs)

Ripple and Noise: 10 mv peak-to-peak maximum for DC to 500 Khz (all outputs)

Transient Response: Less than 50 μsec for 50% load change (all outputs)

Remote Sensing: Provided on +5V output only, regulate at load.

Stability: ±0.05% for 8 hours after 30 minutes of warm up (all outputs)

AC power low detection circuitry: TTL high level signal to indicate AC power line is below 105/207 VAC (RMS).

TABLE A-9

TTL "AC LOW" SIGNAL		
PARAMETER	MINIMUM	MAXIMUM
V_{OL}	0.V	0.4V
V_{OH}	2.4V	5.25V
I_{OL} at 0.4V	16.0mA	-
I_{OH} at 3.4V	0.4mA	-

After "AC Low" signal is active, all DC voltages will conform to specification for a minimum of 7.5 msec at any frequency within the input operating frequency range (47-63 hz).

Input and Output Connectors:

Input AC: 4 pin keyed connector.

TABLE A-10

AC INPUT		
P2 CONNECTORS		
PIN	TO	WIRE COLOR
1	T1-1	GRAY
2	T1-2	RED
3	T1-3	ORN
4	T1-4	WHT

Output DC: 7 pin keyed connector, two sets.

TABLE A-11

DC OUTPUT				
	P6		P8	
PIN	OUTPUT	WIRE COLOR	OUTPUT	WIRE COLOR
1	GND	BLK	KEY	-
2	+5V	RED	GND	BLK
3	+5V	RED	-5V	YEL
4	-12V	PURPLE	+12V	BLUE
5	-	-	+5V	RED
6	GND	BLK	+5V	RED
7	KEY	-	GND	BLK

Mechanical Specifications:

Dimension: 12.65" maximum width x 3.19" maximum height x 6.03" maximum depth

Weight: 13 pounds maximum

Finish: Natural aluminum with clear or black anodized plating.

Material: 1/8" thick aluminum

Environmental Specifications:

Operating temperature range: $0^{o}C$ to $55^{o}C$, with 30 CFM of moving air

Temperature Coefficient: $\pm$0.02% per ^{o}C maximum

Humidity: 90% maximum relative humidity with no condensation.

A.4 SBC 604 MODULAR CARDCAGE AND BACKPLANE ASSEMBLY SPECIFICATIONS

BACKPLANE CHARACTERISTICS

All SBC 80 address, data, and command bus lines are bussed to all four connectors on the Printed Circuit Backplane.

Power connectors for ground, +5, -5, +12, -12, -10 volt power supply lines.

Bus signal terminators and backplane male edge expansion connector.

CONNECTORS

4 86-pin PC edge connectors with 0.156 inch contact centers.

2 7-pin right angle wafer connectors with 0.156 inch contact centers.

PHYSICAL DIMENSIONS

Height:	8.5 in.	(21.59 cm)
Width:	14.2 in.	(36.07 cm)
Depth:	3.34 in.	(8.48 cm)
Weight:	35 oz.	(992.23 gm)

ENVIRONMENTAL

Operating Temperature

0°C to 55°C

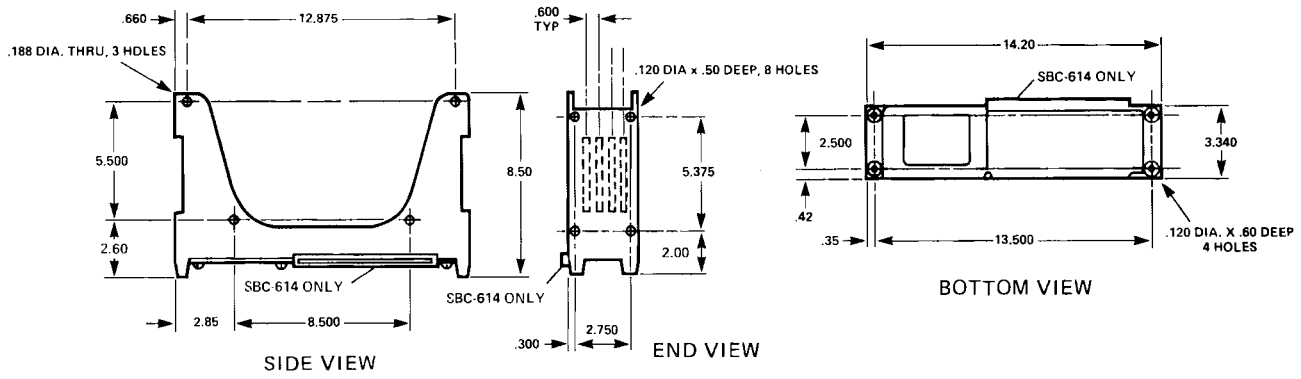
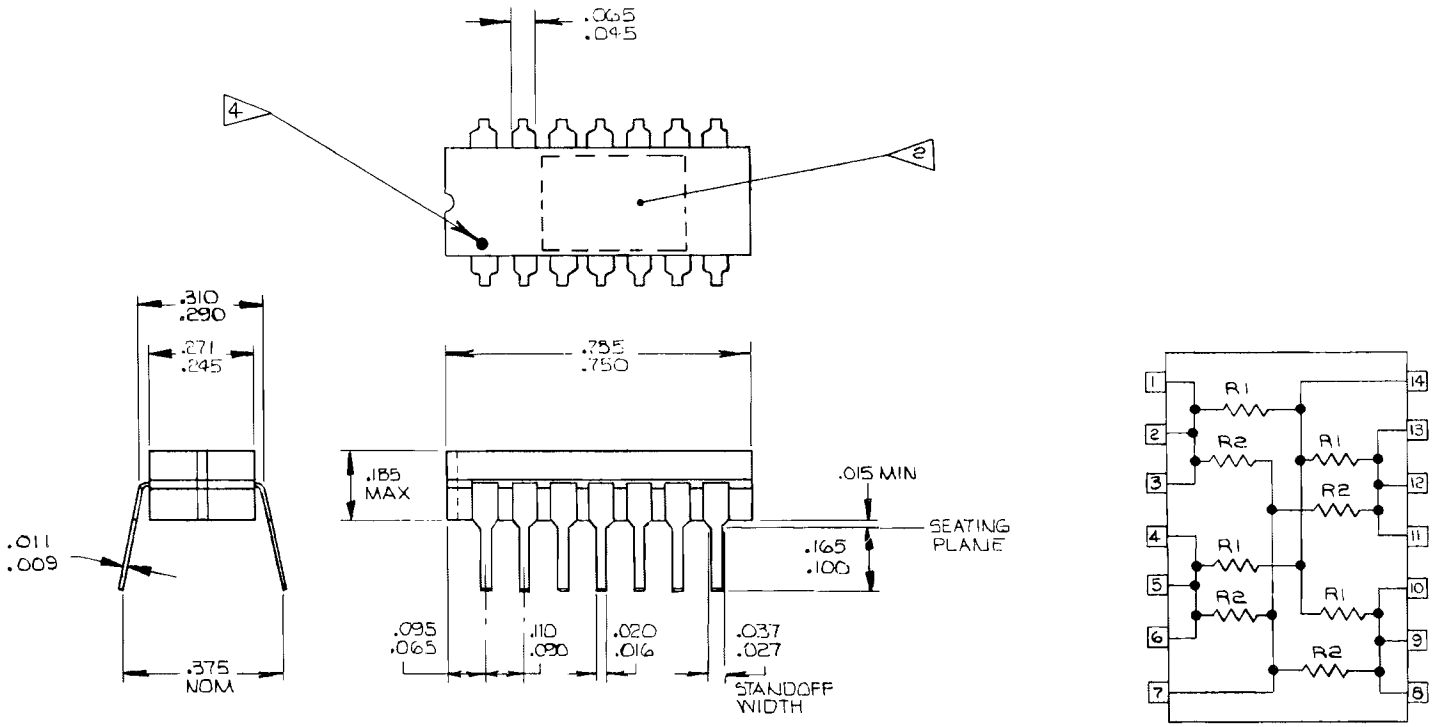


FIGURE A-4. SBC 604 DIMENSIONS

A.5 SBC 901/902 TERMINATION RESISTOR PACKS

The SBC 901 and SBC 902 terminators are both compatible with the I/O driver sockets on the System 80/10 and all the memory and I/O expansion boards. The SBC 901 is a 220Ω/330Ω voltage divider terminator and the SBC 902 is a 1K pull up terminator. See Figure A-5 and A-6 for schematics.



RESISTOR NETWORK SPECIFICATIONS:

RESISTANCE VALUES:
±2% (MAX)

OPERATING TEMPERATURE:
0°C TO +70°C

TEMPERATURE COEFFICIENT:
±200 PPM/°C OVER TEMPERATURE RANGE OF 0°C TO +70°C

OPERATING VOLTAGE:
6.0 VDC (MAX)

POWER RATING:
AT 70°C, 0.7 WATT PER PACK

TRACKING RESISTANCE RATIO:
±1.0% (MAX)

STABILITY:
±1% YEAR (MAX)

LOAD LIFE:
±1% (ΔR) OVER 1000 HOURS

PACKAGE:
DUAL IN LINE - CERAMIC OR PLASTIC

NOTES:
UNLESS OTHERWISE SPECIFIED,

1. PART NO IS 4500644-01.

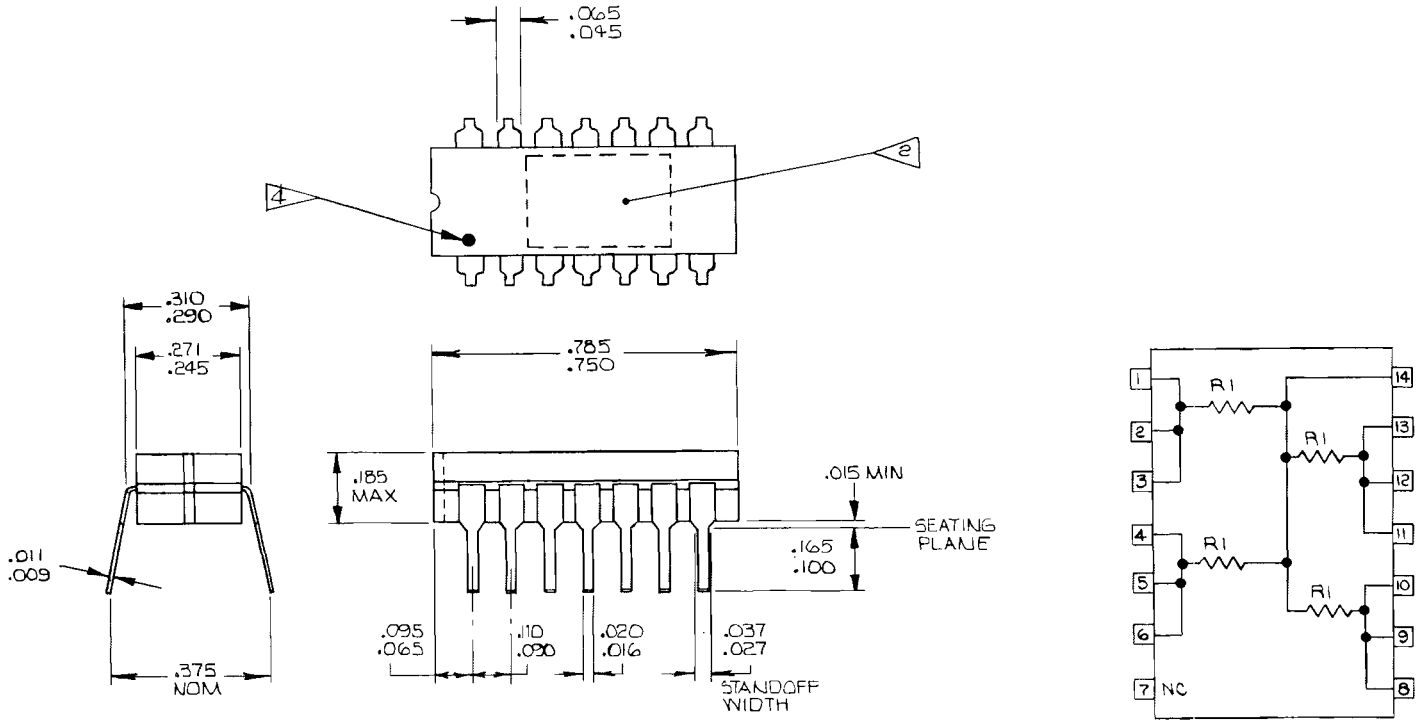
2. INK STAMP PRODUCT CODE, RESISTOR VALUE, PART NO, AND DASH NUMBER WITH CONTRASTING COLOR INK USING MIN .05 HIGH CHARACTERS. NO OTHER MARKINGS PERMITTED EXCEPT MANUF BATCH NO.

E.G.) SBC-901
R220/330
4500644-01

3. FOR PROCUREMENT SEE LV 4500644-01.

4. IDENTIFY PIN ONE CLEARLY ON TOP OF PACKAGE.

FIGURE A-5. SBC 901 TERMINATOR SCHEMATIC



RESISTOR NETWORK SPECIFICATIONS:

RESISTANCE VALUES:
±2% (MAX)

OPERATING TEMPERATURE:
0°C TO +70°C

TEMPERATURE COEFFICIENT:
±200 PPM/°C OVER TEMPERATURE RANGE OF 0°C TO +70°C

OPERATING VOLTAGE:
6.0 VDC (MAX)

POWER RATING:
AT 70°C, 0.7 WATT PER PACK

TRACKING RESISTANCE RATIO:
±1.0% (MAX)

STABILITY:
±1% YEAR (MAX)

LOAD LIFE:
±1% (ΔR) OVER 1000 HOURS

PACKAGE:
DUAL IN LINE - CERAMIC OR PLASTIC

NOTES:
UNLESS OTHERWISE SPECIFIED,

1. PART NO IS 4500645-01.

2. INK STAMP PRODUCT CODE, RESISTOR VALUE, PART NO AND DASH NUMBER WITH CONTRASTING COLOR AND MIN .12 HIGH CHARACTERS. NO OTHER MARKINGS ARE PERMITTED EXCEPT FOR MANUF BATCH NO.

E.G.) SBC-902
R 1K
4500645-01

3. FOR PROCUREMENT SEE LV4500645

4. IDENTIFY PIN ONE CLEARLY ON TOP OF PACKAGE.

FIGURE A-6. SBC 902 TERMINATOR SCHEMATIC

APPENDIX B
SYSTEM 80/10 SCHEMATICS

Schematic drawings for the System 80/10 are provided in this appendix. Information and diagrams in this section are subject to change without notice. References should be made to schematics shipped with this system.

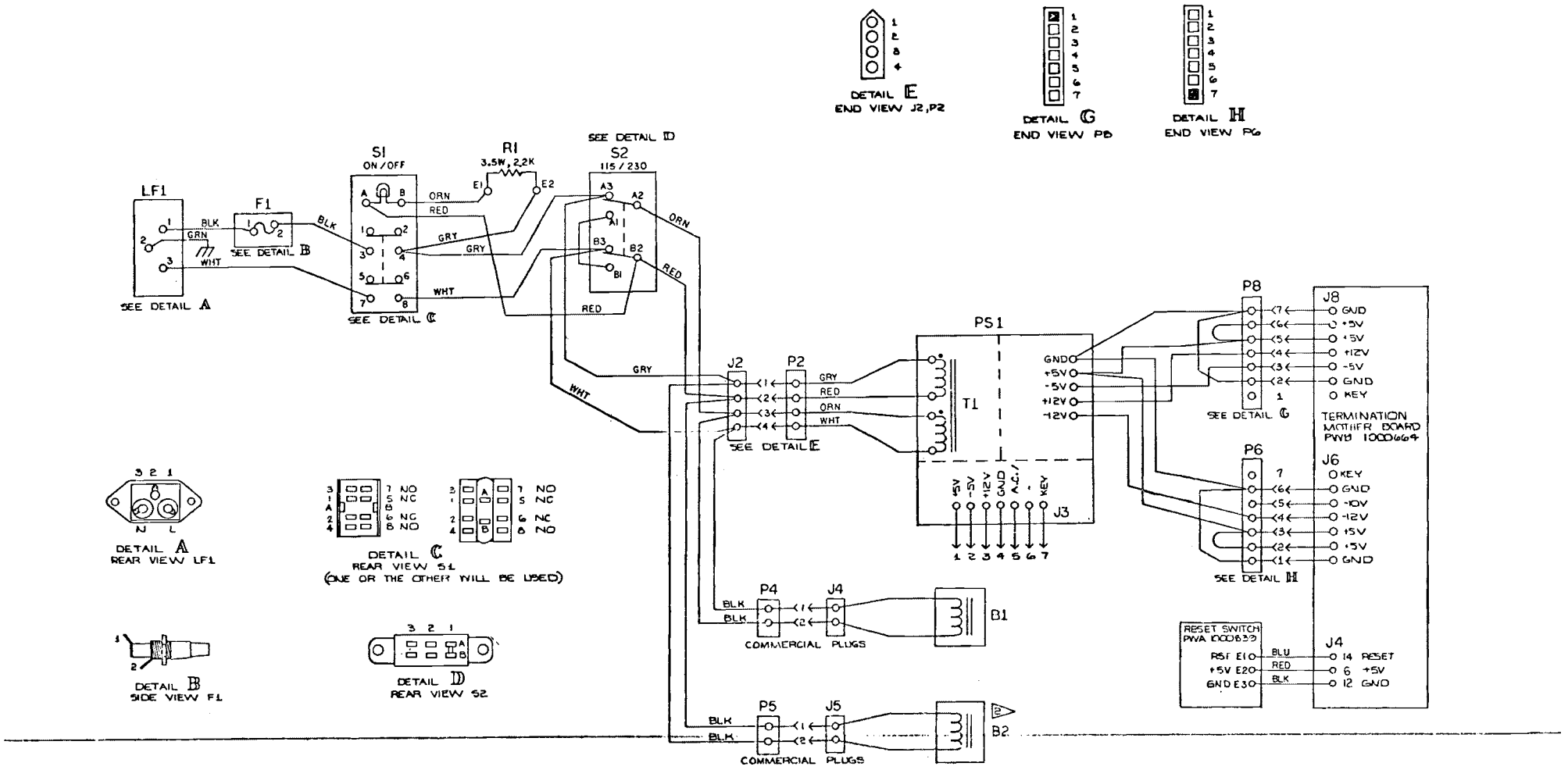
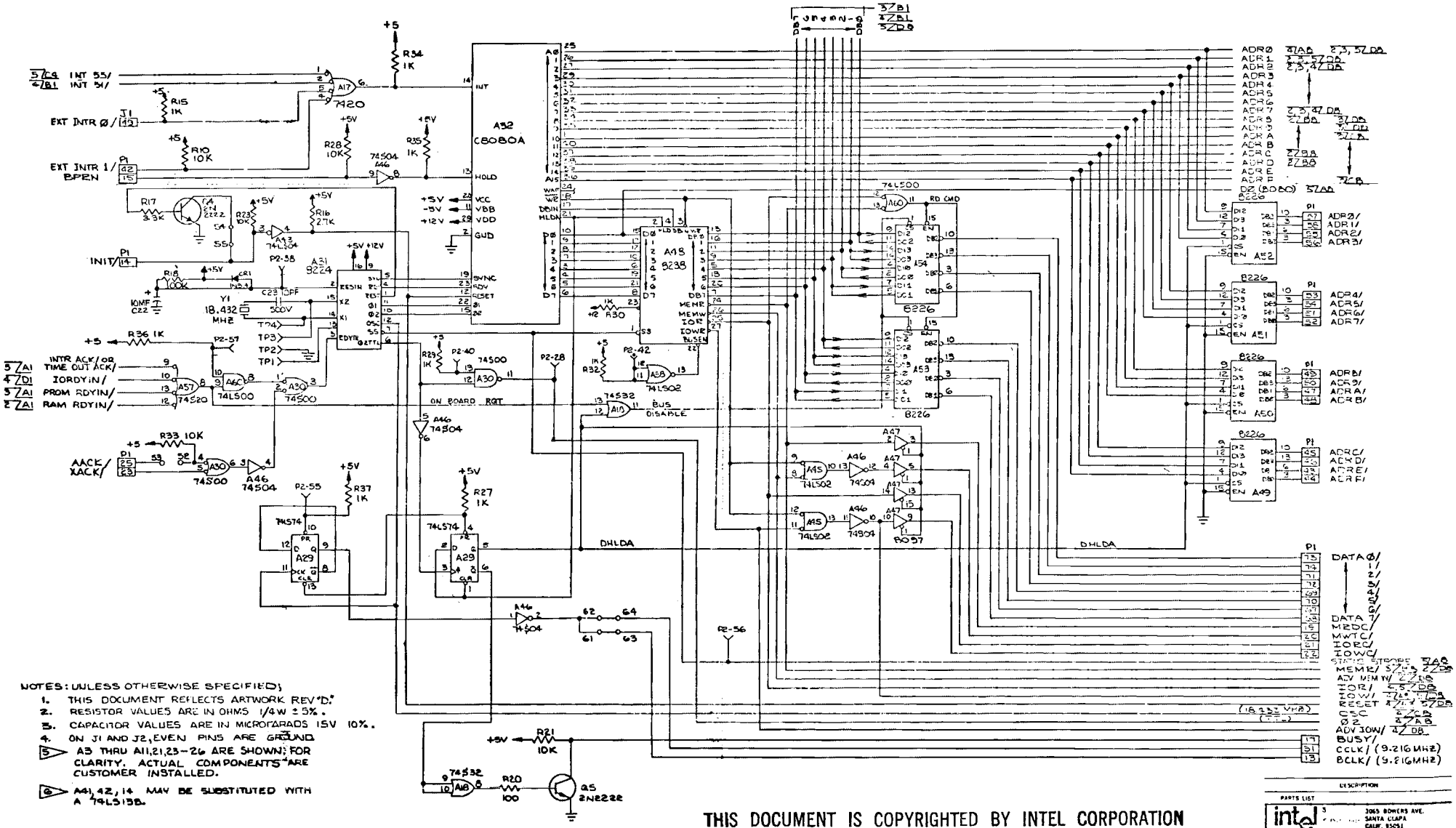


FIGURE B-1. AC/DC POWER DISTRIBUTION DIAGRAM

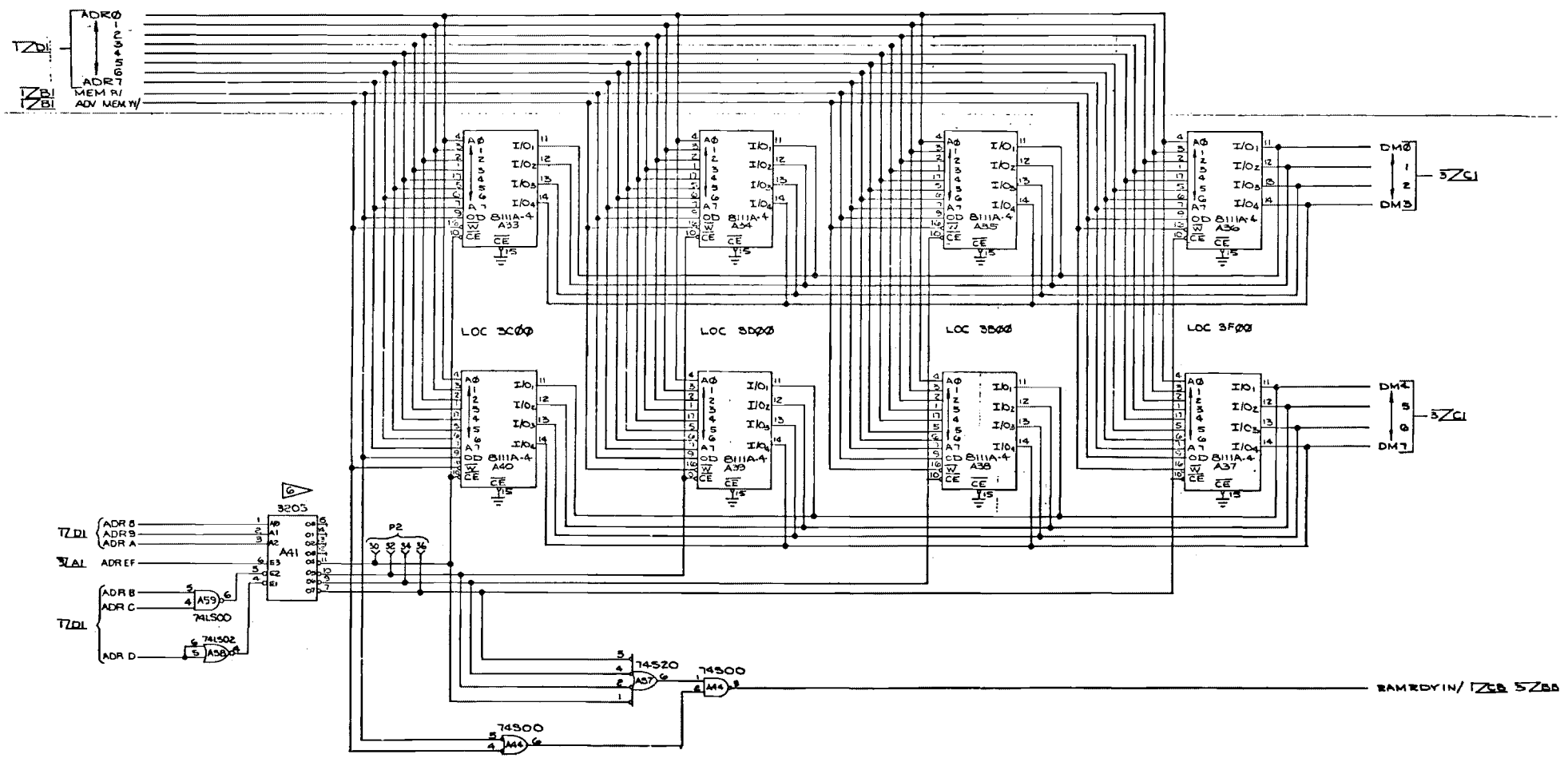
DESCRIPTION			
PARTS LIST			
intel	3065 BOWERS AVE. SANTA CLARA CALIF. 95051		
TITLE	WIRE DIAGRAM A.C./D.C. POWER		
SIZE	DEPT	DRAWING NO.	REV
D	410	2000766	A



NOTES: (UNLESS OTHERWISE SPECIFIED)
1. THIS DOCUMENT REFLECTS ARTWORK REV "D".
2. RESISTOR VALUES ARE IN OHMS 1/4W ±5%.
3. CAPACITOR VALUES ARE IN MICROFARADS 15V 10%.
4. ON J1 AND J2, EVEN PINS ARE GROUND.
5. A3 THRU A11, 21, 23–26 ARE SHOWN FOR CLARITY. ACTUAL COMPONENTS ARE CUSTOMER INSTALLED.
6. A41, 42, 14 MAY BE SUBSTITUTED WITH A 74LS138.

FIGURE B-2. SBC 80/10 SCHEMATIC (1 of 5)

intel	3065 BOWERS AVE. SANTA CLARA CALIF. 95051
TITLE	SCHEMATIC, SINGLE BOARD COMPUTER 80/10
SIZE	D
DEPT	410
DRAWING NO.	2000620
REV	G



SIZE D | DRAWING NO. 2000620 | REV G | SHEET 2 OF 5

FIGURE B-2. SBC 80/10 SCHEMATIC (2 of 5)

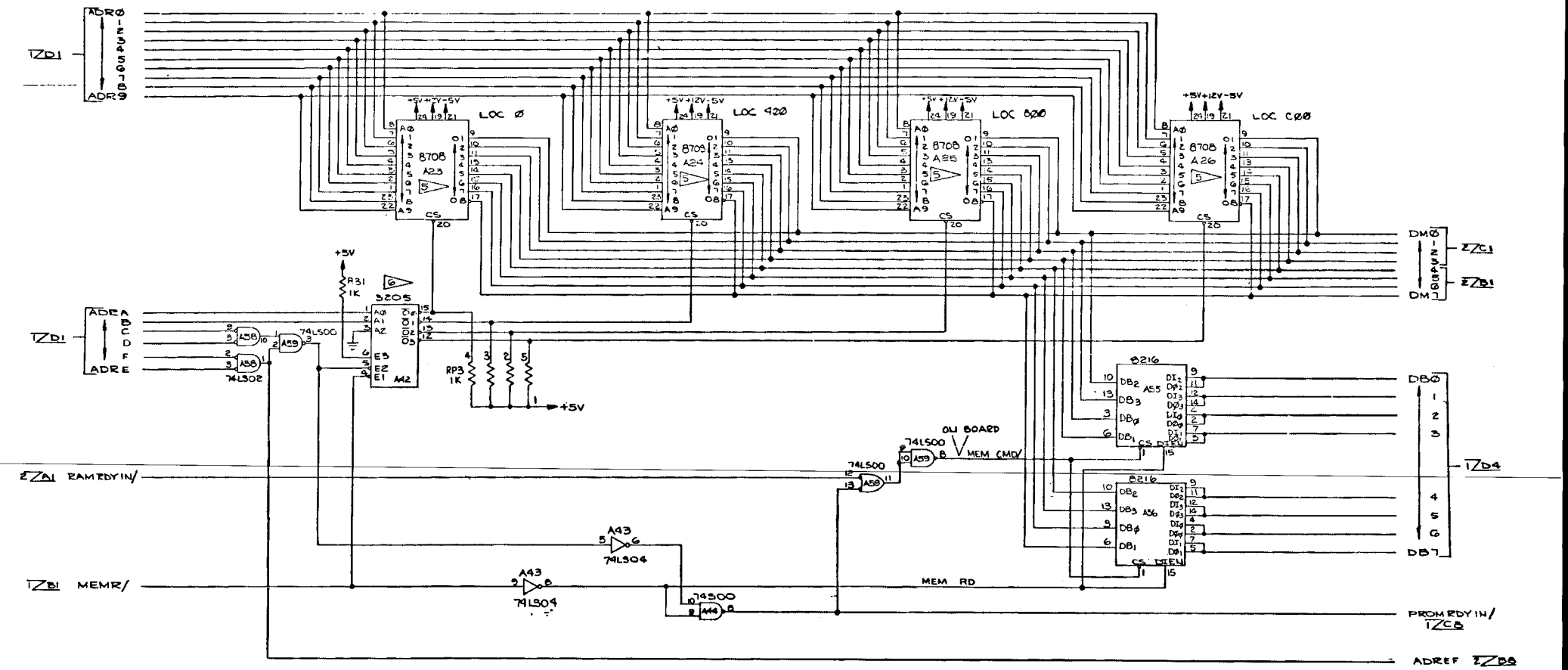
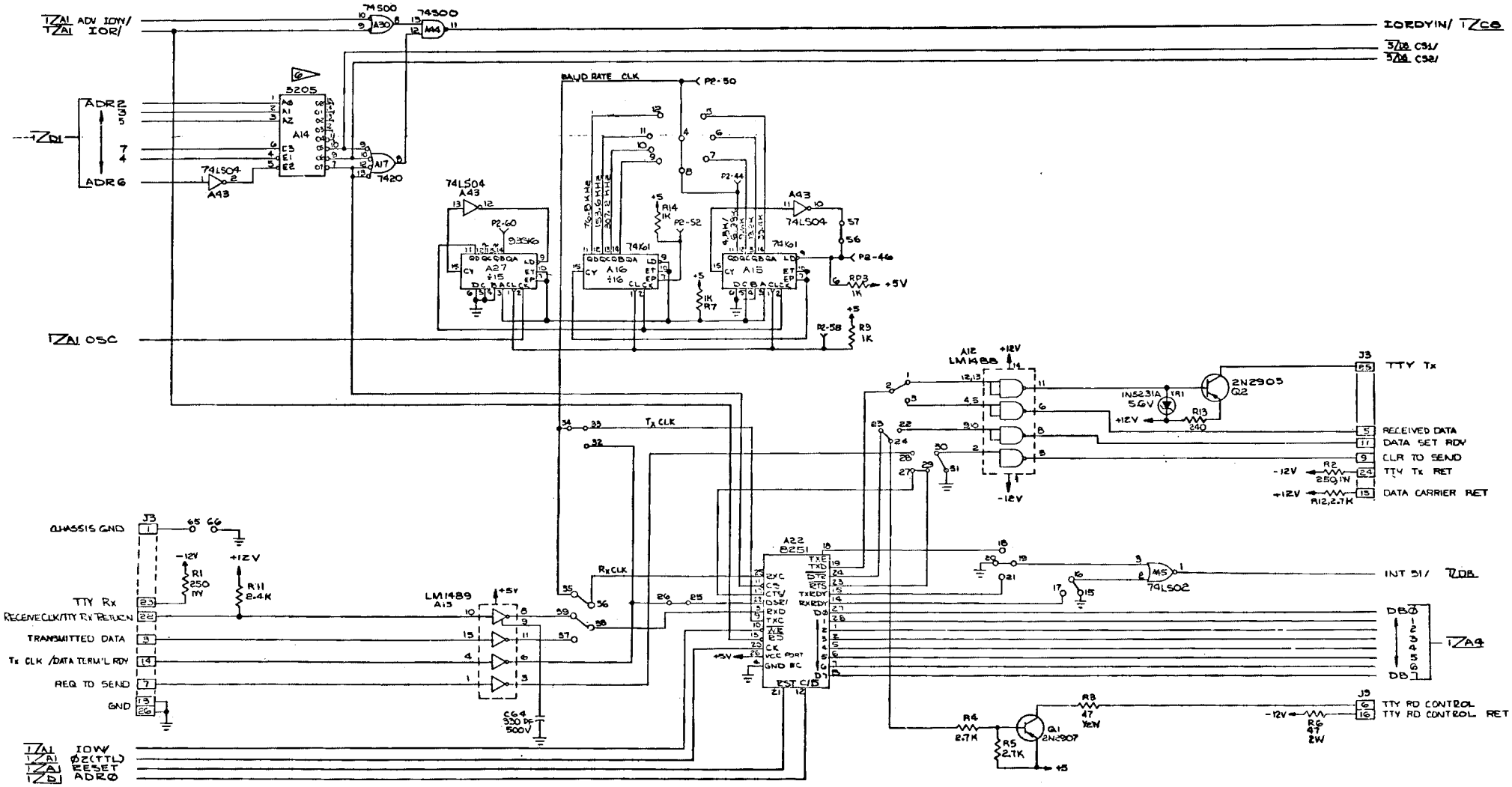


FIGURE B-2. SBC 80/10 SCHEMATIC (3 of 5)

SCALE:	SIZE	DEPT	DRAWING NO.	REV
SHEET 3 OF 5	D	410	2000620	G



SCALE: SHEET 4 OF 5 | SIZE D | DEPT 410 | DRAWING NO. 2000620 | REV 16

FIGURE B-2. SBC 80/10 SCHEMATIC (4 of 5)

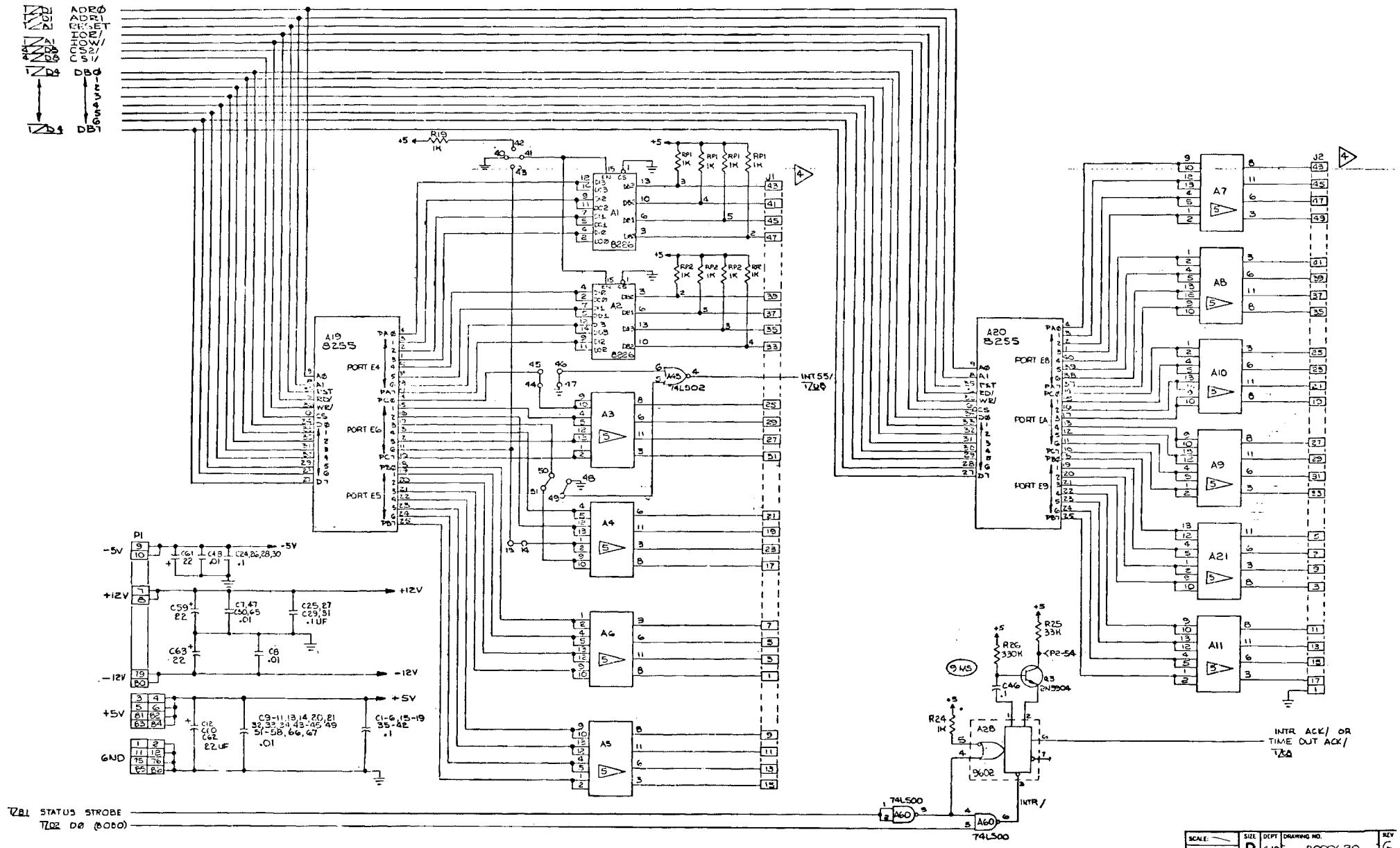
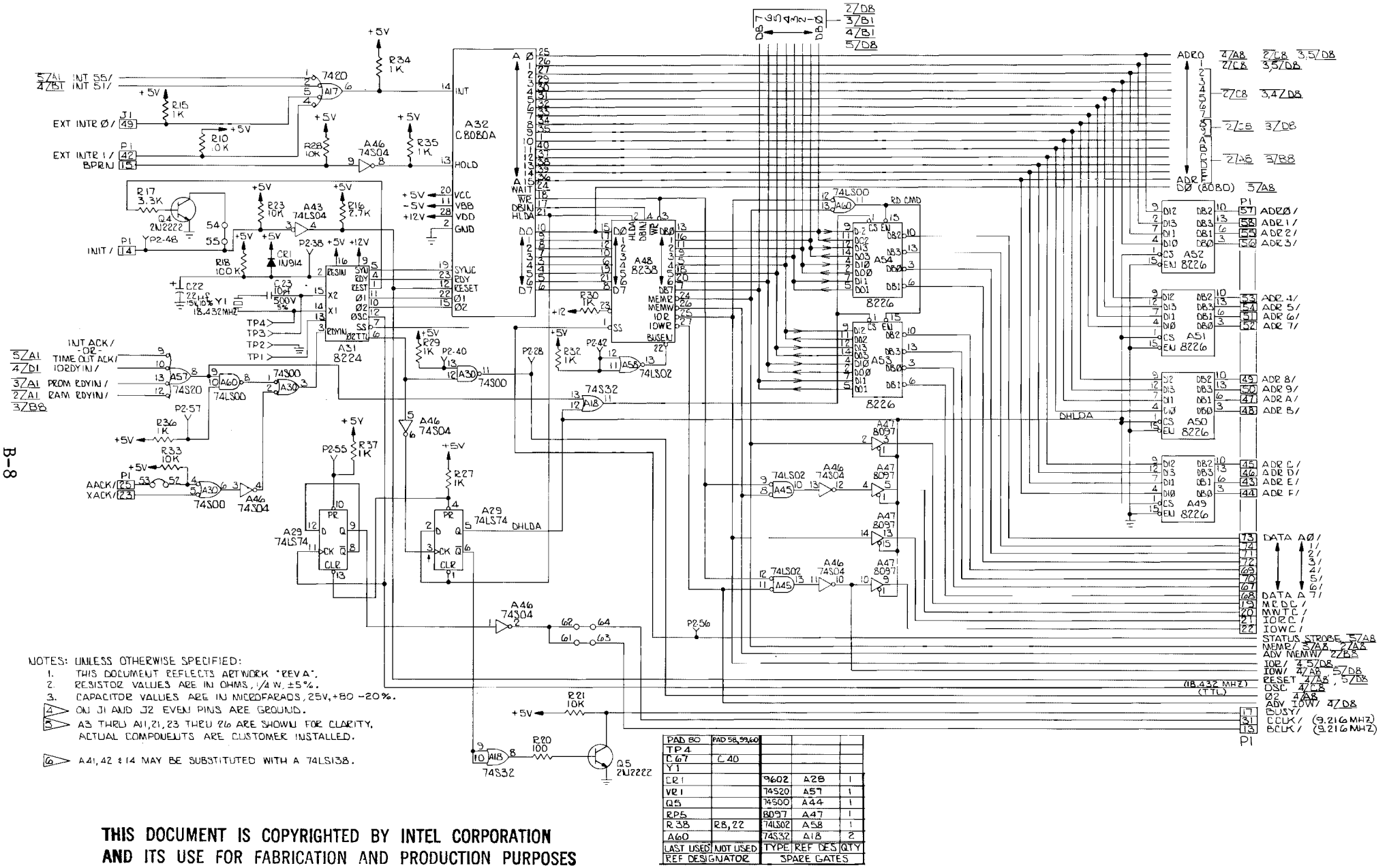


FIGURE B-2. SBC 80/10 SCHEMATIC (5 of 5)



PAD 80	PAD 58, 59, 60			
TP4				
C67	C40			
Y1				
CR1		9602	A28	1
VR1		74S20	A57	1
Q5		74S00	A44	1
RP5		8097	A47	1
R38	R8, 22	74LS02	A58	1
A60		74S32	A18	2
LAST USED	NOT USED	TYPE	REF DES	QTY
REF DESIGNATOR		SPARE GATES		

FIGURE B-3. SBC 80/10A SCHEMATIC (SHEET 1 of 5)

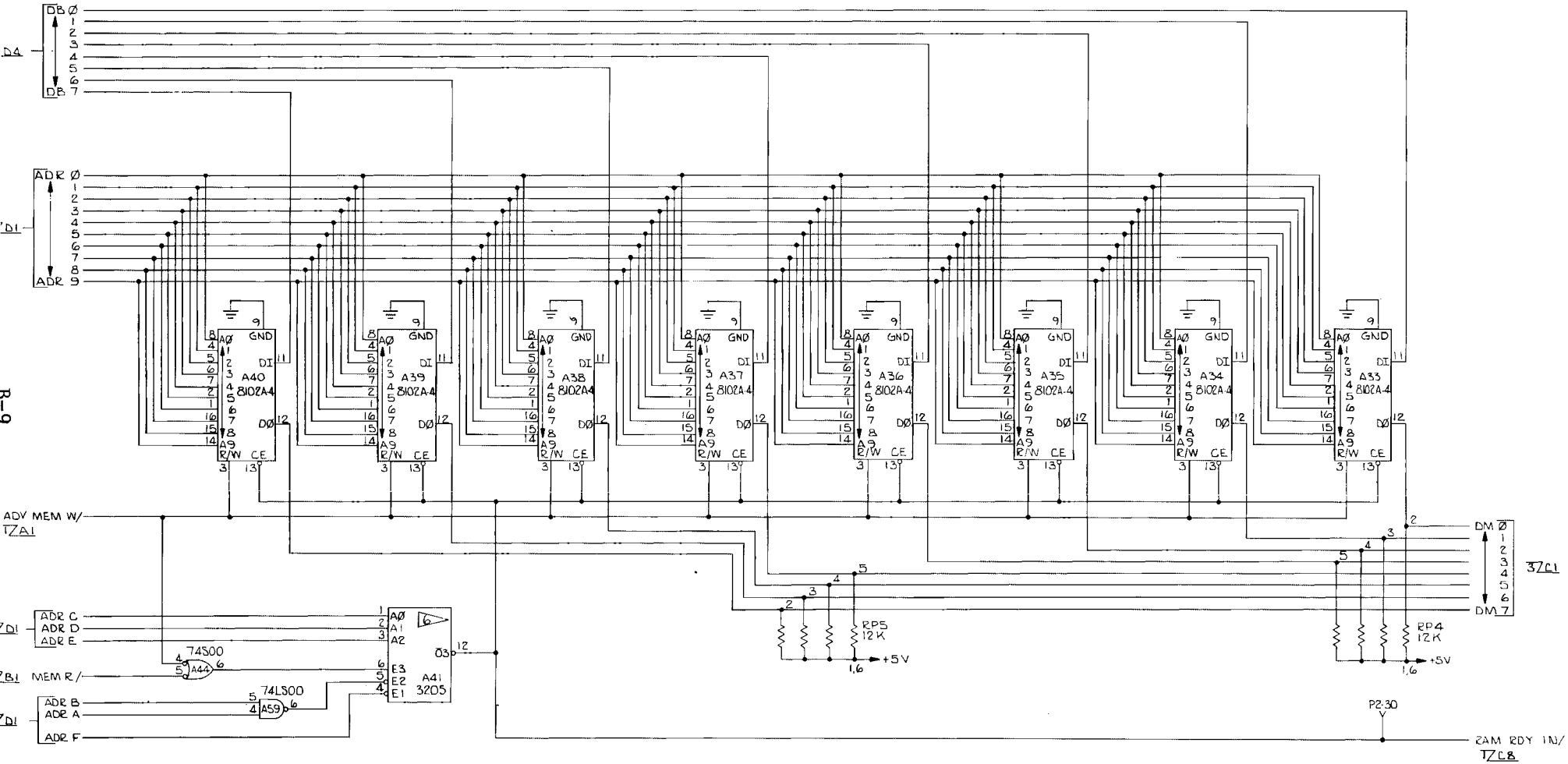
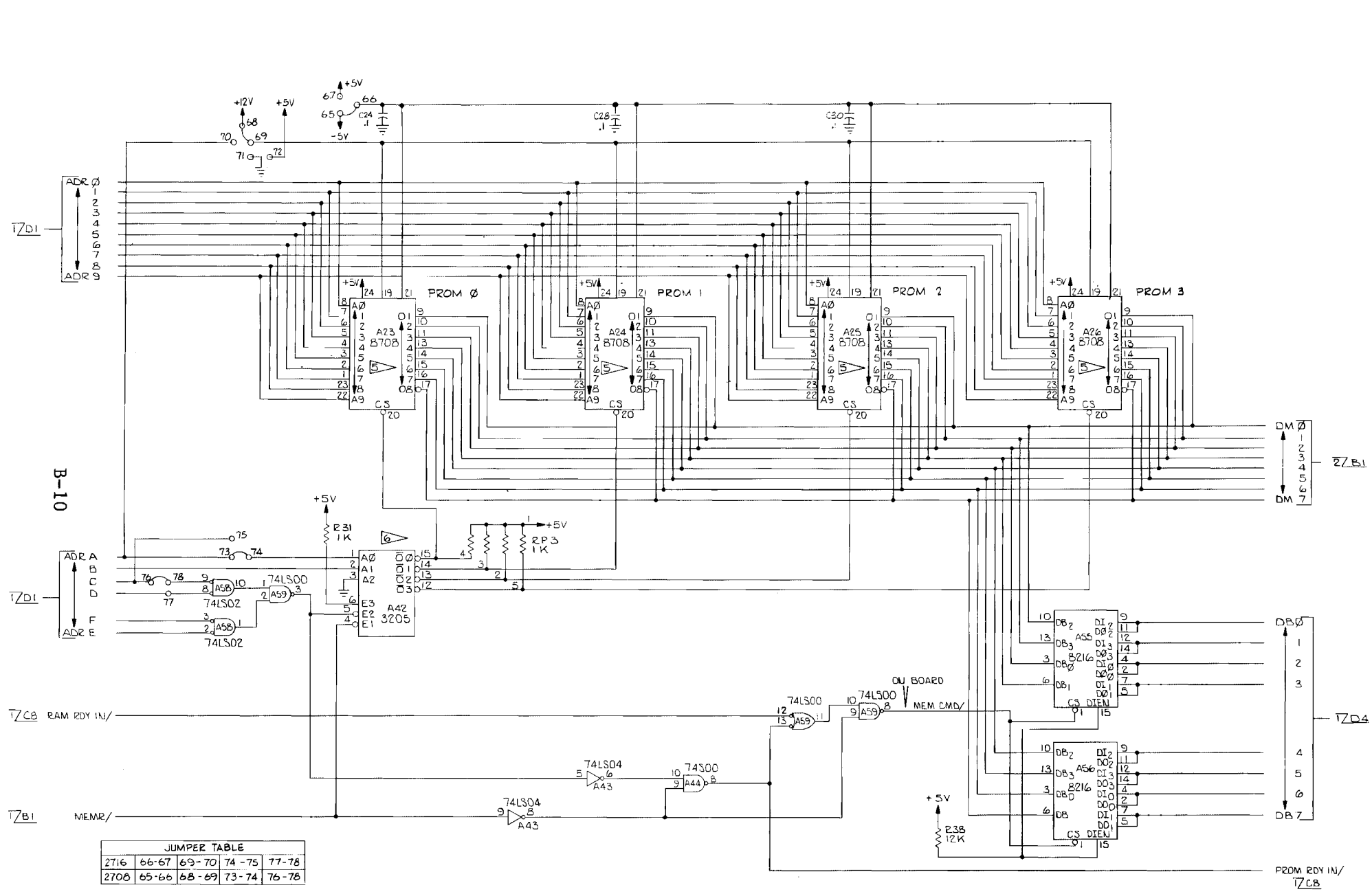


FIGURE B-3. SBC 80/10A SCHEMATIC (SHEET 2 of 5)



JUMPER TABLE				
2716	66-67	69-70	74-75	77-78
2708	65-66	68-69	73-74	76-78

	A23	A24	A25	A26
2716	0-7FF	1000-17FF	800-FFF	1800-1FFF
2708	0-3FF	400-7FF	800-BFF	C00-FFF

FIGURE B-3. SBC 80/10A SCHEMATIC (SHEET 3 of 5)

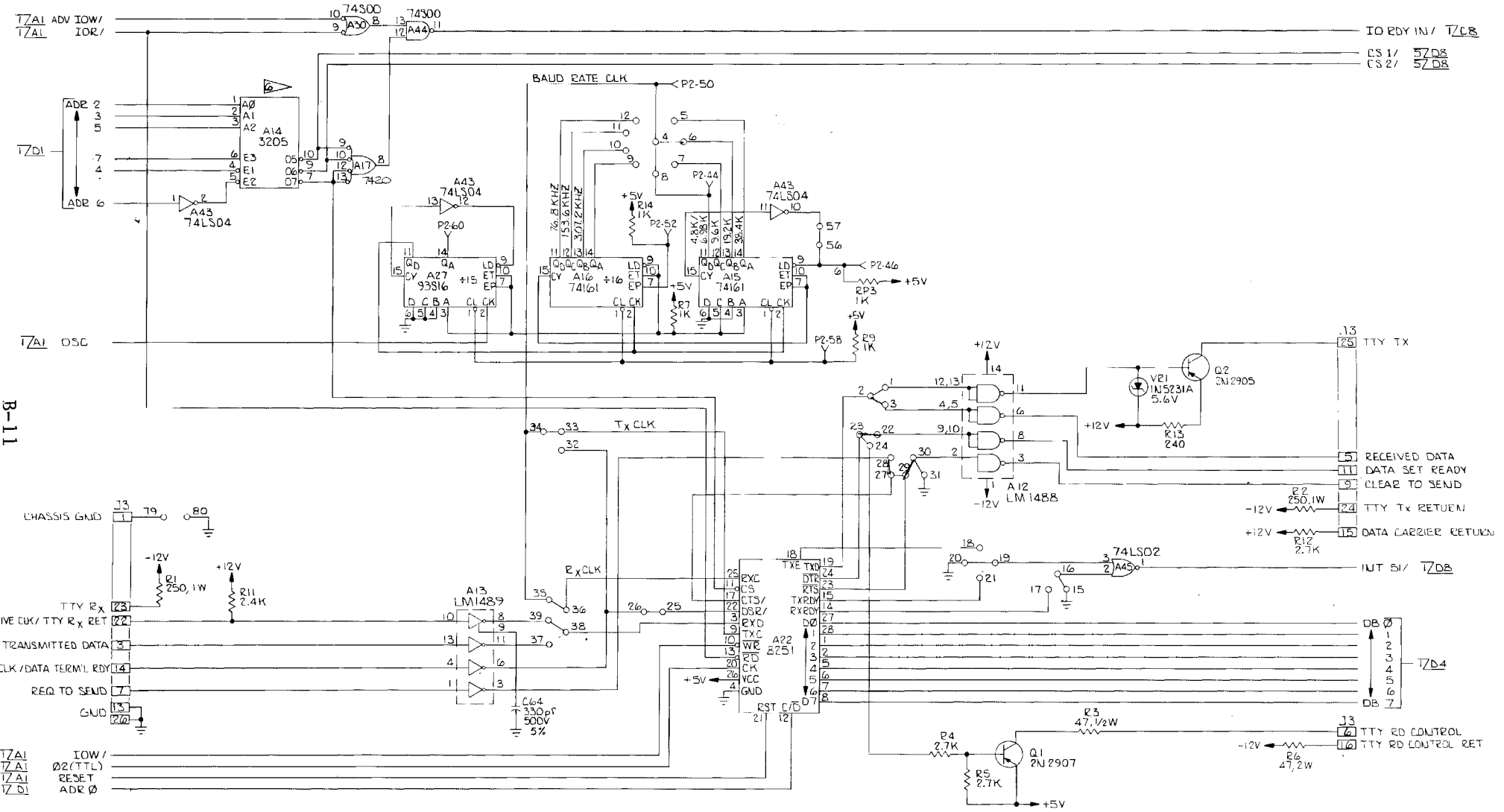


FIGURE B-3. SBC 80/10A SCHEMATIC (SHEET 4 of 5)

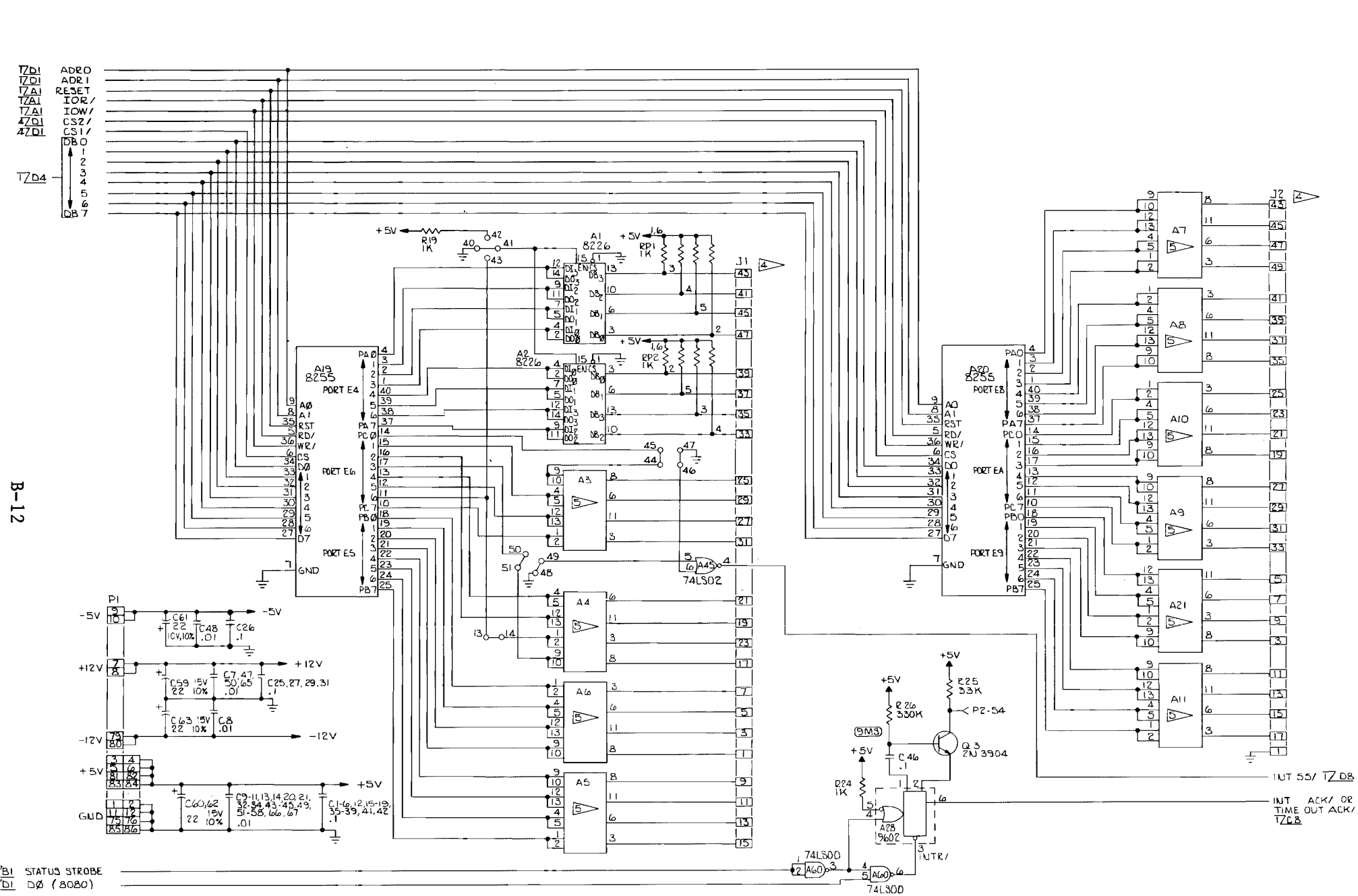
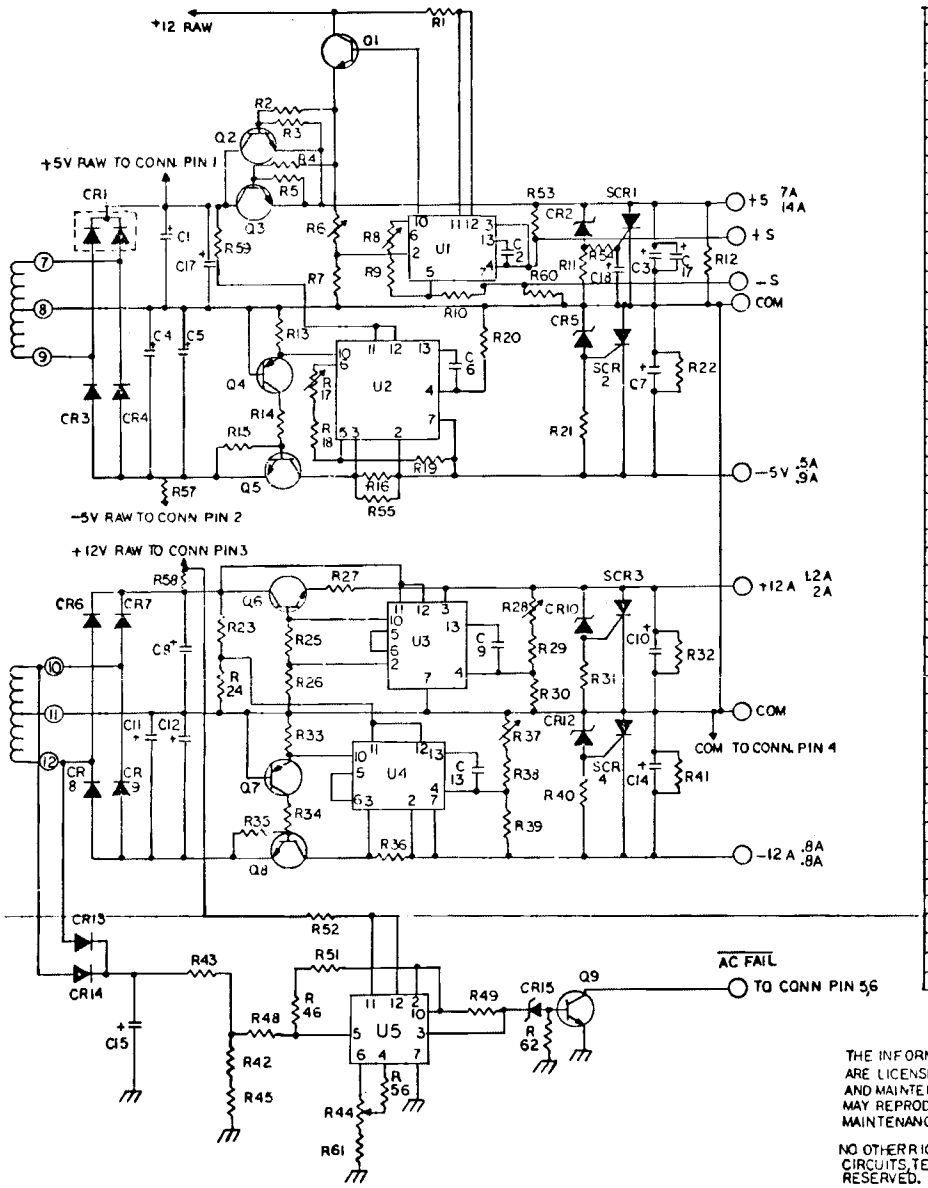


FIGURE B-3. SBC 80/10A SCHEMATIC (SHEET 5 of 5)



REF DES	4500770	DESCRIPTION
C1	64000/15	CAPACITOR, ELECT.
C3,4,5,10,14,17	1000/16	
C7	220/16	
C8	3600/35	
C2,13	.001/100	MYLAR
C11,12	330/35	ELECT
C18	10/25	
C17	64000/15	
C9,16	500 PF	MYLAR
C6	.003/100	CAPACITOR, MYLAR
C15	1/50	CAPACITOR, ELECT
CR1	R731	DIODE BRIDGE
CR3,4,5,9,13,14	AE1C	1A 200V
CR6,7	AE38	3A 100V
CR2,5,15	IN752A	ZENER
CR10,12	IN965A	DIODE, ZENER
C2,13	.001/100	CAPACITOR, MYLAR
SCR1	S0715LS3	8A SCR
SCR 2,3,4	S0303LS3	3A SCR
R1, 11	82Ω	RESISTOR, ½W, 5%, CF
R7	22K	
R35	22Ω	
R46, 51	51K	
R15, 35	36K	
R12,21,22,14,50	82Ω	
R13,40,23,25,31	330	
R53,60,2,4,42,54	6.8Ω	
R16,57	1.0Ω	
R24,32,41,20,49	1K	
R55	.5Ω	
R33,34,52	330Ω	
R43	20K	½ W 5% CF
R27	.22Ω	2W WW BWH
R36, 58	.56Ω	2W WW BWH
R62	10K	¼ W 2% MF
R10,19,45,48	4.75K	
R30,39	2K	
R9,18,29,38,47,6,5,8	1.2K	RESISTOR, ¼ W, 2% MF
R26	4.7K	RESISTOR, ½ W, 5% CF
R8,6,17,44,37,28	15K	POT, CTS 1/E54719
Q1,8,5	12500-5	TRANSISTOR NPN POWER
Q2,3,6	12505-1	NPN POWER
Q4,7	2N2907	PNP SIG.
Q9	2N2219	TRANSISTOR NPN SIG.
U1,2,3,4,5	UA 723	IC VOLTAGE REGULATOR
T1	13236	TRANSFORMER
CHASSIS	13171	CHASSIS
PCB	13192	PRINTED CIRCUIT BOARD

DESCRIPTION

PARTS LIST

intel 3065 BOWERS AVE. SANTA CLARA CALIF. 95051

TITLE SCHEMATIC POWER SUPPLY V/C AA

SIZE	DEPT	DRAWING NO.	REV
D	410	2000952	A

FIGURE B-4. POWER SUPPLY SCHEMATIC (V/C AA)

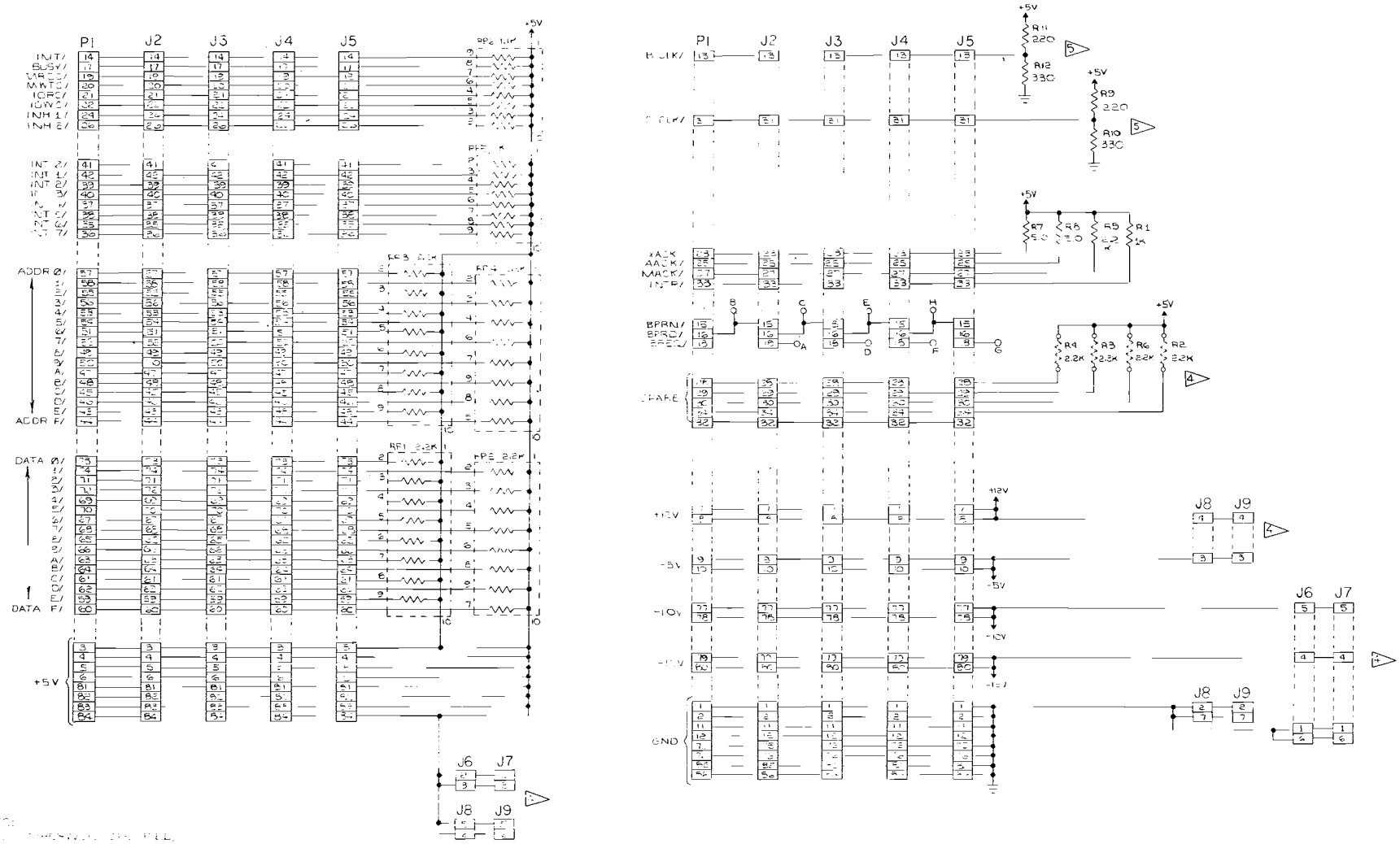
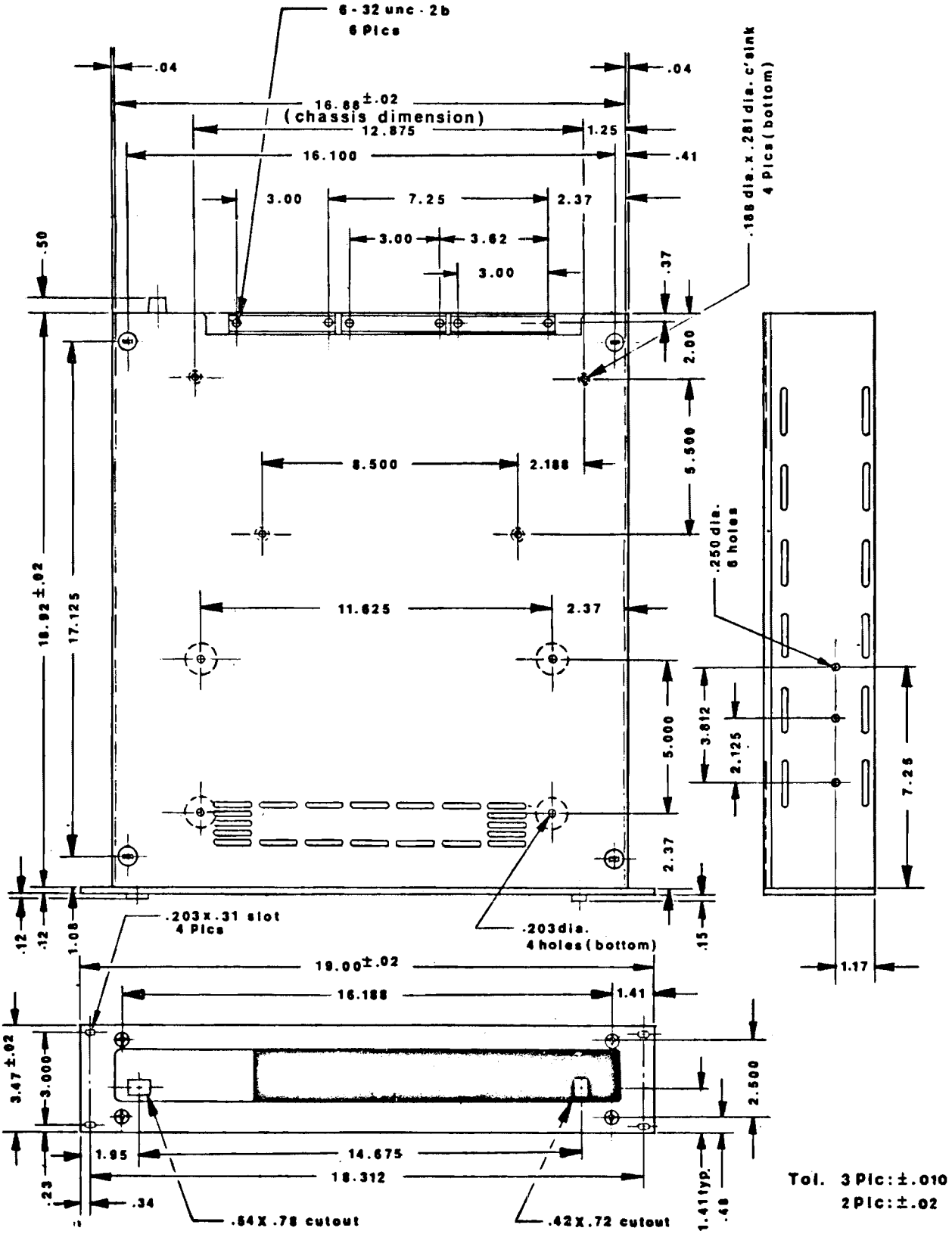


FIGURE B-5. TERMINATION BACKPLANE SCHEMATIC

APPENDIX C

SYSTEM AND SUB-ASSEMBLY OUTLINES

C.1 SYSTEM 80/10 OUTLINE



system 80/10 outline
98-318 A

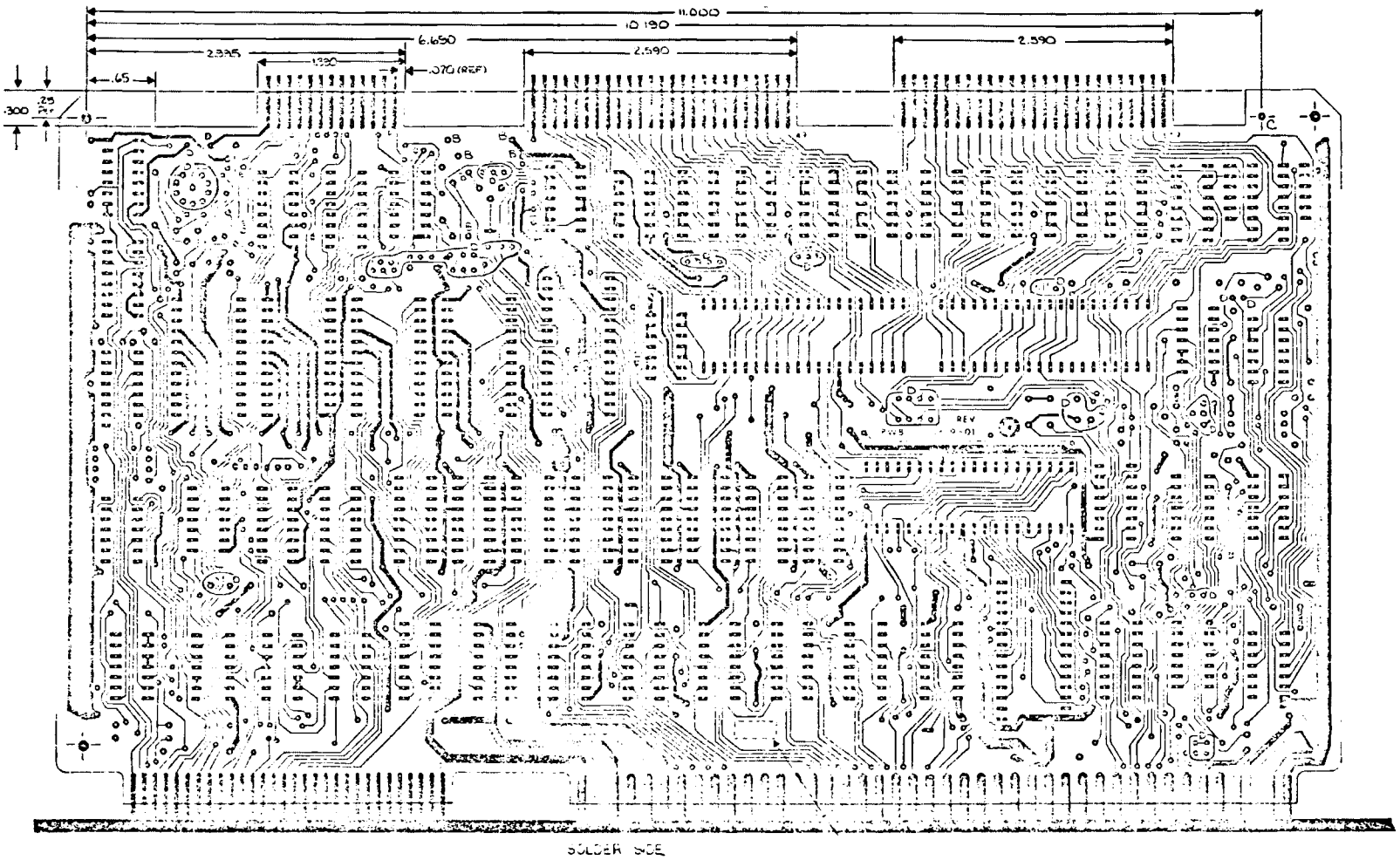
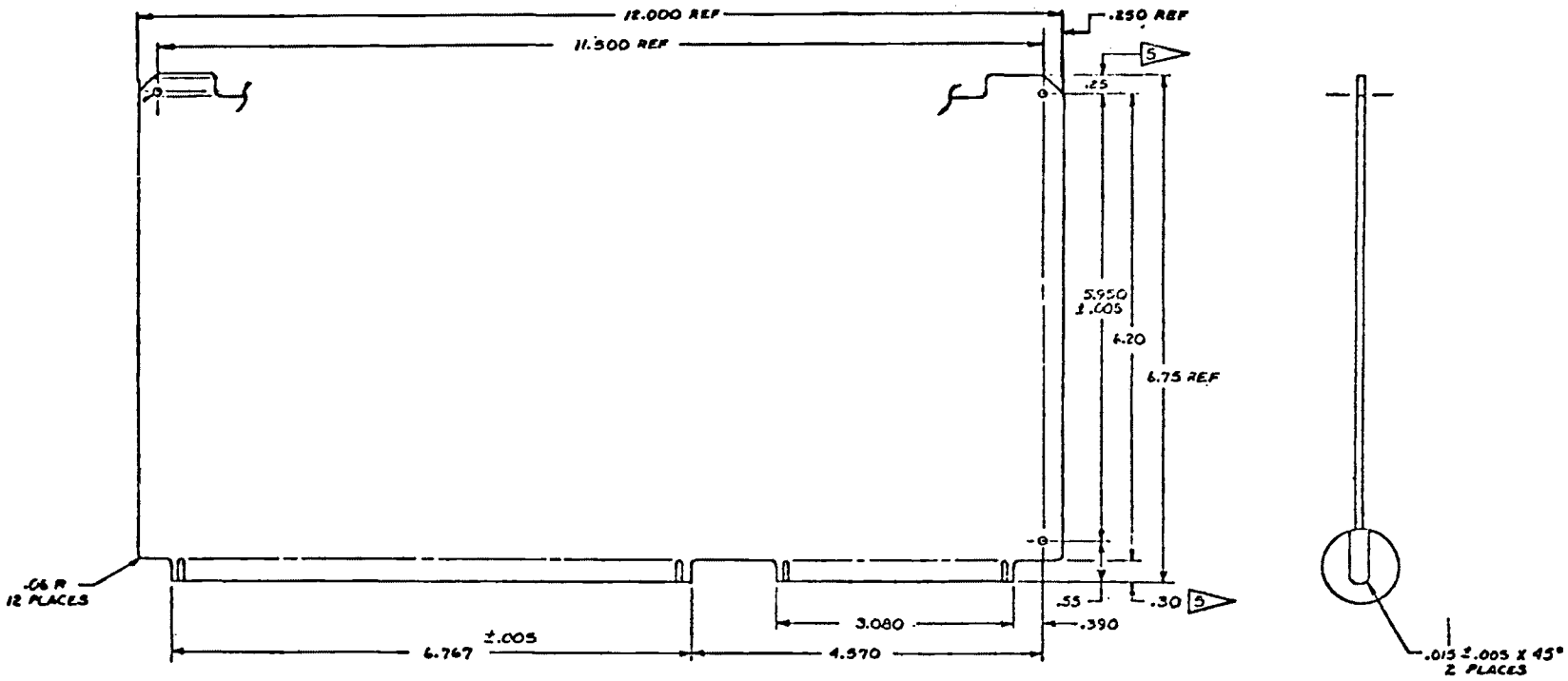


Figure C-2. SBC 80/10 AND SBC 80/10A BOARD OUTLINE (1 of 2)

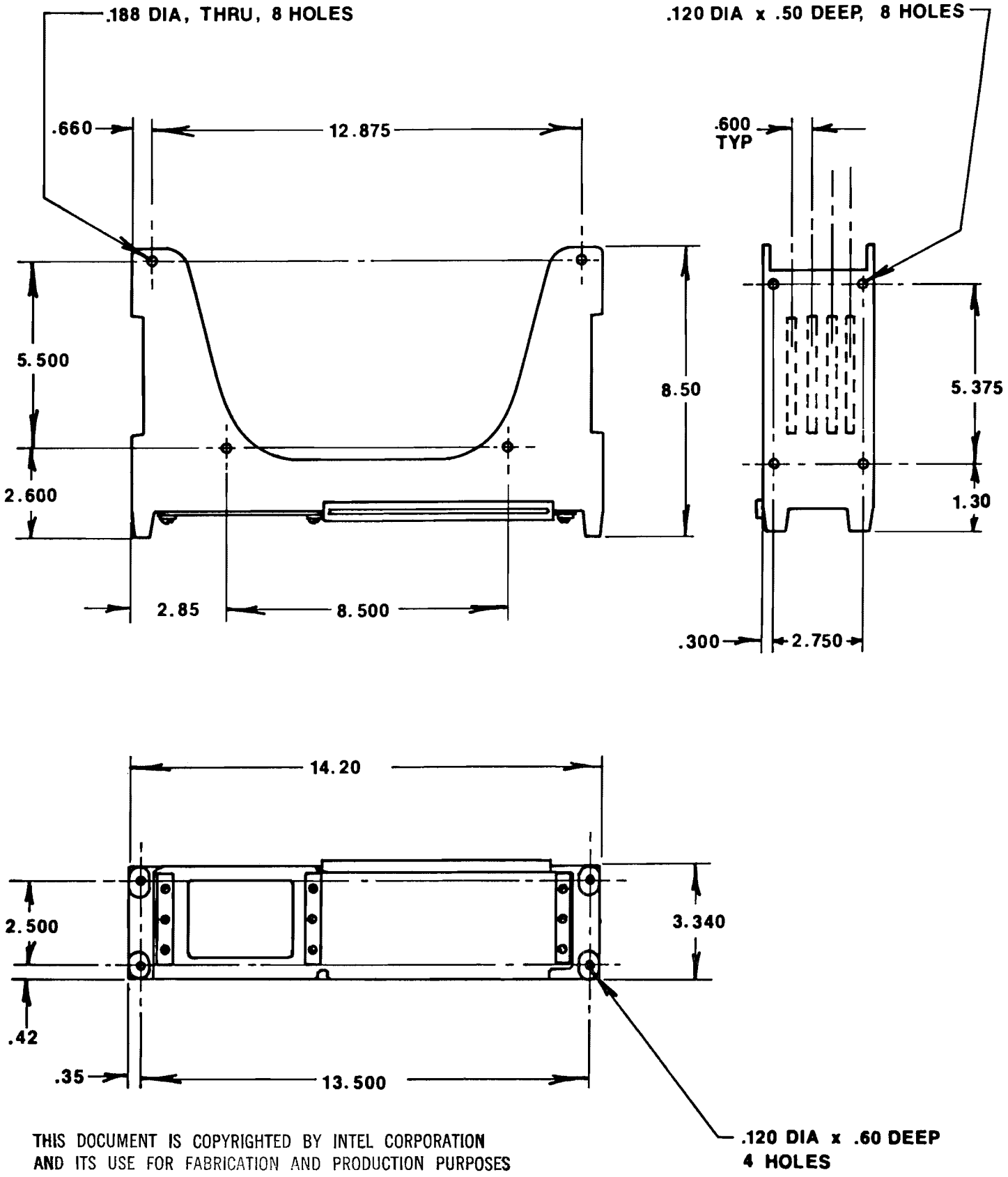


NOTES:

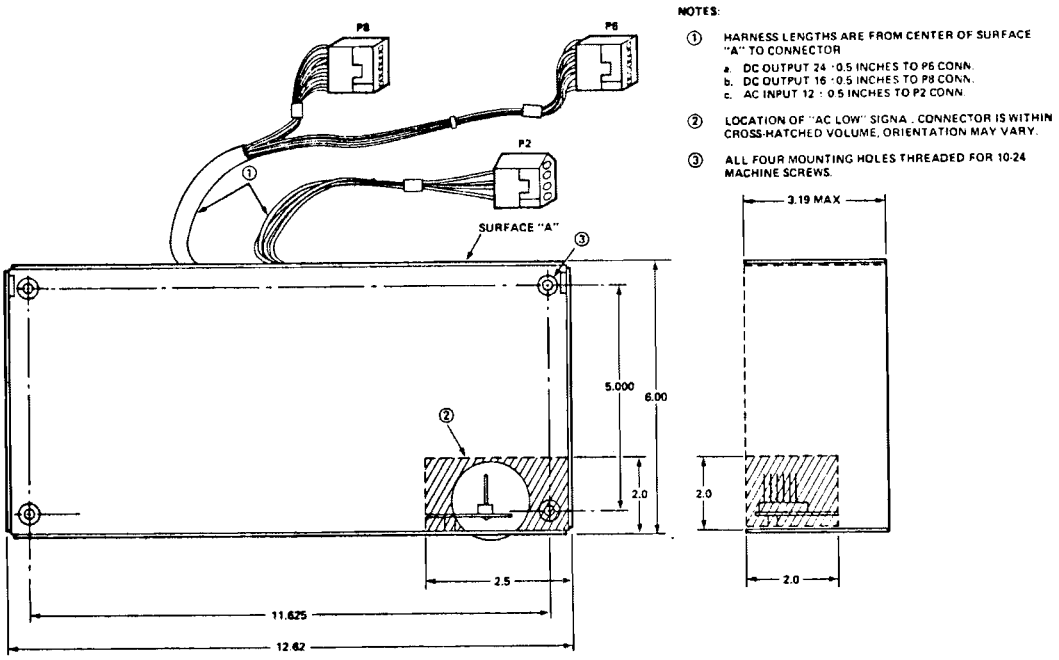
1. MATERIAL : .062 THK, 1 OZ COPPER CLAD, NATURAL EPOXY GLASS, TYPE G10 (2OZ AFTER PLATING THRU)
2. BOARD EDGES ARE LOCATED FROM INDEX HOLES. INDEX HOLES ARE ON .050 GRID INTERSECTION AND ARE USED FOR ARTWORK REGISTRATION AND MAY BE USED AS TOOLING HOLES, PLATING OPTIONAL.
3. HOLES ARE PLATED THRU WITH COPPER WALL THICKNESS OF .0007 MINIMUM.
4. HOLE SIZES SPECIFIED ARE AFTER PLATING; ±.003 TOLERANCE
5. CONTACT FINGERS ARE OVERPLATED WITH A MINIMUM OF 50 MILLIONTHS GOLD OVER NICKEL TO DIMENSION SHOWN.
6. APPLY SOLDER MASK OVER SOLDER PLATE USING MATERIAL; MECUMASK GREEN
7.
8. DRILL FROM CIRCUIT SIDE.
9. TRACE WIDTHS MUST BE WITHIN .004 OF ARTWORK NEGATIVES.
10. APPLY SILKSCREEN ON COMPONENT SIDE, AFTER SOLDER MASK IS APPLIED, USING WHITE EPOXY INK.

Figure C-2. SBC 80/10 AND SBC 80/10A BOARD OUTLINE (2 of 2)

C.3 MODULAR CARDCAGE OUTLINE



C.4 POWER SUPPLY OUTLINE DRAWING FOR V/C AA*



*The System 80/10 Power Supply may come in any one of several versions. External electrical characteristics (including connector types) and mounting information (including overall size) are given for all versions in the above outline and in Chapter 6. Internal parts, schematic, and exact outline dimensions will vary between versions. To determine the version you have, see the two letter code on the side panel closest to the "AC Low" signal output connector. The vendor code is the third group of code following the assembly number and revision level code.

See silkscreen on power supply PC board for location of voltage and current adjustment trimmers.

APPENDIX D

8080 INSTRUCTION SET SUMMARY

A computer, no matter how sophisticated, can only do what it is "told" to do. One "tells" the computer what to do via a series of coded instructions referred to as a Program. The realm of the programmer is referred to as Software, in contrast to the Hardware that comprises the actual computer equipment. A computer's software refers to all of the programs that have been written for that computer.

When a computer is designed, the engineers provide the Central Processing Unit (CPU) with the ability to perform a particular set of operations. The CPU is designed such that a specific operation is performed when the CPU control logic decodes a particular instruction. Consequently, the operations that can be performed by a CPU define the computer's Instruction Set.

Each computer instruction allows the programmer to initiate the performance of a specific operation. All computers implement certain arithmetic operations in their instruction set, such as an instruction to add the contents of two registers. Often logical operations (e.g., OR the contents of two registers) and register operate instructions (e.g., increment a register) are included in the instruction set. A computer's instruction set will also have instructions that move data between registers, between a register and memory, and between register and an I/O device. Most instruction sets also provide Conditional Instructions. A conditional instruction specifies an operation to be performed only if certain conditions have been met; for example, jump to a particular instruction if the result of the last operation was zero. Conditional instructions provide a program with a decision-making capability.

By logically organizing a sequence of instructions into a coherent program, the programmer can "tell" the computer to perform a very specific and useful function.

The computer, however, can only execute programs whose instructions are in a binary coded form (i.e., a series of 1's and 0's), that is called Machine Code. Because it would be extremely cumbersome to program in machine code, programming languages have been developed. There are programs available which convert the programming language instructions into machine code that can be interpreted by the processor.

One type of programming language is Assembly Language. A unique assembly language mnemonic is assigned to each of the computer's instructions. The programmer can write a program (called the Source Program) using these mnemonics and certain operands; the source program is then converted into machine instructions (called the Object Code). Each assembly language instructions is converted into one machine code instruction (1 or more bytes) by an Assembler program. Assembly languages are usually machine dependent (i.e., they are usually able to run on only one type of computer).

THE 8080 INSTRUCTION SET

The 8080 instruction set includes five different types of instructions:

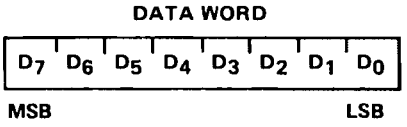
- *Data Transfer Group* – move data between registers or between memory and registers.
- *Arithmetic Group* – add, subtract, increment or decrement data in registers or in memory.
- *Logical Group* – AND, OR, EXCLUSIVE-OR, compare, rotate or complement data in registers or in memory.
- *Branch Group* – conditional and unconditional jump instructions, subroutine call instructions and return instructions.
- *Stack, I/O and Machine Control Group* – includes I/O instructions, as well as instructions for maintaining the stack and internal control flags.

Instruction and Data Formats

Memory for the 8080 is organized into 8-bit quantities, called Bytes. Each byte has a unique 16-bit binary address corresponding to its sequential position in memory.

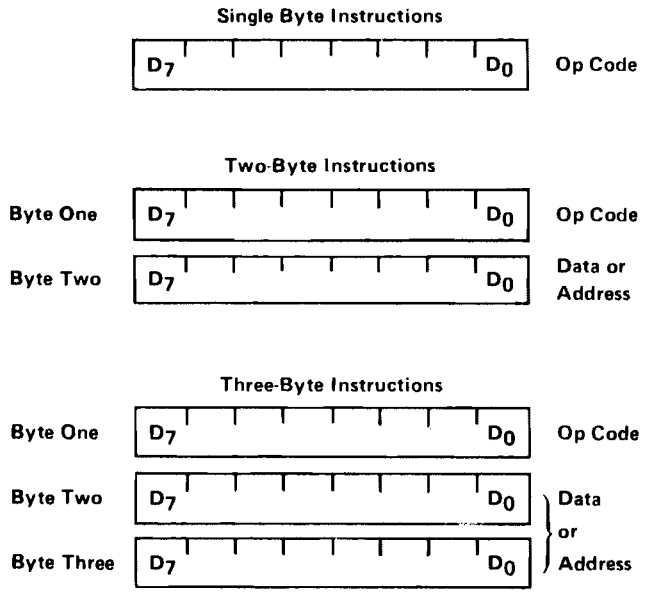
The 8080 can directly address up to 65,536 bytes of memory, which may consist of both read-only memory (ROM) elements and random-access memory (RAM) elements (read/write memory).

Data in the 8080 is stored in the form of 8-bit binary integers:



When a register or data word contains a binary number, it is necessary to establish the order in which the bits of the number are written. In the Intel 8080, BIT 0 is referred to as the Least Significant Bit (LSB), and BIT 7 (of an 8-bit number) is referred to as the Most Significant Bit (MSB).

The 8080 program instructions may be one, two or three bytes in length. Multiple byte instructions must be stored in successive memory locations; the address of the first byte is always used as the address of the instructions. The exact instruction format will depend on the particular operation to be executed.



Addressing Modes

Often the data that is to be operated on is stored in memory. When multi-byte numeric data is used, the data, like instructions, is stored in successive memory locations, with the least significant byte first, followed by increasingly significant bytes. The 8080 has four different modes for addressing data stored in memory or in registers:

- *Direct* – Bytes 2 and 3 of the instruction contain the exact memory address of the data item (the low-order bits of the address are in byte 2, the high-order bits in byte 3).

- *Register* – The instruction specifies the register-pair in which the data is located.

- *Register Indirect* – The instruction specifies a register-pair which contains the memory address where the data is located (the high-order bits of the address are in the first register of the pair, the low-order bits in the second).

- *Immediate* – The instruction contains the data itself. This is either an 8-bit quantity or a 16-bit quantity (least significant byte first, most significant byte second).

Unless directed by an interrupt or branch instruction, the execution of instructions proceeds through consecutively increasing memory locations. A branch instruction can specify the address of the next instruction to be executed in one of two ways:

- *Direct* – The branch instruction contains the address of the next instruction to be executed. (Except for the 'RST' instruction, byte 2 contains the low-order address and byte 3 the high-order address.)

- *Register Indirect* – The branch instruction indicates a register-pair which contains the address of the next instruction to be executed. (The high-order bits of the address are in the first register of the pair, the low-order bits in the second.)

The RST instruction is a special one-byte call instruction (usually used during interrupt sequences).

RST includes a three-bit field; program control is transferred to the instruction whose address is eight times the contents of this three-bit field.

Condition Flags

There are five condition flags associated with the execution of instructions on the 8080. They are Zero, Sign, Parity, Carry, and Auxiliary Carry, and are each represented by a 1-bit register in the CPU. A flag is "set" by forcing the bit to 1; "reset" by forcing the bit to 0.

Unless indicated otherwise, when an instruction affects a flag, it affects it in the following manner:

Zero: If the result of an instruction has the value 0, this flag is set; otherwise it is reset.

Sign: If the most significant bit of the result of the operation has the value 1, this flag is set; otherwise it is reset.

Parity: If the modulo 2 sum of the bits of the result of the operation is 0 (i.e., if the result has even parity), this flag is set; otherwise it is reset (i.e., if the result has odd parity).

Carry: If the instruction resulted in a carry (from addition), or a borrow (from subtraction of a comparison) out of the high-order bit, this flag is set; otherwise it is reset.

Auxiliary Carry: If the instruction caused a carry out of bit 3 and into bit 4 of the resulting value, the auxiliary carry is set; otherwise it is reset. This flag is affected by single precision additions, subtractions, increments, decrements, comparisons, and logical operations, but is principally used with additions and increments preceding a DAA (Decimal Adjust Accumulator) instruction.

Symbols and Abbreviations

The following symbols and abbreviations are used in the subsequent description of the 8080 instructions:

SYMBOLS	MEANING
accumulator	Register A
addr	16-bit address quantity
data	8-bit data quantity
data 16	16-bit data quantity
byte 2	The second byte of the instruction
byte 3	The third byte of the instruction
port	8-bit address of an I/O device
r,r1,r2	One of the registers A,B,C,D,E,H,L
DDD,SSS	The bit pattern designating one of the registers A,B,C,D,E,H,L. (DD= destination, SSS=source):

DDD or SSS	REGISTER NAME
111	A
000	B
001	C
010	D
011	E
100	H
101	L

rp	One of the register pairs: B represents the B,C pair with B as the high-order register and C as the low-order register; D represents the D,E pair with D as the high-order register and E as the low-order register; H represents the H,L pair with H as the high-order register and L as the low-order register; SP represents the 16-bit stack pointer register.
RP	The bit pattern designating one of the register pairs B,D,H,SP:

RP	REGISTER PAIR
00	B-C
01	D-E
10	H-L
11	SP

rh	The first (high-order) register of a designated pair.
rl	The second (low-order) register of a designated register pair.

PC	16-bit program counter register (PCH and PCL are used to refer to the high-order and low-order 8 bits, respectively).
SP	16-bit stack pointer register (SPH and SPL are used to refer to the high-order and low-order 8 bits, respectively).
r_m	Bit m of the register r (bits are number 7 through 0 from left to right).
Z,S,P,CY,AC	The condition flags: Zero, Sign, Parity, Carry, and Auxiliary Carry, respectively.
( )	The contents of the memory location or registers enclosed in the parentheses.
←	"Is transferred to"A
∧	Logical AND
∨	Exclusive OR
V	Inclusive OR
+	Addition
–	Two's complement subtraction
*	Multiplication
↔	"Is exchanged with"
—	The one's complement (e.g., $(\bar{A})$)
n	The restart number 0 through 7
NNN	The binary representation 000 through 111 for restart number 0 through 7, respectively.

Description Format

The following pages provide a detailed description of the instruction set of the 8080. Each instruction is described in the following manner:

1. The MAC 80 assembler format, consisting of the instruction mnemonic and operand fields, is printed in BOLDFACE on the left side of the first line.

2. The name of the instruction is enclosed in parenthesis on the right side of the first line.

3. The next line(s) contain a symbolic description of the operation of the instruction.

4. This is followed by a narative description of the operand of the instruction.

5. The following line(s) contain the binary fields and patterns that comprise the machine instruction.

6. The last four lines contain incidental information about the execution of the instruction. The number of machine cycles and states required to execute the instruction are listed first. If the instruction has two possible execution times, as in a Conditional Jump, both times will be listed, separated by a slash. Next, any significant data addressing modes (see Page A-2) are listed. The last line lists any of the five Flags that are affected by the execution of the instruction.

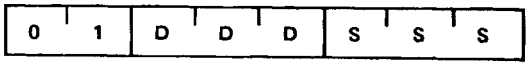
Data Transfer Group

This group of instructions transfer data to and from registers and memory. Condition flags are not affected by any instruction in this group.

MOV r1, r2 (Move Register)

(r1) ← (r2)

The content of register r2 is moved to register r1.



Cycles: 1
States: 5
Addressing: register
Flags: none

MOV r,M (Move from memory)

(r) ← ((H) (L))

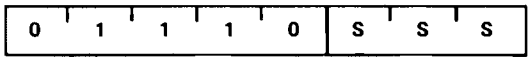
The content of the memory location, whose address is in registers H and L, is moved to register r.

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

MOV M, r (Move to memory)

((H)(L)) ← (r)

The content of register r is moved to the memory location whose address is in registers H and L.

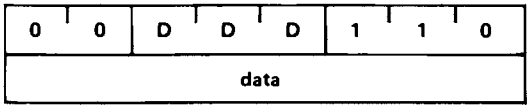


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

MVI r, data (Move Immediate)

(r) ← (byte 2)

The content of byte 2 of the instruction is moved to register r.

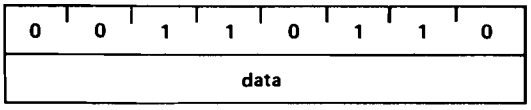


Cycles: 2
States: 7
Addressing: immediate
Flags: none

MVI M, data (Move to memory immediate)

((H)(L)) ← (byte 2)

The content of byte 2 of the instruction is moved to the memory location whose address is in registers H and L.

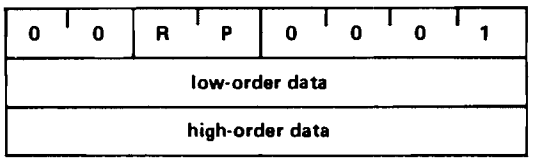


Cycles: 3
States: 10
Addressing: immed./reg. indirect
Flags: none

LXI rp, data 16 (Load register pair immediate)

(rh) ← (byte 3),
(rl) ← (byte 2)

Byte 3 of the instruction is moved into the high-order register (rh) of the register pair rp. Byte 2 of the instruction is moved into the low-order register (rl) of the register pair rp.

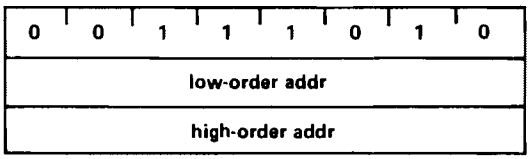


Cycles: 3
States: 10
Addressing: immediate
Flags: none

LDA addr (Load Accumulator direct)

(A) ← ((byte 3)(byte 2))

The content of the memory location, whose address is specified in byte 2 and byte 3 of the instruction, is moved to register A.

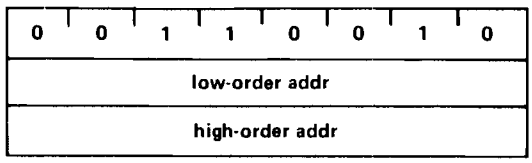


Cycles: 4
States: 13
Addressing: direct
Flags: none

STA addr (Store Accumulator direct)

((byte 3)(byte 2)) ← (A)

The content of the accumulator is moved to the memory location whose address is specified in byte 2 and byte 3 of the instruction.



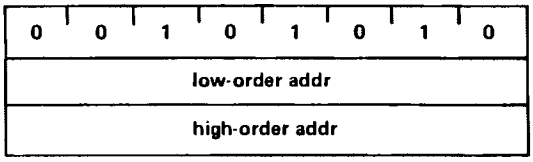
Cycles: 4
States: 13
Addressing: direct
Flags: none

LHLD addr (Load H and L direct)

(L) ← ((byte 3)(byte 2))

(H) ← ((byte 3)(byte 2) + 1)

The content of the memory location, whose address is specified in byte 2 and byte 3 of the instruction, is moved to register L. The content of the memory location at the succeeding address is moved to register H.



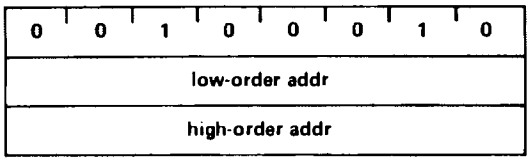
Cycles: 5
States: 16
Addressing: direct
Flags: none

SHLD addr (Store H and L direct)

((byte 3)(byte 2)) ← (L)

((byte 3)(byte 2) + 1) ← (H)

The content of register L is moved to the memory location whose address is specified in byte 2 and byte 3. The content of register H is moved to the succeeding memory location.

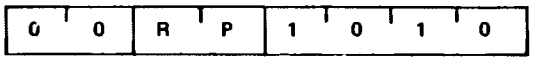


Cycles: 5
States: 16
Addressing: direct
Flags: none

LDAX rp (Load accumulator indirect)

(A) ← ((rp))

The content of the memory location, whose address is in the register pair rp, is moved to register A. Note: Only register pairs rp=B (registers B and C) or rp=D (registers D and E) may be specified.

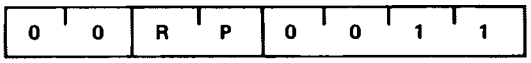


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

STAX rp (Store accumulator indirect)

((rp)) ← (A)

The content of register A is moved to the memory location whose address is in the register pair rp. Note: Only register pairs rp=B (registers B and C) or rp=D (registers D and E) may be specified.



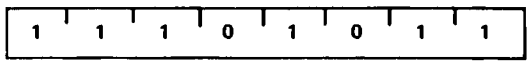
Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

XCHG (Exchange H and L with D and E)

(H) ↔ (D)

(L) ↔ (E)

The contents of registers H and L are exchanged with the contents of registers D and E.

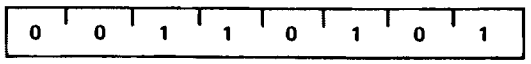


Cycles: 1
States: 4
Addressing: register
Flags: none

DCR M (Decrement memory)

((H)(L)) ← ((H)(L)) – 1

The content of the memory location whose address is contained in the H and L registers is decremented by one. Note: All condition flags except CY are affected.

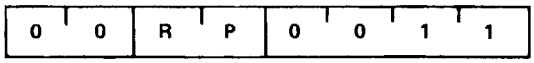


Cycles: 3
States: 10
Addressing: reg. indirect
Flags: Z,S,P,AC

INX rp (Increment register pair)

(rh)(rl) ← (rh)(rl) + 1

The content of the register pair rp is incremented by one. Note: No condition flags are affected.



Cycles: 1
States: 5
Addressing: register
Flags: none

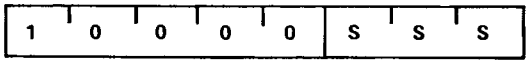
Arithmetic Group

This group of instructions performs arithmetic operations on data in registers and memory.

Unless otherwise indicated, all instructions in this group affect the Zero, Sign, Parity, Carry, and Auxiliary Carry flags according to the standard rules.

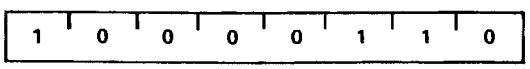
All subtraction operations are performed via two's complement arithmetic, and set the carry flag to one to indicate a borrow and clear it to indicate no borrow.

ADD r (Add Register)
(A) ← (A) + (r)
The content of register r is added to the content of the accumulator. The result is placed in the accumulator.



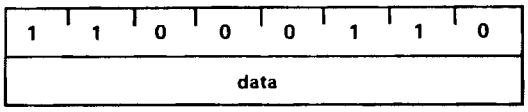
Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

ADD M (Add Memory)
(A) ← (A) + ((H)(L))
The content of the memory location whose address is contained in the H and L register is added to the content of the accumulator. The result is placed in the accumulator.



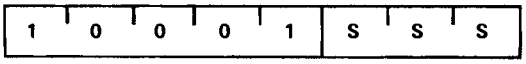
Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

ADI data (Add Immediate)
(A) ← (A) + (byte 2)
The content of the second byte of the instruction is added to the constant of the accumulator. The result is placed in the accumulator.



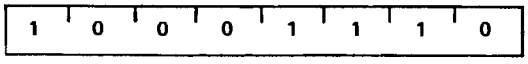
Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

ADC r (Add Register with Carry)
(A) ← (A) + (r) + (CY)
The content of register r and the content of the carry bit are added to the content of the accumulator. The result is placed in the accumulator.



Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

ADC M (Add Memory with Carry)
(A) ← (A) + ((H)(L)) + (CY)
The content of the memory location whose address is contained in the H and L registers and the content of the CY flag are added to the accumulator. The result is placed in the accumulator.

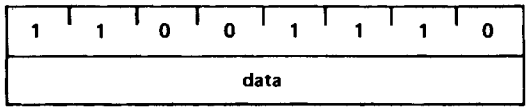


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

ACI data (Add Immediate with Carry)

(A) ← (A) + (byte 2) + (CY)

The content of the second byte of the instruction and the content of the CY flag are added to the contents of the accumulator. The result is placed in the accumulator.

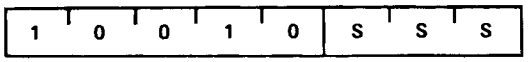


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

SUB r (Subtract Register)

(A) ← (A) – (r)

The content of register r is subtracted from the content of the accumulator. The result is placed in the accumulator.

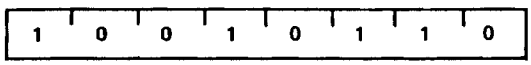


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

SUB M (Subtract Memory)

(A) ← (A) – ((H)(L))

The content of the memory location whose address is contained in the H and L registers is subtracted from the content of the accumulator. The result is placed in the accumulator.

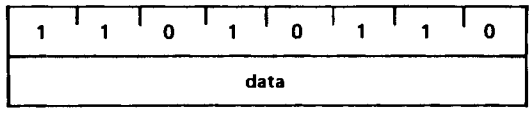


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

SUI data (Subtract Immediate)

(A) ← (A) – (byte 2)

The content of the second byte of the instruction is subtracted from the content of the accumulator. The result is placed in the accumulator.

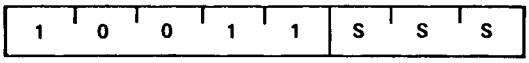


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

SBB r (Subtract Register with Borrow)

(A) ← (A) – (r) – (CY)

The content of register r and the content of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.



Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

SBB M (Subtract Memory with Borrow)

(A) ← (A) – ((H)(L)) – (CY)

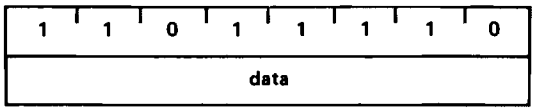
The content of the memory location whose address is contained in the H and L registers and the content of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

SBI data (Subtract Immediate with Borrow)

(A) ← (A) – (byte 2) – (CY)

The contents of the second byte of the instruction and the contents of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.

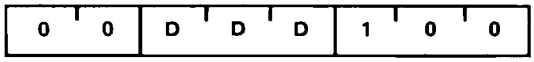


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

INR r (Increment Register)

(r) ← (r) + 1

The content of register r is incremented by one. Note: All condition flags except CY are affected.

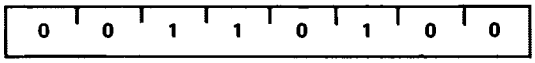


Cycles: 1
States: 5
Addressing: register
Flags: Z,S,P,AC

INR M (Increment Memory)

((H)(L)) ← ((H)(L)) + 1

The content of the memory location whose address is contained in the H and L registers is incremented by one. Note: All condition flags except CY are affected.

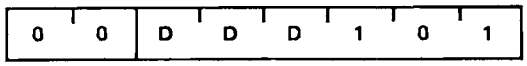


Cycles: 3
States: 10
Addressing: reg. indirect
Flags: Z,S,P,AC

DCR r (Decrement Register)

(r) ← (r) – 1

The content of register r is decremented by one. Note: All condition flags except CY are affected.



Cycles: 1
States: 5
Addressing: register
Flags: Z,S,P,AC

DCX rp (Decrement register pair)

(rh)(rl) ← (rh)(rl) – 1

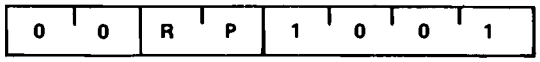
The content of the register pair rp is decremented by one. Note: No condition flags are affected.

Cycles: 1
States: 5
Addressing: register
Flags: none

DAD rp (Add register pair to H and L)

(H)(L) ← (H)(L) + (rh)(rl)

The content of the register pair rp is added to the content of the register pair H and L. The result is placed in the register pair H and L. Note: Only the CY flag is affected. It is set if there is a carry out of the double precision add; otherwise it is reset.



Cycles: 3
States: 10
Addressing: register
Flags: CY

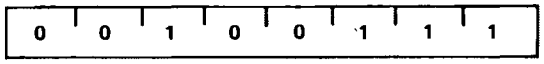
DAA (Decimal Adjust Accumulator)

The 8-bit number in the accumulator is adjusted to form two 4-bit Binary-Coded-Decimal digits by the following process:

1. If the value of the least significant 4 bits of the accumulator is greater than 9 or if the AC flag is set, 6 is added to the accumulator.

2. If the value of the most significant 4 bits of the accumulator is now greater than 9, or if the CY flag is set, 6 is added to the most significant 4 bits of the accumulator.

NOTE: All flags are affected.



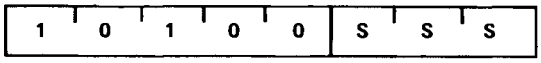
Cycles: 1
States: 4
Flags: Z,S,P,CY,AC

Logical Group

This group of instructions performs logical (Boolean) operations on data in registers and memory and on condition flags.

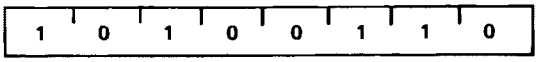
Unless indicated otherwise, all instructions in this group affect the Zero, Sign, Parity, Auxiliary Carry, and Carry flags according to the standard rules.

ANA r (AND Register)
(A) ← (A) ∧ (r)
The content of register r is logically ANDed with the content of the accumulator. The result is placed in the accumulator. The CY flag is cleared.



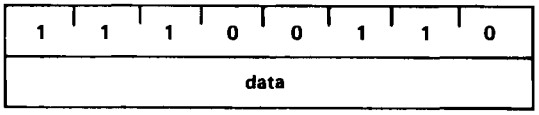
Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

ANA M (AND memory)
(A) ← (A) ∧ ((H)(L))
The contents of the memory location whose address is contained in the H and L registers is logically ANDed with the content of the accumulator. The CY flag is cleared.



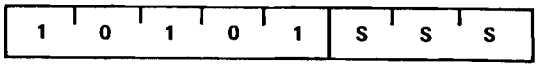
Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

ANI data (AND immediate)
(A) ← (A) ∧ (byte 2)
The content of the second byte of the instruction is logically ANDed with the contents of the accumulator. The result is placed in the accumulator. The CY flag is cleared.



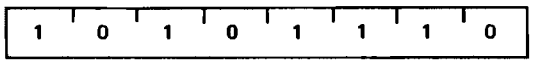
Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

XRA r (Exclusive OR Register)
(A) ← (A) ∀ (r)
The content of register r is exclusive-ORed with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.



Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

XRA M (Exclusive OR Memory)
(A) ← (A) ∀ ((H)(L))
The content of the memory location whose address is contained in the H and L registers is exclusive-ORed with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

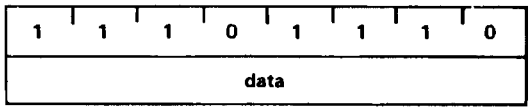


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

XRI data (Exclusive OR immediate)

(A) ← (A) V (byte 2)

The content of the second byte of the instruction is exclusive-ORed with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

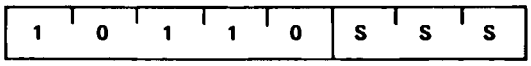


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

ORA r (OR Register)

(A) ← (A) V (r)

The content of register r is inclusive-ORed with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

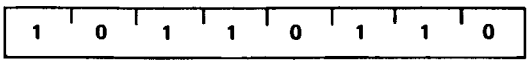


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

ORA M (OR Memory)

(A) ← (A) V ((H)(L))

The content of the memory location whose address is contained in the H and L registers is inclusive-ORed with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

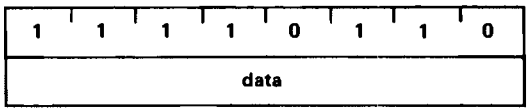


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

ORI data (OR Immediate)

(A) ← (A) V (byte 2)

The content of the second byte of the instruction is inclusive-ORed with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.



Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

CMP r (Compare Register)

(A) – (r)

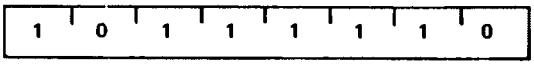
The content of register r is subtracted from the accumulator. The accumulator remains unchanged. The condition flags are set as a result of the subtraction. The Z flag is set to 1 if (A) = (r). The CY flag is set to 1 if (A) < (r).

Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

CMP M (Compare memory)

(A) – ((H)(L))

The content of the memory location whose address is contained in the H and L registers is subtracted from the accumulator. The accumulator remains unchanged. The condition flags are set as a result of the subtraction. The Z flag is set to 1 of (A) = ((H)(L)). The CY flag is set to 1 if (A) < ((H)(L)).

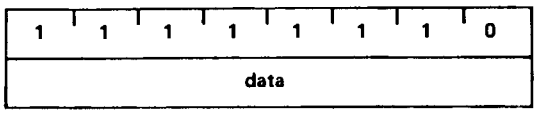


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S.P,CY,AC

CPI data (Compare immediate)

(A) – (byte 2)

The content of the second byte of the instruction is subtracted from the accumulator. The condition flags are set by the result of the subtraction. The Z flag is set to 1 if (A) = (byte 2). The CY flag is set to 1 if (A) < (byte 2).



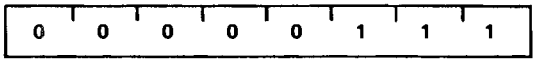
Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

RLC (Rotate left)

$(A_{n+1}) \leftarrow (A_n)$; $(A_0) \leftarrow (A_7)$

$(CY) \leftarrow (A_7)$

The content of the accumulator is rotated left one position. The low-order bits and the CY flag are both set to the value shifted out of the high-order bit position. Only the CY flag is affected.



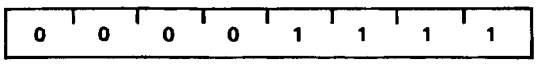
Cycles: 1
States: 4
Flags: CY

RRC (Rotate right)

$(A_n) \leftarrow (A_{n-1})$; $(A_7) \leftarrow (A_0)$

$(CY) \leftarrow (A_0)$

The content of the accumulator is rotated right one position. The high-order bit and the CY flag are both set to the value shifted out of the low-order bit position. Only the CY flag is affected.



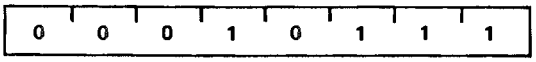
Cycles: 1
States: 4
Flags: CY

RAL (Rotate left through carry)

$(A_{n+1}) \leftarrow (A_n)$; $(CY) \leftarrow (A_7)$

$(A_0) \leftarrow (CY)$

The content of the accumulator is rotated left one position through the CY flag. The low-order bit is set equal to the CY flag and the CY flag is set to the value shifted out of the high-order bit. Only the CY flag is affected.



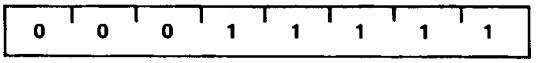
Cycles: 1
States: 4
Flags: CY

RAR (Rotate right through carry)

$(A_n) \leftarrow (A_{n+1})$; $(CY) \leftarrow (A_0)$

$(A_7) \leftarrow (CY)$

The content of the accumulator is rotated right one position through the CY flag. The high-order bit is set to the CY flag and the CY flag is set to the value shifted out of the low-order bit. Only the CY flag is affected.

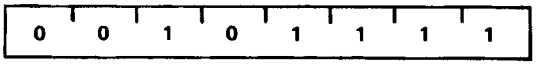


Cycles: 1
States: 4
Flags: CY

CMA (Complement accumulator)

$(A) \leftarrow (A)$

The contents of the accumulator are complemented (zero bits become 1, one bits become 0). No flags are affected.

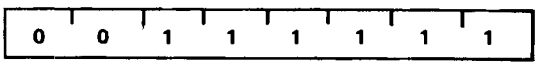


Cycles: 1
States: 4
Flags: none

CMC (Complement carry)

$(CY) \leftarrow (CY)$

The CY flag is complemented. No other flags are affected.

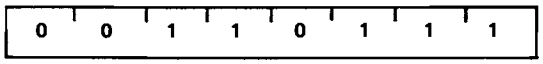


Cycles: 1
States: 4
Flags: CY

STC (Set carry)

(CY) ← 1

The CY flag is set to 1. No other flags are affected.



Cycles: 1
States: 4
Flags: CY

Branch Group

This group of instructions alter normal sequential program flow.

Condition flags are not affected by an instruction in this group.

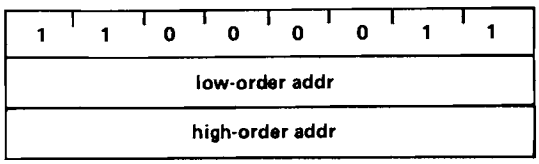
The two types of branch instructions are unconditional and conditional. Unconditional transfers simply perform the specified operation on register PC (the program counter). Conditional transfers examine the status of one of the four processor flags to determine if the specified branch is to be executed. The conditions that may be specified are as follows:

CONDITION		CCC
NZ	– not zero (Z=0)	000
Z	– zero (Z = 1)	001
NC	– no carry (C = 0)	010
C	– carry (CY = 1)	011
PO	– parity odd (P = 0)	100
PE	– parity even (P = 1)	101
P	– plus (S = 0)	110
M	– minus (S = 1)	111

JMP addr (Jump)

(PC) → (byte 3)(byte 2)

Control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction.



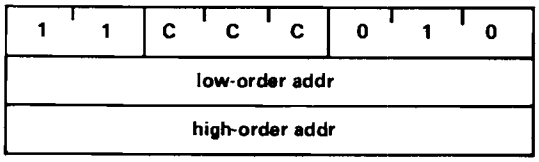
Cycles: 3
States: 10
Addressing: immediate
Flags: none

Jcondition addr (Conditional jump)

If (CCC),

(PC) ← (byte 3)(byte 2)

If the specified condition is true, control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction; otherwise, control continues sequentially.



Cycles: 3
States: 10
Addressing: immediate
Flags: none

CALL addr (Call)

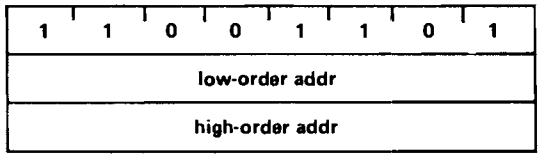
((SP) – 1) ← (PCH)

((SP) – 2) ← (PCL)

(SP) ← (SP) – 2

(PC) ← (byte 3)(byte 2)

The high-order 8 bits of the next instruction address are moved to the memory location whose address is one less than the content of register SP. The low-order 8 bits of the next instruction address are moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by 2. Control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction.

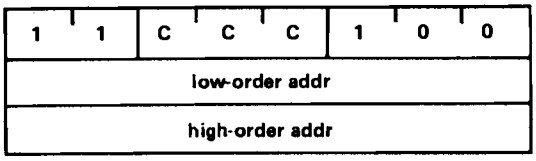


Cycles: 5
States: 17
Addressing: immed./reg. indirect
Flags: none

Ccondition addr (Condition call)

If (CCC),
((SP) – 1) ← (PCH)
((SP) – 2) ← (PCL)
(SP) ← (SP) – 2
(PC) ← (byte 3)(byte 2)

If the specified condition is true, the actions specified in the CALL instruction (see above) are performed; otherwise, control continues sequentially.

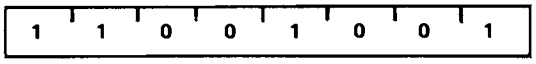


Cycles: 3/5
States: 11/17
Addressing: immed./reg. indirect
Flags: none

RET (Return)

(PCL) ← ((SP));
(PCH) ← ((SP) + 1);
(SP) ← (SP) + 2;

The content of the memory location whose address is specified in register SP is moved to the low-order 8 bits of register PC. The content of the memory location whose address is one more than the content of register SP is moved to the high-order 8 bits of register PC. The content of register SP is incremented by 2.

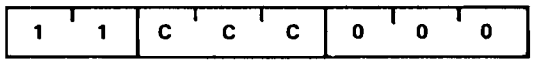


Cycles: 3
States: 10
Addressing: reg. indirect
Flags: none

Rcondition (Conditional return)

If (CCC),
(PCL) ← ((SP))
(PCH) ← ((SP) + 1)
(SP) ← (SP) + 2

If the specified condition is true, the actions specified in the RET instruction (see above) are performed; otherwise, control continues sequentially.



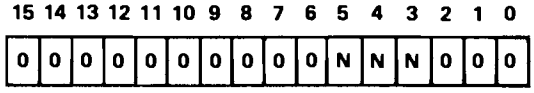
Cycles: 1/3
States: 5/11
Addressing: reg. indirect
Flags: none

RST n (Restart)

((SP) – 1) ← (PCH)
((SP) – 2) ← (PCL)
(SP) ← (SP) – 2
(PC) ← 8 * (NNN)

The high-order 8 bits of the next instruction address are moved to the memory location whose address is one less than the content of register SP. The low-order 8 bits of the next instruction address are moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by two. Control is transferred to the instruction whose address is eight times the content of NNN.

Cycles: 3
States: 11
Addressing: reg. indirect
Flags: none

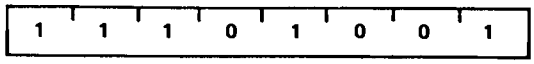


Program Counter After Restart

PCHL (Jump H and L indirect – move H and L to PC)

(PCH) ← (H)
(PCL) ← (L)

The content of register H is moved to the high-order 8 bits of register PC. The content of register L is moved to the low-order 8 bits of register PC.



Cycles: 1
States: 5
Addressing: register
Flags: none

Stack, I/O, and Machine Control Group

This group of instructions performs I/O, manipulates the Stack, and alters internal control flags.

Unless otherwise specified, condition flags are not affected by any instructions in this group.

FLAG WORD

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z	0	AC	0	P	1	CY

PUSH rp (Push)

((SP) − 1) ← (rh)
((SP) − 2) ← (rl)
(SP) ← (SP) − 2

The content of the high-order register of register pair rp is moved to the memory location whose address is one less than the content of register SP. The content of the low-order register of register pair rp is moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by 2. Note: Register pair rp=SP may not be specified.

1	1	R	P	0	1	0	1

Cycles: 3
States: 11
Addressing: reg. indirect
Flags: none

POP rp (Pop)

(rl) ← ((SP))
(rh) ← ((SP) + 1)
(SP) ← (SP) + 2

The content of the memory location, whose address is specified by the content of register SP, is moved to the low-order register of register pair rp. The content of the memory location, whose address is one more than the content of register SP, is moved to the high-order register of register pair rp. The content of register SP is incremented by 2. Note: Register pair rp=SP may not be specified.

1	1	R	P	0	0	0	1

Cycles: 3
States: 10
Addressing: reg. indirect
Flags: none

PUSH PSW (Push processor status word)

((SP) − 1) ← (A)
$((SP) - 2)_0$ ← (CY, $((SP) - 2)_1$ ← 1
$((SP) - 2)_2$ ← (P), $((SP) - 2)_3$ ← 0
$((SP) - 2)_4$ ← (AC), $((SP) - 2)_5$ ← 0
$((SP) - 2)_6$ ← (Z), $((SP) - 2)_7$ ← (S)
(SP) ← (SP) − 2

The content of register A is moved to the memory location whose address is one less than register SP. The contents of the condition flags are assembled into a processor status word and the word is moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by two.

1	1	1	1	0	1	0	1

Cycles: 3
States: 11
Addressing: reg. indirect
Flags: none

POP PSW (Pop processor status word)

(CY) ← $((SP))_0$
(P) ← $((SP))_2$
(AC) ← $((SP))_4$
(Z) ← $((SP))_6$
(S) ← $((SP))_7$
(A) ← ((SP) + 1)
(SP) ← (SP) + 2

The content of the memory location whose address is specified by the content of register SP is used to restore the condition flags. The content of the memory location whose address is one more than the content of register SP is moved to register A. The content of register SP is incremented by 2.

1	1	1	1	0	0	0	1

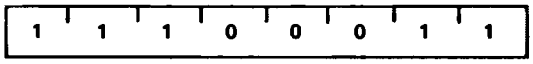
Cycles: 3
States: 10
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

XTHL (Exchange stack top with H and L)

(L) ↔ ((SP))

(H) ↔ ((SP) + 1)

The content of the L register is exchanged with the content of the memory location whose address is specified by the content of register SP. The content of the H register is exchanged with the content of the memory location whose address is one more than the content of register SP.

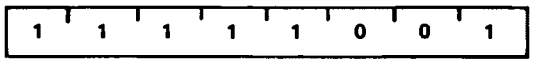


Cycles: 5
States: 18
Addressing: reg. indirect
Flags: none

SPHL (Move HL to SP)

(SP) ← (H)(L)

The contents of registers H and L (16 bits) are moved to register SP.

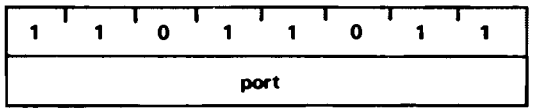


Cycles: 1
States: 5
Addressing: register
Flags: none

IN port (Input)

(A) ← (data)

The data placed on the 8-bit bidirectional data bus by the specified port is moved to register A.

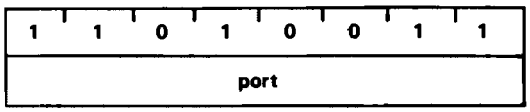


Cycles: 3
States: 10
Addressing: direct
Flags: none

OUT port (Output)

(data) ← (A)

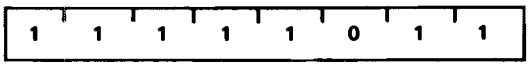
The content of register A is placed on the 8-bit bidirectional data bus for transmission to the specified port.



Cycles: 3
States: 10
Addressing: direct
Flags: none

EI (Enable interrupt)

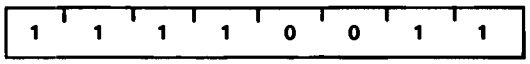
The interrupt system is enabled following the execution of the next instruction.



Cycles: 1
States: 4
Flags: none

DI (Disable interrupts)

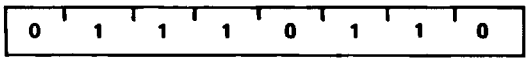
The interrupt system is disabled immediately following the execution of the DI instruction.



Cycles: 1
States: 4
Flags: none

HLT (Halt)

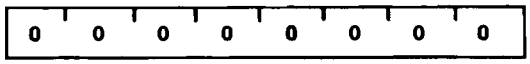
The processor is stopped. The registers and flags are unaffected.



Cycles: 1
States: 7
Flags: none

NOP (No op)

No operation is performed. The registers and flags are unaffected.



Cycles: 1
States: 4
Flags: none

INSTRUCTION SET

Summary of Processor Instructions

MNEMONIC	DESCRIPTION	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	CLOCK[2] CYCLES
$MOV_{r1,r2}$	Move register to register	0	1	D	D	D	S	S	S	5
MOV M,r	Move register to memory	0	1	1	1	0	S	S	S	7
MOV r,M	Move memory to register	0	1	D	D	D	1	1	0	7
HLT	Halt	0	1	1	1	0	1	1	0	7
MVI r	Move immediate register	0	0	D	D	D	1	1	0	7
MVI M	Move immediate memory	0	0	1	1	0	1	1	0	10
INR r	Increment register	0	0	D	D	D	1	0	0	5
DCR r	Decrement register	0	0	D	D	D	1	0	1	5
INR M	Increment memory	0	0	1	1	0	1	0	0	10
DCR M	Decrement memory	0	0	1	1	0	1	0	1	10
ADD r	Add register to A	1	0	0	0	0	S	S	S	4
ADC r	Add register to A with carry	1	0	0	0	1	S	S	S	4
SUB r	Subtract register from A	1	0	0	1	0	S	S	S	4
SBB r	Subtract register from A with borrow	1	0	0	1	1	S	S	S	4
ANA r	And register with A	1	0	1	0	0	S	S	S	4
XRA r	Exclusive Or register with A	1	0	1	0	1	S	S	S	4
ORA r	Or register with A	1	0	1	1	0	S	S	S	4
CMP r	Compare register with A	1	0	1	1	1	S	S	S	4
ADD M	Add memory to A	1	0	0	0	0	1	1	0	7
ADC M	Add memory to A with carry	1	0	0	0	1	1	1	0	7
SUB M	Subtract memory from A	1	0	0	1	0	1	1	0	7
SBB M	Subtract memory from A with borrow	1	0	0	1	1	1	1	0	7
ANA M	And memory with A	1	0	1	0	0	1	1	0	7
XRA M	Exclusive Or memory with A	1	0	1	0	1	1	1	0	7
ORA M	Or memory with A	1	0	1	1	0	1	1	0	7
CMP M	Compare memory with A	1	0	1	1	1	1	1	0	7
ADI	Add immediate to A	1	1	0	0	0	1	1	0	7
ACI	Add immediate to A with carry	1	1	0	0	1	1	1	0	7
SUI	Subtract immediate from A	1	1	0	1	0	1	1	0	7
SBI	Subtract immediate from A with borrow	1	1	0	1	1	1	1	0	7
ANI	And immediate with A	1	1	1	0	0	1	1	0	7
XRI	Exclusive Or immediate with A	1	1	1	0	1	1	1	0	7
ORI	Or immediate with A	1	1	1	1	0	1	1	0	7
CPI	Compare immediate with A	1	1	1	1	1	1	1	0	7
RLC	Rotate A left	0	0	0	0	0	1	1	1	4
RRC	Rotate A right	0	0	0	0	1	1	1	1	4
RAL	Rotate A left through carry	0	0	0	1	0	1	1	1	4
RAR	Rotate A right through carry	0	0	0	1	1	1	1	1	4
JMP	Jump unconditional	1	1	0	0	0	0	1	1	10
JC	Jump on carry	1	1	0	1	1	0	1	0	10
JNC	Jump on no carry	1	1	0	1	0	0	1	0	10
JZ	Jump on zero	1	1	0	0	1	0	1	0	10
JNZ	Jump on no zero	1	1	0	0	0	0	1	0	10
JP	Jump on positive	1	1	1	1	0	0	1	0	10
JM	Jump on minus	1	1	1	1	1	0	1	0	10
JPE	Jump on parity even	1	1	1	0	1	0	1	0	10
JPO	Jump on parity odd	1	1	1	0	0	0	1	0	10
CALL	Call unconditional	1	1	0	0	1	1	0	1	17
CC	Call on carry	1	1	0	1	1	1	0	0	11/17
CNC	Call on no carry	1	1	0	1	0	1	0	0	11/17
CZ	Call on zero	1	1	0	0	1	1	0	0	11/17
CNZ	Call on no zero	1	1	0	0	0	1	0	0	11/17
CP	Call on positive	1	1	1	1	0	1	0	0	11/17
CM	Call on minus	1	1	1	1	1	1	0	0	11/17
CPE	Call on parity even	1	1	1	0	1	1	0	0	11/17
CPO	Call on parity odd	1	1	1	0	0	1	0	0	11/17
RET	Return	1	1	0	0	1	0	0	1	10
RC	Return on carry	1	1	0	1	1	0	0	0	5/11
RNC	Return on no carry	1	1	0	1	0	0	0	0	5/11
RZ	Return on zero	1	1	0	0	1	0	0	0	5/11
RNZ	Return on no zero	1	1	0	0	0	0	0	0	5/11
RP	Return on positive	1	1	1	1	0	0	0	0	5/11
RM	Return on minus	1	1	1	1	1	0	0	0	5/11
RPE	Return on parity even	1	1	1	0	1	0	0	0	5/11
RPO	Return on parity odd	1	1	1	0	0	0	0	0	5/11
RST	Restart	1	1	A	A	A	1	1	1	11
IN	Input	1	1	0	1	1	0	1	1	10
OUT	Output	1	1	0	1	0	0	1	1	10
LXI B	Load immediate register Pair B & C	0	0	0	0	0	0	0	1	10
LXI D	Load immediate register Pair D & E	0	0	0	1	0	0	0	1	10
LXI H	Load immediate register Pair H & L	0	0	1	0	0	0	0	1	10
LXI SP	Load immediate stack pointer	0	0	1	1	0	0	0	1	10
PUSH B	Push register Pair B & C on stack	1	1	0	0	0	1	0	1	11
PUSH D	Push register Pair D & E on stack	1	1	0	1	0	1	0	1	11
PUSH H	Push register Pair H & L on stack	1	1	1	0	0	1	0	1	11
PUSH PSW	Push A and Flags on stack	1	1	1	1	0	1	0	1	11
POP B	Pop register pair B & C off stack	1	1	0	0	0	0	0	1	10
POP D	Pop register pair D & E off stack	1	1	0	1	0	0	0	1	10
POP H	Pop register pair H & L off stack	1	1	1	0	0	0	0	1	10
POP PSW	Pop A and Flags off stack	1	1	1	1	0	0	0	1	10
STA	Store A direct	0	0	1	1	0	0	1	0	13
LDA	Load A direct	0	0	1	1	1	0	1	0	13
XCHG	Exchange D & E, H & L Registers	1	1	1	0	1	0	1	1	4
XTHL	Exchange top of stack H & L	1	1	1	0	0	0	1	1	18
SPHL	H & L to stack pointer	1	1	1	1	1	0	0	1	5
PCHL	H & L to program counter	1	1	1	0	1	0	0	1	5
DAD B	Add B & C to H & L	0	0	0	0	1	0	0	1	10
DAD D	Add D & E to H & L	0	0	0	1	1	0	0	1	10
DAD H	Add H & L to H & L	0	0	1	0	1	0	0	1	10
DAD SP	Add stack pointer to H & L	0	0	1	1	1	0	0	1	10
STAX B	Store A indirect	0	0	0	0	0	0	1	0	7
STAX D	Store A indirect	0	0	0	1	0	0	1	0	7
LDAX B	Load A indirect	0	0	0	0	1	0	1	0	7
LDAX D	Load A indirect	0	0	0	1	1	0	1	0	7
INX B	Increment B & C registers	0	0	0	0	0	0	1	1	5
INX D	Increment D & E registers	0	0	0	1	0	0	1	1	5
INX H	Increment H & L registers	0	0	1	0	0	0	1	1	5
INX SP	Increment stack pointer	0	0	1	1	0	0	1	1	5
DCX B	Decrement B & C	0	0	0	0	1	0	1	1	5
DCX D	Decrement D & E	0	0	0	1	1	0	1	1	5
DCX H	Decrement H & L	0	0	1	0	1	0	1	1	5
DCX SP	Decrement stack pointer	0	0	1	1	1	0	1	1	5
CMA	Complement A	0	0	1	0	1	1	1	1	4
STC	Set carry	0	0	1	1	0	1	1	1	4
CMC	Complement carry	0	0	1	1	1	1	1	1	4
DAA	Decimal adjust A	0	0	1	0	0	1	1	1	4
SHLD	Store H & L direct	0	0	1	0	0	0	1	0	16
LHLD	Load H & L direct	0	0	1	0	1	0	1	0	16
EI	Enable Interrupts	1	1	1	1	1	0	1	1	4
DI	Disable interrupt	1	1	1	1	1	0	0	1	4
NOP	No-operation	0	0	0	0	0	0	0	0	4

NOTES: 1. DDD or SSS - 000 B - 001 C - 010 D - 011 E - 100 H - 101 L - 110 Memory - 111 A.
2. Two possible cycle times, (5/11) indicate instruction cycles dependent on condition flags.

APPENDIX E

SBC 80P MONITOR PROGRAM LISTING

```
        TITLE '80/10 MONITOR, VERSION 1.1, 1 NOVEMBER 1976'
;
;
;*****************************************************************
;*****************************************************************
;
;                       80/10 MONITOR
;                       M80/10
;                       VERSION 1.1
;                       1 NOVEMBER 1976
;
;*****************************************************************
;*****************************************************************
;
;    (C) 1976 INTEL CORPORATION. ALL RIGHTS RESERVED. NO PART OF THIS
;    PROGRAM OR PUBLICATION MAY BE REPRODUCED, TRANSMITTED, TRANSCRIBED,
;    STORED IN A RETRIEVAL SYSTEM, OR TRANSLATED INTO ANY LANGUAGE OR
;    COMPUTER LANGUAGE, IN ANY FORM OR BY ANY MEANS, ELECTRONIC,
;    MECHANICAL, MAGNETIC, OPTICAL, CHEMICAL, MANUAL OR OTHERWISE,
;    WITHOUT THE PRIOR WRITTEN PERMISSION OF INTEL CORPORATION,
;    3065 BOWERS AVENUE, SANTA CLARA, CALIFORNIA 95051.
;
;*****************************************************************
;*****************************************************************
;
;
; ABSTRACT
; ========
;
; THIS PROGRAM RUNS ON THE SBC 80/10 BOARD AND IS DESIGNED TO PROVIDE
; THE USER WITH A MINIMAL MONITOR.  BY USING THIS PROGRAM,
; THE USER CAN EXAMINE AND CHANGE MEMORY OR CPU REGISTERS, LOAD
; A PROGRAM (IN ABSOLUTE HEX) INTO RAM, AND EXECUTE INSTRUCTIONS
; ALREADY IN MEMORY.  THE MONITOR ALSO PROVIDES THE USER WITH
; ROUTINES FOR PERFORMING CONSOLE I/O AND PAPER TAPE I/O.
;
;
; PROGRAM ORGANIZATION
; ======= ============
;
; THE LISTING IS ORGANIZED IN THE FOLLOWING WAY. FIRST THE BASIC
; MONITOR FUNCTIONS TOGETHER WITH THE CONSOLE I/O ARE LOCATED IN THE
; FIRST 1K OF ROM FOLLOWED BY THE PAPER TAPE FUNCTIONS AND I/O IN THE
; SECOND 1K OF ROM. WITHIN THE FIRST ROM IS CONTAINED THE COMMAND
; RECOGNIZER, WHICH IS THE HIGHEST LEVEL ROUTINE IN THE PROGRAM.
; NEXT THE ROUTINES TO IMPLEMENT THE VARIOUS COMMANDS.  FINALLY,
; THE UTILITY ROUTINES WHICH ACTUALLY DO THE DIRTY WORK.  WITHIN
; EACH SECTION, THE ROUTINES ARE ORGANIZED IN ALPHABETICAL
; ORDER, BY ENTRY POINT OF THE ROUTINE. THE SECOND ROM IS ORGANIZED
; IN THE SAME MANNER AS THE FIRST WITH THE ROUTINES WHICH IMPLIMENT
; THE COMMANDS FOLLOWED BY THE UTILITY ROUTINES WHICH ACTUALLY DO THE
```

```
; MORE DETAILED OPERATIONS.
;
; THE PROGRAM HAS BEEN PARTITIONED IN SUCH A MANNER THAT THE SECOND
; ROM NEED NOT BE PLUGGED INTO THE BOARD IF ONLY THE BASIC MONITOR
; FUNCTIONS ARE REQUIRED. HOWEVER IF THE PAPER TAPE FUCTIONS ARE DESIRED
; BOTH ROMS ARE REQUIRED.
;
; THIS PROGRAM EXPECTS TO RUN IN THE FIRST 2K OF ADDRESS SPACE.
; IF, FOR SOME REASON, THE PROGRAM IS RE-ORG'ED, CARE SHOULD
; BE TAKEN TO MAKE SURE THAT THE TRANSFER INSTRUCTIONS FOR RST 1
; AND RST 7 ARE ADJUSTED APPROPRIATELY.
;
; THE PROGRAM ALSO EXPECTS THAT RAM LOCATIONS 3C00H TO 3C3FH,
; INCLUSIVE, ARE RESERVED FOR THE PROGRAM'S OWN USE.  THESE
; LOCATIONS MAY BE ALTERED, HOWEVER, BY CHANGING THE EQU'ED
; SYMBOL "DATA" AS DESIRED.
;
; LIST OF FUNCTIONS
; ==== == =========
;
;     ********
;     1 ST ROM
;     ********
;
;     GETCM
;     -----
;
;     DCMD
;     GCMD
;     ICMD
;     MCMD
;     RCMD
;     SCMD
;     WCMD
;     XCMD
;     -----
;
;     ADRD
;     ADROUT
;     BREAK
;     CI
;     CNVBN
;     CO
;     CROUT
;     ECHO
;     ERROR
;     FRET
;     GETCH
;     GETHX
;     GETNM
;     HILO
```

```
                ;       INUST
                ;       NMOUT
                ;       PRVAL
                ;       REGDS
                ;       RGADR
                ;       RSTTF
                ;       SRET
                ;       STHF0
                ;       STHLF
                ;       VALDG
                ;       VALDL
                ;       -----
                ;
                ;       ********
                ;       2 ND ROM
                ;       ********
                ;
                ;       RCMD
                ;       WCMD
                ;       ----
                ;
                ;       BYTE
                ;       DELAY
                ;       LEAD
                ;       PADR
                ;       PBYTE
                ;       PEOF
                ;       PEOL
                ;       PO
                ;       RI
                ;       RICH
                ;       -----
                ;
0000                    ORG     0H
                ;
                ;*****************************************************************
                ;
                ;
                ;                       MONITOR EQUATES
                ;
                ;
                ;*****************************************************************
                ;
                ;
001B            BRCHR EQU       1BH     ; CODE FOR BREAK CHARACTER (ESCAPE)
3C3D            BRLOC EQU       3C3DH   ; LOCATION OF USER BRANCH INSTRUCTION IN RAM
03FA            BRTAB EQU       3FAH    ; LOCATION OF START OF BRANCH TABLE IN ROM
0025            CMD   EQU       025H    ; COMMAND INSTRUCTION FOR USART INITIALIZATION
00ED            CNCTL EQU       0EDH    ; CONSOLE (USART) CONTROL PORT
00EC            CNIN  EQU       0ECH    ; CONSOLE INPUT PORT
00EC            CNOUT EQU       0ECH    ; CONSOLE OUTPUT PORT
```

```
00ED            CONST EQU     0EDH    ; CONSOLE STATUS INPUT PORT
000D            CR    EQU     0DH     ; CODE FOR CARRIAGE RETURN
3C00            DATA  EQU     15*1024 ; START OF MONITOR RAM USAGE
001B            ESC   EQU     1BH     ; CODE FOR ESCAPE CHARACTER
000F            HCHAR EQU     0FH     ; MASK TO SELECT LOWER HEX CHAR FROM BYTE
00FF            INVRT EQU     0FFH    ; MASK TO INVERT HALF BYTE FLAG
000A            LF    EQU     0AH     ; CODE FOR LINE FEED
                ;LSGNON       EQU     ---     ; LENGTH OF SIGNON MESSAGE - DEFINED LATER
00CF            MODE  EQU     0CFH    ; MODE SET FOR USART INITIALIZATION
                ;MSTAK        EQU     ---     ; START OF MONITOR STACK - DEFINED LATER
                ;NCMDS        EQU     ---     ; NUMBER OF VALID COMMANDS
000F            NEWLN EQU     0FH     ; MASK FOR CHECKING MEMORY ADDR DISPLAY
007F            PRTY0 EQU     07FH    ; MASK TO CLEAR PARITY BIT FROM CONSOLE CHAR
3C2E            REGS  EQU     DATA+64-18      ; START OF REGISTER SAVE AREA
0002            RBR   EQU     2       ; MASK TO TEST RECEIVER STATUS
0038            RSTU  EQU     38H     ; TRANSFER LOCATION FOR RST 7 INSTRUCTION
                ;RTABS        EQU     ---     ; SIZE OF ENTRY IN RTAB TABLE
001B            TERM  EQU     1BH     ; CODE FOR ICMD TERMINATING CHARACTER (ESCAPE)
0001            TRDY  EQU     1       ; MASK TO TEST TRANSMITTER STATUS
00FF            UPPER EQU     0FFH    ; DENOTES UPPER HALF OF BYTE IN ICMD
0004            TXBE  EQU     04H     ; USART TRANSMITTER BUFFER EMPTY
0027            TTYADV        EQU     27H     ; TTY READER ADVANCE COMMAND
0083            ONEMS EQU     131     ; 1 MILLISECOND CONSTANT
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;                       MONITOR MACROS
                ;
                ;
                ;*****************************************************************
                ;
                ;
      1         TRUE  MACRO   WHERE   ; BRANCH IF FUNCTION RETURNS TRUE (SUCCESS)
      1               JC      WHERE
                      ENDM
                ;
      1         FALSE MACRO   WHERE   ; BRANCH IF FUNCTION RETURNS FALSE (FAILURE)
      1               JNC     WHERE
                      ENDM
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;                       USART INITIALIZATION CODE
                ;
                ;
                ;*****************************************************************
                ;
```

```
                ;
                ;   THE USART IS ASSUMED TO COME UP IN THE RESET POSITION (THIS
                ;   FUNCTION IS TAKEN CARE OF BY THE HARDWARE).  THE USART WILL
                ;   BE INITIALIZED IN THE SAME WAY FOR EITHER A TTY OR CRT
                ;   INTERFACE.  THE FOLLOWING PARAMETERS ARE USED:
                ;
                ;      MODE INSTRUCTION
                ;      ==== ===========
                ;
                ;      2 STOP BITS
                ;      PARITY DISABLED
                ;      8 BIT CHARACTERS
                ;      BAUD RATE FACTOR OF 64
                ;
                ;      COMMAND INSTRUCTION
                ;      ======= ===========
                ;
                ;      NO HUNT MODE
                ;      NOT(RTS) FORCED TO 0
                ;      RECEIVE ENABLED
                ;      TRANSMIT ENABLED
                ;
0000  3ECE      START:  MVI     A,MODE
0002  D3ED              OUT     CNCTL   ; OUTPUT MODE SET TO USART
0004  C3B202            JMP     INUST   ; BRANCH TO COMPLETE USART INITIALIZATION
0007  00                NOP             ; FILLER
                ;
                ;*****************************************************************
                ;
                ;
                ;                       RESTART ENTRY POINT
                ;
                ;
                ;*****************************************************************
                ;
                ;
0008            GO:
0008  22343C            SHLD    LSAVE   ; SAVE HL REGISTERS
000B  E1                POP     H       ; GET TOP OF STACK ENTRY
000C  22363C            SHLD    PSAVE   ; ASSUME THIS IS LAST P COUNTER
000F  F5                PUSH    PSW     ; SAVE A,F/F'S
0010  210200            LXI     H,2     ; SET HL TO 2 SO THAT STACK POINTER SAVED CORRECTLY
0013  39                DAD     SP      ; GET STACK POINTER VALUE
0014  22383C            SHLD    SSAVE   ; SAVE USER'S STACK POINTER
0017  F1                POP     PSW     ; RESTORE A,F/F'S
0018  31343C            LXI     SP,ASAVE+1      ; NEW VALUE FOR STACK POINTER
001B  C3B101            JMP     ADROUT
                ;
                ;*************************************************************
                ;
                ;
```

```
                        ;                       PRINT SIGNON MESSAGE
                        ;
                        ;
                        ;*************************************************************
                        ;
                        ;
001E                    SOMSG:
001E    219503          LXI     H,SGNON ; GET ADDRESS OF SIGNON MESSAGE
0021    0611            MVI     B,LSGNON        ; COUNTER FOR CHARACTERS IN MESSAGE
0023                    MSGL:
0023    4E              MOV     C,M     ; FETCH NEXT CHAR TO C REG
0024    CDE801          CALL    CO      ; SEND IT TO THE CONSOLE
0027    23              INX     H       ; POINT TO NEXT CHARACTER
0028    05              DCR     B       ; DECREMENT BYTE COUNTER
0029    C22300          JNZ     MSGL    ; RETURN FOR NEXT CHARACTER
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;                       COMMAND RECOGNIZING ROUTINE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ; FUNCTION: GETCM
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: GETCH,ECHO,ERROR
                        ; DESTROYS: A,B,C,H,L,F/F'S
                        ; DESCRIPTION: GETCM RECEIVES AN INPUT CHARACTER FROM THE USER
                        ;               AND ATTEMPTS TO LOCATE THIS CHARACTER IN ITS COMMAND
                        ;               CHARACTER TABLE.  IF SUCCESSFUL, THE ROUTINE
                        ;               CORRESPONDING TO THIS CHARACTER IS SELECTED FROM
                        ;               A TABLE OF COMMAND ROUTINE ADDRESSES, AND CONTROL
                        ;               IS TRANSFERRED TO THIS ROUTINE.  IF THE CHARACTER
                        ;               DOES NOT MATCH ANY ENTRIES, CONTROL IS PASSED TO
                        ;               THE ERROR HANDLER.
                        ;
002C                    GETCM:
002C    312E3C          LXI     SP,MSTAK        ; ALWAYS WANT TO RESET STACK PTR TO MONITOR
                                                ; /STARTING VALUE SO ROUTINES NEEDN'T CLEAN UP
002F    0E2E            MVI     C,'.'   ; PROMPT CHARACTER TO C
0031    CDF901          CALL    ECHO    ; SEND PROMPT CHARACTER TO USER TERMINAL
0034    C33C00          JMP     GTC03   ; WANT TO LEAVE ROOM FOR RST BRANCH
                        ;
0038                    ORG     RSTU    ; ORG TO RST TRANSFER LOCATION
0038    C33D3C          JMP     USRBR   ; JUMP TO USER BRANCH LOCATION
003B    00              NOP             ; FILLER
                        ;
003C                    GTC03:
```

```
003C   CD2002          CALL    GETCH   ; GET COMMAND CHARACTER TO A
003F   CDF901          CALL    ECHO    ; ECHO CHARACTER TO USER
0042   79              MOV     A,C     ; PUT COMMAND CHARACTER INTO ACCUMULATOR
0043   010800          LXI     B,NCMDS ; C CONTAINS LOOP AND INDEX COUNT
0046   21B803          LXI     H,CTAB  ; HL POINTS INTO COMMAND TABLE
0049                GTC05:
0049   BE              CMP     M       ; COMPARE TABLE ENTRY AND CHARACTER
004A   CA5500          JZ      GTC10   ; BRANCH IF EQUAL - COMMAND RECOGNIZED
004D   23              INX     H       ; ELSE, INCREMENT TABLE POINTER
004E   0D              DCR     C       ; DECREMENT LOOP COUNT
004F   C24900          JNZ     GTC05   ; BRANCH IF NOT AT TABLE END
0052   C31202          JMP     ERROR   ; ELSE, COMMAND CHARACTER IS ILLEGAL
0055                GTC10:
0055   21A603          LXI     H,CADR  ; IF GOOD COMMAND, LOAD ADDRESS OF TABLE
                                       ; /OF COMMAND ROUTINE ADDRESSES
0058   09              DAD     B       ; ADD WHAT IS LEFT OF LOOP COUNT
0059   09              DAD     B       ; ADD AGAIN - EACH ENTRY IN CADR IS 2 BYTES LONG
005A   7E              MOV     A,M     ; GET LSP OF ADDRESS OF TABLE ENTRY TO A
005B   23              INX     H       ; POINT TO NEXT BYTE IN TABLE
005C   66              MOV     H,M     ; GET MSP OF ADDRESS OF TABLE ENTRY TO H
005D   6F              MOV     L,A     ; PUT LSP OF ADDRESS OF TABLE ENTRY INTO L
005E   E9              PCHL            ; NEXT INSTRUCTION COMES FROM COMMAND ROUTINE
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ;            COMMAND IMPLEMENTING ROUTINES
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION: DCMD
                    ; INPUTS: NONE
                    ; OUTPUTS: NONE
                    ; CALLS: ECHO,NMOUT,HILO,GETCM,CROUT,GETNM
                    ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                    ; DESCRIPTION: DCMD IMPLEMENTS THE DISPLAY MEMORY (D) COMMAND
                    ;
005F                DCMD:
005F   0E02            MVI     C,2     ; GET TWO NUMBERS FROM INPUT STREAM
0061   CD5B02          CALL    GETNM
0064   D1              POP     D       ; ENDING ADDRESS TO DE
0065   E1              POP     H       ; STARTING ADDRESS TO HL
0066                DCM05:
0066   CDF301          CALL    CROUT   ; ECHO CARRIAGE RETURN/LINE FEED
0069   CDA801          CALL    ADRD    ; DISPLAY ADDRESS
006C                DCM10:
006C   0E20            MVI     C,' '
006E   CDF901          CALL    ECHO    ; USE BLANK AS SEPARATOR
```

```
0071   7E            MOV     A,M     ; GET CONTENTS OF NEXT MEMORY LOCATION
0072   CDC202        CALL    NMOUT   ; DISPLAY CONTENTS
0075   CDC201        CALL    BREAK   ; SEE IF USER WANTS OUT
     1          +     TRUE    EXIT    ; IF SO, BRANCH TO EXIT
0078 1 DA1702   +     JC      EXIT
007B   CDA002        CALL    HILO    ; SEE IF ADDRESS OF DISPLAYED LOCATION IS
                                     ; /GREATER THAN OR EQUAL TO ENDING ADDRESS
     1          +     TRUE    EXIT    ; EXIT IF NO MORE TO DISPLAY
007E 1 DA1702   +     JC      EXIT
0081   23            INX     H       ; IF MORE TO GO, POINT TO NEXT LOC TO DISPLAY
0082   7D            MOV     A,L     ; GET LOW ORDER BITS OF NEW ADDRESS
0083   E60F          ANI     NEWLN   ; SEE IF LAST HEX DIGIT OF ADDRESS DENOTES
                                     ; /START OF NEW LINE
0085   C26C00        JNZ     DCM10   ; NO - NOT AT END OF LINE
0088   C36600        JMP     DCM05   ; YES - START NEW LINE WITH ADDRESS
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: GCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ERROR,GETHX,RSTTF
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: GCMD IMPLEMENTS THE BEGIN EXECUTION (G) COMMAND.
                ;
008B            GCMD:
008B   CD2702        CALL    GETHX   ; GET ADDRESS (IF PRESENT) FROM INPUT STREAM
     1          +     FALSE   GCM05   ; BRANCH IF NO NUMBER PRESENT
008E 1 D2A000   +     JNC     GCM05
0091   7A            MOV     A,D     ; ELSE, GET TERMINATOR
0092   FE0D          CPI     CR      ; SEE IF CARRIAGE RETURN
0094   C21202        JNZ     ERROR   ; ERROR IF NOT PROPERLY TERMINATED
0097   21363C        LXI     H,PSAVE ; WANT NUMBER TO REPLACE SAVE PGM COUNTER
009A   71            MOV     M,C
009B   23            INX     H
009C   70            MOV     M,B
009D   C3A600        JMP     GCM10
00A0            GCM05:
00A0   7A            MOV     A,D     ; IF NO STARTING ADDRESS, MAKE SURE THAT
00A1   FE0D          CPI     CR      ; /CARRIAGE RETURN TERMINATED COMMAND
00A3   C21202        JNZ     ERROR   ; ERROR IF NOT
00A6 .          GCM10:
00A6   C32703        JMP     RSTTF   ; RESTORE REGISTERS AND BEGIN EXECUTION

                ;
                ;
                ;*****************************************************************
                ;
                ;
```

```
                        ; FUNCTION: ICMD
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: ERROR,ECHO,GETCH,VALDL,VALDG,CNVBN,STHLF,GETNM,CROUT
                        ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                        ; DESCRIPTION: ICMD IMPLEMENTS THE INSERT CODE INTO MEMORY (I) COMMAND.
                        ;
00A9                    ICMD:
00A9   0E01                     MVI     C,1
00AB   CD5B02                   CALL    GETNM   ; GET SINGLE NUMBER FROM INPUT STREAM
00AE   3EFF                     MVI     A,UPPER
00B0   323A3C                   STA     TEMP    ; TEMP WILL HOLD THE UPPER/LOWER HALF BYTE FLAG
00B3   D1                       POP     D       ; ADDRESS OF START TO DE
00B4                    ICM05:
00D4   CD2002                   CALL    GETCH   ; GET A CHARACTER FROM INPUT STREAM
00B7   CDF901                   CALL    ECHO    ; ECHO IT
00BA   79                       MOV     A,C     ; PUT CHARACTER BACK INTO A
00BB   FE1B                     CPI     TERM    ; SEE IF CHARACTER IS A TERMINATING CHARACTER
00BD   CAE900                   JZ      ICM25   ; IF SO, ALL DONE ENTERING CHARACTERS
00C0   CD8203                   CALL    VALDL   ; ELSE, SEE IF VALID DELIMITER
     1                  +       TRUE    ICM05   ; IF SO SIMPLY IGNORE THIS CHARACTER
00C3 1 DAB400           +       JC      ICM05
00C6   CD6703                   CALL    VALDG   ; ELSE, CHECK TO SEE IF VALID HEX DIGIT
     1                  +       FALSE   ICM20   ; IF NOT, BRANCH TO HANDLE ERROR CONDITION
00C9 1 D2E300           +       JNC     ICM20
00CC   CDDF01                   CALL    CNVBN   ; CONVERT DIGIT TO BINARY
00CF   4F                       MOV     C,A     ; MOVE RESULT TO C
00D0   CD4803                   CALL    STHLF   ; STORE IN APPROPRIATE HALF WORD
00D3   3A3A3C                   LDA     TEMP    ; GET HALF BYTE FLAG
00D6   B7                       ORA     A       ; SET F/F'S
00D7   C2DB00                   JNZ     ICM10   ; BRANCH IF FLAG SET FOR UPPER
00DA   13                       INX     D       ; IF LOWER, INC ADDRESS OF BYTE TO STORE IN
00DB                    ICM10:
00DB   EEFF                     XRI     INVRT   ; TOGGLE STATE OF FLAG
00DD   323A3C                   STA     TEMP    ; PUT NEW VALUE OF FLAG BACK
00E0   C3B400                   JMP     ICM05   ; PROCESS NEXT DIGIT
00E3                    ICM20:
00E3   CD3D03                   CALL    STHF0   ; ILLEGAL CHARACTER
00E6   C31202                   JMP     ERROR   ; MAKE SURE ENTIRE BYTE FILLED THEN ERROR
00E9                    ICM25:
00E9   CD3D03                   CALL    STHF0   ; HERE FOR ESCAPE CHARACTER - INPUT IS DONE
00EC   C31702                   JMP     EXIT
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: MCMD
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: GETCM,HILO,GETNM
```

```
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: MCMD IMPLEMENTS THE MOVE DATA IN MEMORY (M) COMMAND.
                ;
00EF            MCMD:
00EF   0E03             MVI     C,3
00F1   CD5B02           CALL    GETNM   ; GET 3 NUMBERS FROM INPUT STREAM
00F4   C1               POP     B       ; DESTINATION ADDRESS TO BC
00F5   E1               POP     H       ; ENDING ADDRESS TO HL
00F6   D1               POP     D       ; STARTING ADDRESS TO DE
00F7            MCM05:
00F7   E5               PUSH    H       ; SAVE ENDING ADDRESS
00F8   62               MOV     H,D
00F9   6B               MOV     L,E     ; SOURCE ADDRESS TO HL
00FA   7E               MOV     A,M     ; GET SOURCE BYTE
00FB   60               MOV     H,B
00FC   69               MOV     L,C     ; DESTINATION ADDRESS TO HL
00FD   77               MOV     M,A     ; MOVE BYTE TO DESTINATION
00FE   03               INX     B       ; INCREMENT DESTINATION ADDRESS
00FF   78               MOV     A,B
0100   B1               ORA     C       ; TEST FOR DESTINATION ADDRESS OVERFLOW
0101   CA2C00           JZ      GETCM   ; IF SO, CAN TERMINATE COMMAND
0104   13               INX     D       ; INCREMENT SOURCE ADDRESS
0105   E1               POP     H       ; ELSE, GET BACK ENDING ADDRESS
0106   CDA002           CALL    HILO    ; SEE IF ENDING ADDR>=SOURCE ADDR
     1         +        FALSE   GETCM   ; IF NOT, COMMAND IS DONE
0109 1 D22C00  +        JNC     GETCM
010C   C3F700           JMP     MCM05   ; MOVE ANOTHER BYTE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: SCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: GETHX,GETCM,NMOUT,ECHO
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: SCMD IMPLEMENTS THE SUBSTITUTE INTO MEMORY (S) COMMAND.
                ;
010F            SCMD:
010F   CD2702           CALL    GETHX   ; GET A NUMBER, IF PRESENT, FROM INPUT
0112   C5               PUSH    B
0113   E1               POP     H       ; GET NUMBER TO HL - DENOTES MEMORY LOCATION
0114            SCM05:
0114   7A               MOV     A,D     ; GET TERMINATOR
0115   FE20             CPI     ' '     ; SEE IF SPACE
0117   CA1F01           JZ      SCM10   ; YES - CONTINUE PROCESSING
011A   FE2C             CPI     ','     ; ELSE, SEE IF COMMA
011C   C22C00           JNZ     GETCM   ; NO - TERMINATE COMMAND
011F            SCM10:
011F   7E               MOV     A,M     ; GET CONTENTS OF SPECIFIED LOCATION TO A
```

```
0120   CDC202          CALL    NMOUT   ; DISPLAY CONTENTS ON CONSOLE
0123   0E2D            MVI     C,'-'
0125   CDF901          CALL    ECHO    ; USE DASH FOR SEPARATOR
0128   CD2702          CALL    GETHX   ; GET NEW VALUE FOR MEMORY LOCATION, IF ANY
     1          +       FALSE   SCM15   ; IF NO VALUE PRESENT, BRANCH
012B 1 D22F01   +       JNC     SCM15
012E   71              MOV     M,C     ; ELSE, STORE LOWER 8 BITS OF NUMBER ENTERED
012F             SCM15:
012F   23              INX     H       ; INCREMENT ADDRESS OF MEMORY LOCATION TO VIEW
0130   C31401          JMP     SCM05
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: XCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: GETCH,ECHO,REGDS,GETCM,ERROR,RGADR,NMOUT,CROUT,GETHX
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: XCMD IMPLEMENTS THE REGISTER EXAMINE AND CHANGE (X)
                ;              COMMAND.
                ;
0133             XCMD:
0133   CD2002          CALL    GETCH   ; GET REGISTER IDENTIFIER
0136   CDF901          CALL    ECHO    ; ECHO IT
0139   79              MOV     A,C
013A   FE0D            CPI     CR
013C   C24501          JNZ     XCM05   ; BRANCH IF NOT CARRIACE RETURN
013F   CDDF02          CALL    REGDS   ; ELSE, DISPLAY REGISTER CONTENTS
0142   C32C00          JMP     GETCM   ; THEN TERMINATE COMMAND
0145             XCM05:
0145   4F              MOV     C,A     ; GET REGISTER IDENTIFIER TO C
0146   CD1003          CALL    RGADR   ; CONVERT IDENTIFIER INTO RTAB TABLE ADDR
0149   C5              PUSH    B
014A   E1              POP     H       ; PUT POINTER TO REGISTER ENTRY INTO HL
014B   0E20            MVI     C,' '
014D   CDF901          CALL    ECHO    ; ECHO SPACE TO USER
0150   79              MOV     A,C
0151   323A3C          STA     TEMP    ; PUT SPACE INTO TEMP AS DELIMITER
0154             XCM10:
0154   3A3A3C          LDA     TEMP    ; GET TERMINATOR
0157   FE20            CPI     ' '     ; SEE IF A BLANK
0159   CA6101          JZ      XCM15   ; YES - GO CHECK POINTER INTO TABLE
015C   FE2C            CPI     ','     ; NO - SEE IF COMMA
015E   C22C00          JNZ     GETCM   ; NO - MUST BE CARRIAGE RETURN TO END COMMAND
0161             XCM15:
0161   7E              MOV     A,M
0162   B7              ORA     A       ; SET F/F'S
0163   CA1702          JZ      EXIT    ; BRANCH IF AT END OF TABLE
0166   E5              PUSH    H       ; PUT POINTER ON STACK
```

```
0167   5E            MOV     E,M
0168   163C          MVI     D,DATA SHR 8    ; FETCH ADDRESS OF SAVE LOCATION FROM
016A   23            INX     H               ; /TABLE
016B   46            MOV     B,M     ; FETCH LENGTH FLAG FROM TABLE
016C   D5            PUSH    D       ; SAVE ADDRESS OF SAVE LOCATION
016D   D5            PUSH    D
016E   E1            POP     H       ; MOVE ADDRESS TO HL
016F   C5            PUSH    B       ; SAVE LENGTH FLAG
0170   7E            MOV     A,M     ; GET 8 BITS OF REGISTER FROM SAVE LOCATION
0171   CDC202        CALL    NMOUT   ; DISPLAY IT
0174   F1            POP     PSW     ; GET BACK LENGTH FLAG
0175   F5            PUSH    PSW     ; SAVE IT AGAIN
0176   B7            ORA     A       ; SET F/F'S
0177   CA7F01        JZ      XCM20   ; IF 8 BIT REGISTER, NOTHING MORE TO DISPLAY
017A   2B            DCX     H       ; ELSE, FOR 16 BIT REGISTER, GET LOWER 8 BITS
017B   7E            MOV     A,M
017C   CDC202        CALL    NMOUT   ; DISPLAY THEM
017F             XCM20:
017F   0E2D          MVI     C,'-'
0181   CDF901        CALL    ECHO    ; USE DASH AS SEPARATOR
0184   CD2702        CALL    GETHX   ; SEE IF THERE IS A VALUE TO PUT INTO REGISTER
     1         +     FALSE   XCM30   ; NO - GO CHECK FOR NEXT REGISTER
0187 1 D29F01  +     JNC     XCM30
018A   7A            MOV     A,D
018B   323A3C        STA     TEMP    ; ELSE, SAVE THE TERMINATOR FOR NOW
018E   F1            POP     PSW     ; GET BACK LENGTH FLAG
018F   E1            POP     H       ; PUT ADDRESS OF SAVE LOCATION INTO HL
0190   B7            ORA     A       ; SET F/F'S
0191   CA9601        JZ      XCM25   ; IF 8 BIT REGISTER, BRANCH
0194   70            MOV     M,B     ; SAVE UPPER 8 BITS
0195   2B            DCX     H       ; POINT TO SAVE LOCATION FOR LOWER 8 BITS
0196             XCM25:
0196   71            MOV     M,C     ; STORE ALL OF 8 BIT OR LOWER 1/2 OF 16 BIT REG
0197             XCM27:
0197   110300        LXI     D,RTABS ; SIZE OF ENTRY IN RTAB TABLE
019A   E1            POP     H       ; POINTER INTO REGISTER TABLE RTAB
019B   19            DAD     D       ; ADD ENTRY SIZE TO POINTER
019C   C35401        JMP     XCM10   ; DO NEXT REGISTER
019F             XCM30:
019F   7A            MOV     A,D     ; GET TERMINATOR
01A0   323A3C        STA     TEMP    ; SAVE IN MEMORY
01A3   D1            POP     D       ; CLEAR STACK OF LENGTH FLAG AND ADDRESS
01A4   D1            POP     D       ; /OF SAVE LOCATION
01A5   C39701        JMP     XCM27   ; GO INCREMENT REGISTER TABLE POINTER
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;                       UTILITY ROUTINES
                ;
```

```
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION ADRD
                        ; INPUTS: HL - ADDRESS TO BE DISPLAYED
                        ; OUTPUTS: NONE
                        ; CALLS: NMOUT
                        ; DESTROYS: A
                        ; DESCRIPTION: ADRD OUTPUTS TO THE CONSOLE THE ADDRESS
                        ;              CONTAINED IN THE H,L REGISTERS.
                        ;
01A8                    ADRD:
01A8  7C                      MOV     A,H     ; DISPLAY FIRST HALF OF ADDRESS
01A9  CDC202                  CALL    NMOUT
01AC  7D                      MOV     A,L     ; DISPLAY SECOND HALF OF ADDRESS
01AD  CDC202                  CALL    NMOUT
01B0  C9                      RET             ; RETURN TO CALLING ROUTINE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION ADROUT
                        ; INPUTS: USER REGISTERS ON THE STACK
                        ; OUTPUTS: NOTHING
                        ; CALLS: ECHO,ADRD
                        ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                        ; DESCRIPTION: ADROUT SAVES THE USER REGISTERS AND OUTPUTS TO THE
                        ;              CONSOLE THE USER P COUNTER AFTER A RST 1 INSTRUCTION.
                        ;
01B1                    ADROUT:
01B1  F5                      PUSH    PSW     ; SAVE A AND FLAGS
01B2  C5                      PUSH    B       ; SAVE B AND C
01B3  D5                      PUSH    D       ; SAVE D AND E
01B4  0E23                    MVI     C,'#'
01B6  CDF901                  CALL    ECHO    ; OUTPUT '#'
01B9  2A363C                  LHLD    PSAVE   ; LOAD USER P COUNTER
01BC  CDA801                  CALL    ADRD    ; DISPLAY ADDRESS
01BF  C31702                  JMP     EXIT    ; GET NEW COMMAND
                        ;
                        ;
                        ; FUNCTION: BREAK
                        ; INPUTS: NONE
                        ; OUTPUTS: CARRY - 1 IF ESCAPE CHARACTER INPUT
                        ;                - 0 IF ANY OTHER CHARACTER OR NO CHARACTER PENDING
                        ; CALLS: NOTHING
                        ; DESTROYS: A,F/F'S
```

```
                ; DESCRIPTION: BREAK IS USED TO SENSE AN ESCAPE CHARACTER FROM
                ;              THE USER.  IF NO CHARACTER IS PENDING, OR IF THE
                ;              PENDING CHARACTER IS NOT THE ESCAPE, THEN A FAILURE
                ;              RETURN (CARRY=0) IS TAKEN.  IN THIS CASE, THE
                ;              PENDING CHARACTER (IF ANY) IS LOST.  IF THE PENDING
                ;              CHARACTER IS AN ESCAPE CHARACTER, BREAK TAKES A SUCCESS
                ;              RETURN (CARRY=1).
                ;
 01C2           BREAK:
 01C2   DBED          IN      CONST   ; GET CONSOLE STATUS
 01C4   E602          ANI     RBR     ; SEE IF CHARACTER PENDING
 01C6   CA1D02        JZ      FRET    ; NO - TAKE FAILURE RETURN
 01C9   DBEC          IN      CNIN    ; YES - PICK UP CHARACTER
 01CB   E67F          ANI     PRTY0   ; STRIP OFF PARITY BIT
 01CD   FE1B          CPI     BRCHR   ; SEE IF BREAK CHARACTER
 01CF   CA3B03        JZ      SRET    ; YES - SUCCESS RETURN
 01D2   C31D02        JMP     FRET    ; NO - FAILURE RETURN - CHARACTER LOST
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: CI
                ; INPUTS: NONE
                ; OUTPUTS: A - CHARACTER FROM CONSOLE
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: CI WAITS UNTIL A CHARACTER HAS BEEN ENTERED AT THE
                ;              CONSOLE AND THEN RETURNS THE CHARACTER, VIA THE A
                ;              REGISTER, TO THE CALLING ROUTINE.  THIS ROUTINE
                ;              IS CALLED BY THE USER VIA A JUMP TABLE IN RAM.
                ;
 01D5           CI:
 01D5   DBED          IN      CONST   ; GET STATUS OF CONSOLE
 01D7   E602          ANI     RBR     ; CHECK FOR RECEIVER BUFFER READY
 01D9   CAD501        JZ      CI      ; NOT YET - WAIT
 01DC   DBEC          IN      CNIN    ; READY SO GET CHARACTER
 01DE   C9            RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: CNVBN
                ; INPUTS: C - ASCII CHARACTER '0'-'9' OR 'A'-'F'
                ; OUTPUTS: A - 0 TO F HEX
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: CNVBN CONVERTS THE ASCII REPRESENTATION OF A HEX
                ;              CHARACTER INTO ITS CORRESPONDING BINARY VALUE.  CNVBN
                ;              DOES NOT CHECK THE VALIDITY OF ITS INPUT.
```

```
                        ;
01DF                    CNVBN:
01DF    79                      MOV     A,C
01E0    D630                    SUI     '0'     ; SUBTRACT CODE FOR '0' FROM ARGUMENT
01E2    FE0A                    CPI     10      ; WANT TO TEST FOR RESULT OF 0 TO 9
01E4    F8                      RM              ; IF SO, THEN ALL DONE
01E5    D607                    SUI     7       ; ELSE, RESULT BETWEEN 17 AND 23 DECIMAL
01E7    C9                      RET             ; SO RETURN AFTER SUBTRACTING BIAS OF 7
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: CO
                        ; INPUTS: C - CHARACTER TO OUTPUT TO CONSOLE
                        ; OUTPUTS: C - CHARACTER OUTPUT TO CONSOLE
                        ; CALLS: NOTHING
                        ; DESTROYS: A,F/F'S
                        ; DESCRIPTION: CO WAITS UNTIL THE CONSOLE IS READY TO ACCEPT A CHARACTER
                        ;               AND THEN SENDS THE INPUT ARGUMENT TO THE CONSOLE.
                        ;
01E8                    CO:
01E8    DBED                    IN      CONST   ; GET STATUS OF CONSOLE
01EA    E601                    ANI     TRDY    ; SEE IF TRANSMITTER READY
01EC    CAE801                  JZ      CO      ; NO - WAIT
01EF    79                      MOV     A,C     ; ELSE, MOVE CHARACTER TO A REGISTER FOR OUTPUT
01F0    D3EC                    OUT     CNOUT   ; SEND TO CONSOLE
01F2    C9                      RET
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION CROUT
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: ECHO
                        ; DESTROYS: A,B,C,F/F'S
                        ; DESCRIPTION: CROUT SENDS A CARRIAGE RETURN (AND HENCE A LINE
                        ;               FEED) TO THE CONSOLE.
                        ;
01F3                    CROUT:
01F3    0E0D                    MVI     C,CR
01F5    CDF901                  CALL    ECHO    ; OUTPUT CARRIAGE RETURN TO USER TERMINAL
01F8    C9                      RET
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: ECHO
```

```
                ; INPUTS: C - CHARACTER TO ECHO TO TERMINAL
                ; OUTPUTS: C - CHARACTER ECHOED TO TERMINAL
                ; CALLS: CO
                ; DESTROYS: A,B,F/F'S
                ; DESCRIPTION: ECHO TAKES A SINGLE CHARACTER AS INPUT AND, VIA
                ;              THE MONITOR, SENDS THAT CHARACTER TO THE USER
                ;              TERMINAL.  A CARRIAGE RETURN IS ECHOED AS A CARRIAGE
                ;              RETURN LINE FEED, AND AN ESCAPE CHARACTER IS ECHOED AS $.
                ;
 01F9           ECHO:
 01F9   41            MOV     B,C     ; SAVE ARGUMENT
 01FA   3E1B          MVI     A,ESC
 01FC   B8            CMP     B       ; SEE IF ECHOING AN ESCAPE CHARACTER
 01FD   C20202        JNZ     ECH05   ; NO - BRANCH
 0200   0E24          MVI     C,'$'   ; YES - ECHO AS $
 0202           ECH05:
 0202   CDE801       CALL    CO      ; DO OUTPUT THROUGH MONITOR
 0205   3E0D          MVI     A,CR
 0207   B8            CMP     B       ; SEE IF CHARACTER ECHOED WAS A CARRIAGE RETURN
 0208   C21002        JNZ     ECH10   ; NO - NO NEED TO TAKE SPECIAL ACTION
 020B   0E0A          MVI     C,LF    ; YES - WANT TO ECHO LINE FEED, TOO
 020D   CDE801        CALL    CO
 0210           ECH10:
 0210   48            MOV     C,B     ; RESTORE ARGUMENT
 0211   C9            RET
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: ERROR
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ECHO,CROUT,GETCM
                ; DESTROYS: A,B,C,F/F'S
                ; DESCRIPTION: ERROR PRINTS THE ERROR CHARACTER (CURRENTLY AN ASTERISK)
                ;              ON THE CONSOLE, FOLLOWED BY A CARRIAGE RETURN-LINE FEED,
                ;              AND THEN RETURNS CONTROL TO THE COMMAND RECOGNIZER.
                ;
 0212           ERROR:
 0212   0E23          MVI     C,'#'
 0214   CDF901        CALL    ECHO    ; SEND # TO CONSOLE
 0217           EXIT:
 0217   CDF301        CALL    CROUT   ; SKIP TO BEGINNING OF NEXT LINE
 021A   C32C00        JMP     GETCM   ; TRY AGAIN FOR ANOTHER COMMAND
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: PRET
```

```
                ; INPUTS: NONE
                ; OUTPUTS: CARRY - ALWAYS 0
                ; CALLS: NOTHING
                ; DESTROYS: CARRY
                ; DESCRIPTION: FRET IS JUMPED TO BY ANY ROUTINE THAT WISHES TO
                ;              INDICATE FAILURE ON RETURN.  FRET SETS THE CARRY
                ;              FALSE, DENOTING FAILURE, AND THEN RETURNS TO THE
                ;              CALLER OF THE ROUTINE INVOKING FRET.
                ;
  021D          FRET:
  021D  37            STC             ; FIRST SET CARRY TRUE
  021E  3F            CMC             ; THEN COMPLEMENT IT TO MAKE IT FALSE
  021F  C9            RET             ; RETURN APPROPRIATELY
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: GETCH
                ; INPUTS: NONE
                ; OUTPUTS: C - NEXT CHARACTER IN INPUT STREAM
                ; CALLS: CI
                ; DESTROYS: A,C,F/F'S
                ; DESCRIPTION: GETCH RETURNS THE NEXT CHARACTER IN THE INPUT STREAM
                ;              TO THE CALLING PROGRAM.
                ;
  0220          GETCH:
  0220  CDD501        CALL    CI      ; GET CHARACTER FROM TERMINAL
  0223  E67F          ANI     PRTY0   ; TURN OFF PARITY BIT IN CASE SET BY CONSOLE
  0225  4F            MOV     C,A     ; PUT VALUE IN C REGISTER FOR RETURN
  0226  C9            RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: GETHX
                ; INPUTS: NONE
                ; OUTPUTS: BC - 16 BIT INTEGER
                ;          D - CHARACTER WHICH TERMINATED THE INTEGER
                ;          CARRY - 1 IF FIRST CHARACTER NOT DELIMITER
                ;                - 0 IF FIRST CHARACTER IS DELIMITER
                ; CALLS: GETCH,ECHO,VALDL,VALDG,CNVBN,ERROR
                ; DESTROYS: A,B,C,D,E,F/F'S
                ; DESCRIPTION: GETHX ACCEPTS A STRING OF HEX DIGITS FROM THE INPUT
                ;              STREAM AND RETURNS THEIR VALUE AS A 16 BIT BINARY
                ;              INTEGER.  IF MORE THAN 4 HEX DIGITS ARE ENTERED,
                ;              ONLY THE LAST 4 ARE USED.  THE NUMBER TERMINATES WHEN
                ;              A VALID DELIMITER IS ENCOUNTERED.  THE DELIMITER IS
                ;              ALSO RETURNED AS AN OUTPUT OF THE FUNCTION.  ILLEGAL
                ;              CHARACTERS (NOT HEX DIGITS OR DELIMITERS) CAUSE AN
```

```
                ;               ERROR INDICATION.  IF THE FIRST (VALID) CHARACTER
                ;               ENCOUNTERED IN THE INPUT STREAM IS NOT A DELIMITER,
                ;               GETHX WILL RETURN WITH THE CARRY BIT SET TO 1;
                ;               OTHERWISE, THE CARRY BIT IS SET TO 0 AND THE CONTENTS
                ;               OF BC ARE UNDEFINED.
                ;
 0227           GETHX:
 0227   E5              PUSH    H       ; SAVE HL
 0228   210000          LXI     H,0     ; INITIALIZE RESULT
 022B   1E00            MVI     E,0     ; INITIALIZE DIGIT FLAG TO FALSE
 022D           GHX05:
 022D   CD2002          CALL    GETCH   ; GET A CHARACTER
 0230   CDF901          CALL    ECHO    ; ECHO THE CHARACTER
 0233   CD8203          CALL    VALDL   ; SEE IF DELIMITER
     1          +       FALSE   GHX10   ; NO - BRANCH
 0236 1 D24502  +       JNC     GHX10
 0239   51              MOV     D,C     ; YES - ALL DONE, BUT WANT TO RETURN DELIMITER
 023A   E5              PUSH    H
 023B   C1              POP     B       ; MOVE RESULT TO BC
 023C   E1              POP     H       ; RESTORE HL
 023D   7B              MOV     A,E     ; GET FLAG
 023E   B7              ORA     A       ; SET F/F'S
 023F   C23B03          JNZ     SRET    ; IF FLAG NON-0, A NUMBER HAS BEEN FOUND
 0242   CA1D02          JZ      FRET    ; ELSE, DELIMITER WAS FIRST CHARACTER
 0245           GHX10:
 0245   CD6703          CALL    VALDG   ; IF NOT DELIMITER, SEE IF DIGIT
     1          +       FALSE   ERROR   ; ERROR IF NOT A VALID DIGIT, EITHER
 0248 1 D21202  +       JNC     ERROR
 024B   CDDF01          CALL    CNVBN   ; CONVERT DIGIT TO ITS BINARY VALUE
 024E   1EFF            MVI     E,0FFH  ; SET DIGIT FLAG NON-0
 0250   29              DAD     H       ; *2
 0251   29              DAD     H       ; *4
 0252   29              DAD     H       ; *8
 0253   29              DAD     H       ; *16
 0254   0600            MVI     B,0     ; CLEAR UPPER 8 BITS OF BC PAIR
 0256   4F              MOV     C,A     ; BINARY VALUE OF CHARACTER INTO C
 0257   09              DAD     B       ; ADD THIS VALUE TO PARTIAL RESULT
 0258   C32D02          JMP     GHX05   ; GET NEXT CHARACTER
                ;
                ;*****************************************************************
                ;
                ; FUNCTION: GETNM
                ; INPUTS: C - COUNT OF NUMBERS TO FIND IN INPUT STREAM
                ; OUTPUTS: TOP OF STACK - NUMBERS FOUND IN REVERSE ORDER (LAST ON TOP
                ;                         OF STACK)
                ; CALLS: GETHX,HILO,ERROR
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: GETNM FINDS A SPECIFIED COUNT OF NUMBERS, BETWEEN 1
                ;               AND 3, INCLUSIVE,  IN THE INPUT
```

```
                        ;               STREAM AND RETURNS THEIR VALUES ON THE STACK.  IF 2
                        ;               OR MORE NUMBERS ARE REQUESTED, THEN THE FIRST MUST BE
                        ;               LESS THAN OR EQUAL TO THE SECOND, OR THE FIRST AND
                        ;               SECOND NUMBERS WILL BE SET EQUAL.  THE LAST NUMBER
                        ;               REQUESTED MUST BE TERMINATED BY A CARRIAGE RETURN
                        ;               OR AN ERROR INDICATION WILL RESULT.
                        ;
  025B                  GETNM:
  025B   2E03                 MVI     L,3       ; PUT MAXIMUM ARGUMENT COUNT INTO L
  025D   79                   MOV     A,C       ; GET THE ACTUAL ARGUMENT COUNT
  025E   E603                 ANI     3         ; FORCE TO MAXIMUM OF 3
  0260   C8                   RZ                ; IF 0, DON'T BOTHER TO DO ANYTHIING
  0261   67                   MOV     H,A       ; ELSE, PUT ACTUAL COUNT INTO H
  0262                  GNM05:
  0262   CD2702               CALL    GETHX     ; GET A NUMBER FROM INPUT STREAM
       1            +         FALSE   ERROR     ; ERROR IF NOT THERE - TOO FEW NUMBERS
  0265 1 D21202       +        JNC     ERROR
  0268   C5                   PUSH    B         ; ELSE, SAVE NUMBER ON STACK
  0269   2D                   DCR     L         ; DECREMENT MAXIMUM ARGUMENT COUNT
  026A   25                   DCR     H         ; DECREMENT ACTUAL ARGUMENT COUNT
  026B   CA7702               JZ      GNM10     ; BRANCH IF NO MORE NUMBERS WANTED
  026E   7A                   MOV     A,D       ; ELSE, GET NUMBER TERMINATOR TO A
  026F   FE0D                 CPI     CR        ; SEE IF CARRIAGE RETURN
  0271   CA1202               JZ      ERROR     ; ERROR IF SO - TOO FEW NUMBERS
  0274   C36202               JMP     GNM05     ; ELSE, PROCESS NEXT NUMBER
  0277                  GNM10:
  0277   7A                   MOV     A,D       ; WHEN COUNT 0, CHECK LAST TERMINATOR
  0278   FE0D                 CPI     CR
  027A   C21202               JNZ     ERROR     ; ERROR IF NOT CARRIAGE RETURN
  027D   01FFFF               LXI     B,0FFFFH          ; HL GETS LARGEST NUMBER
  0280   7D                   MOV     A,L       ; GET WHAT'S LEFT OF MAXIMUM ARG COUNT
  0281   B7                   ORA     A         ; CHECK FOR 0
  0282   CA8A02               JZ      GNM20     ; IF YES, 3 NUMBERS WERE INPUT
  0285                  GNM15:
  0285   C5                   PUSH    B         ; IF NOT, FILL REMAINING ARGUMENTS WITH 0FFFFH
  0286   2D                   DCR     L
  0287   C28502               JNZ     GNM15
  028A                  GNM20:
  028A   C1                   POP     B         ; GET THE 3 ARGUMENTS OUT
  028B   D1                   POP     D
  028C   E1                   POP     H
  028D   CDA002               CALL    HILO      ; SEE IF FIRST >= SECOND
       1            +         FALSE   GNM25     ; NO - BRANCH
  0290 1 D29502       +        JNC     GNM25
  0293   54                   MOV     D,H
  0294   5D                   MOV     E,L       ; YES - MAKE SECOND EQUAL TO THE FIRST
  0295                  GNM25:
  0295   E3                   XTHL              ; PUT FIRST ON STACK - GET RETURN ADDR
  0296   D5                   PUSH    D         ; PUT SECOND ON STACK
  0297   C5                   PUSH    B         ; PUT THIRD ON STACK
  0298   E5                   PUSH    H         ; PUT RETURN ADDRESS ON STACK
```

```
0299                GNM30:
0299   3D               DCR     A       ; DECREMENT RESIDUAL COUNT
029A   F8               RM              ; IF NEGATIVE, PROPER RESULTS ON STACK
029B   E1               POP     H       ; ELSE, GET RETURN ADDR
029C   E3               XTHL            ; REPLACE TOP RESULT WITH RETURN ADDR
029D   C39902           JMP     GNM30   ; TRY AGAIN
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION: HILO
                    ; INPUTS: DE - 16 BIT INTEGER
                    ;         HL - 16 BIT INTEGER
                    ; OUTPUTS: CARRY - 0 IF HL<DE
                    ;                - 1 IF HL>=DE
                    ; CALLS: NOTHING
                    ; DESTROYS: A,F/F'S
                    ; DESCRIPTION: HILO COMPARES THE 2 16 BIT INTEGERS IN HL AND DE.  THE
                    ;              INTEGERS ARE TREATED AS UNSIGNED NUMBERS.  THE CARRY
                    ;              BIT IS SET ACCORDING TO THE RESULT OF THE COMPARISON.
                    ;
02A0                HILO:
02A0   C5               PUSH    B       ; SAVE BC
02A1   47               MOV     B,A     ; SAVE A REGISTER
02A2   23               INX     H       ; INCREMENT HL BY 1
02A3   7C               MOV     A,H     ; WANT TO TEST FOR 0 RESULT AFTER
02A4   B5               ORA     L       ; /INCREMENTING
02A5   2B               DCX     H       ; RESTORE HL
02A6   37               STC             ; SET CARRY
02A7   CAAF02           JZ      HIL05   ; IF SO, CARRY IS SET PROPERLY
02AA   7D               MOV     A,L     ; IF NOT, MOVE L TO A
02AB   93               SUB     E       ; SUBTRACT E
02AC   7C               MOV     A,H     ; MOVE H TO A
02AD   9A               SBB     D       ; SUBTRACT D WITH BORROW
02AE   3F               CMC             ; COMPLIMENT CARRY FOR CORRECT CARRY BIT VALUE
02AF                HIL05:
02AF   78               MOV     A,B     ; RESTORE A
02B0   C1               POP     B       ; RESTORE BC
02B1   C9               RET             ; EXIT
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION INUST
                    ; INPUTS: NONE
                    ; OUTPUTS: NOTHING
                    ; CALLS: NOTHING
                    ; DESTROYS: A,H,L,SP
                    ; DESCRIPTION: INUST OUTPUTS TO THE USART THE COMMAND WORD
```

```
                        ;               AND INITIALIZES THE STACK POINTER.
                        ;
02B2                    INUST:
02B2    3E25                MVI     A,CMD
02B4    D3ED                OUT     CNCTL   ; OUTPUT COMMAND WORD TO USART
02B6    21023C              LXI     H,MSTAK-44      ; LOAD POINTER TO STACK
02B9    22383C              SHLD    SSAVE   ; INITIALIZE USER STACK POINTER
02BC    312E3C              LXI     SP,MSTAK        ; INITIALIZE MONITOR STACK
02BF    C31E00              JMP     SOMSG   ; GO TO PRINT SIGNON MESSAGE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: NMOUT
                        ; INPUTS: A - 8 BIT INTEGER
                        ; OUTPUTS: NONE
                        ; CALLS: ECHO,PRVAL
                        ; DESTROYS: A,B,C,F/F'S
                        ; DESCRIPTION: NMOUT CONVERTS THE 8 BIT, UNSIGNED INTEGER IN THE
                        ;               A REGISTER INTO 2 ASCII CHARACTERS.  THE ASCII CHARACTERS
                        ;               ARE THE ONES REPRESENTING THE 8 BITS.  THESE TWO
                        ;               CHARACTERS ARE SENT TO THE CONSOLE AT THE CURRENT PRINT
                        ;               POSITION OF THE CONSOLE.
                        ;
02C2                    NMOUT:
02C2    F5                  PUSH    PSW     ; SAVE ARGUMENT
02C3    0F                  RRC
02C4    0F                  RRC
02C5    0F                  RRC
02C6    0F                  RRC             ; GET UPPER 4 BITS TO LOW 4 BIT POSITIONS
02C7    CDD502              CALL    PRVAL   ;CONVERT LOWER 4 BITS TO ASCII
02CA    CDF901              CALL    ECHO    ; SEND TO TERMINAL
02CD    F1                  POP     PSW     ; GET BACK ARGUMENT
02CE    CDD502              CALL    PRVAL
02D1    CDF901              CALL    ECHO
02D4    C9                  RET
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION;  PRVAL
                        ; INPUTS: A - INTEGER, RANGE 0 TO F
                        ; OUTPUTS: A - ASCII CHARACTER
                        ; CALLS: NOTHING
                        ; DESTROYS: NOTHING
                        ; DESCRIPTION: PRVAL CONVERTS A NUMBER IN THE RANGE 0 TO F HEX TO
                        ;               THE CORRESPONDING ASCII CHARACTER, 0-9,A-F.  PRVAL
                        ;               DOES NOT CHECK THE VALIDITY OF ITS INPUT ARGUMENT.
                        ;
```

```
02D5                PRVAL:
02D5   E60F             ANI     HCHAR   ; MASK OUT UPPER 4 BITS - WANT 1 HEX CHAR
02D7   C690             ADI     90H     ; SET UP A SO THAT A-F CAUSE A CARRY
02D9   27               DAA             ; ADJUST CONTENTS OF A REGISTER
02DA   CE40             ACI     40H     ; ADD IN CARRY AND ADJUST UPPER 4 BITS
02DC   27               DAA             ; ADJUST CONTENTS OF A REGISTER AGAIN
02DD   4F               MOV     C,A     ; MOVE ASCII CHARACTER TO C
02DE   C9               RET             ; ALL DONE
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION: REGDS
                    ; INPUTS: NONE
                    ; OUTPUTS: NONE
                    ; CALLS: ECHO,NMOUT,ERROR,CROUT
                    ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                    ; DESCRIPTION: REGDS DISPLAYS THE CONTENTS OF THE REGISTER SAVE
                    ;               LOCATIONS, IN FORMATTED FORM, ON THE CONSOLE.  THE
                    ;               DISPLAY IS DRIVEN FROM A TABLE, RTAB, WHICH CONTAINS
                    ;               THE REGISTER'S PRINT SYMBOL, SAVE LOCATION ADDRESS,
                    ;               AND LENGTH (8 OR 16 BITS).
                    ;
02DF                REGDS:
02DF   21C003           LXI     H,RTAB  ; LOAD HL WITH ADDRESS OF START OF TABLE
02E2                REG05:
02E2   4E               MOV     C,M     ; GET PRINT SYMBOL OF REGISTER
02E3   79               MOV     A,C
02E4   B7               ORA     A       ; TEST FOR 0 - END OF TABLE
02E5   C2EC02           JNZ     REG10   ; IF NOT END, BRANCH
02E8   CDF301           CALL    CROUT   ; ELSE, CARRIAGE RETURN/LINE FEED TO END
02EB   C9               RET             ; /DISPLAY
02EC                REG10:
02EC   CDF901           CALL    ECHO    ; ECHO CHARACTER
02EF   0E3D             MVI     C,'='
02F1   CDF901           CALL    ECHO    ; OUTPUT EQUALS SIGN, I.E. A=
02F4   23               INX     H       ; POINT TO START OF SAVE LOCATION ADDRESS
02F5   5E               MOV     E,M     ; GET LSP OF SAVE LOCATION ADDRESS TO E
02F6   163C             MVI     D,DATA SHR 8    ; PUT MSP OF SAVE LOC ADDRESS INTO D
02F8   23               INX     H       ; POINT TO LENGTH FLAG
02F9   1A               LDAX    D       ; GET CONTENTS OF SAVE ADDRESS
02FA   CDC202           CALL    NMOUT   ; DISPLAY ON CONSOLE
02FD   7E               MOV     A,M     ; GET LENGTH FLAG
02FE   B7               ORA     A       ; SET SIGN F/F
02FF   CA0703           JZ      REG15   ; IF 0, REGISTER IS 8 BITS
0302   1B               DCX     D       ; ELSE, 16 BIT REGISTER SO MORE TO DISPLAY
0303   1A               LDAX    D       ; GET LOWER 8 BITS
0304   CDC202           CALL    NMOUT   ; DISPLAY THEM
0307                REG15:
0307   0E20             MVI     C,' '
```

```
0309  CDF901        CALL    ECHO    ; OUTPUT BLANK CHARACTER
030C  23            INX     H       ; POINT TO START OF NEXT TABLE ENTRY
030D  C3E202        JMP     REG05   ; DO NEXT REGISTER
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: RGADR
                ; INPUTS: C - CHARACTER DENOTING REGISTER
                ; OUTPUTS: BC - ADDRESS OF ENTRY IN RTAB CORRESPONDING TO REGISTER
                ; CALLS: ERROR
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: RGADR TAKES A SINGLE CHARACTER AS INPUT.  THIS CHARACTER
                ;              DENOTES A REGISTER.  RGADR SEARCHES THE TABLE RTAB
                ;              FOR A MATCH ON THE INPUT ARGUMENT.  IF ONE OCCURS,
                ;              RGADR RETURNS THE ADDRESS OF THE ADDRESS OF THE
                ;              SAVE LOCATION CORRESPONDING TO THE REGISTER.  THIS
                ;              ADDRESS POINTS INTO RTAB.  IF NO MATCH OCCURS, THEN
                ;              THE REGISTER IDENTIFIER IS ILLEGAL AND CONTROL IS
                ;              PASSED TO THE ERROR ROUTINE.
                ;
0310            RGADR:
0310  21C003        LXI     H,RTAB  ; HL GETS ADDRESS OF TABLE START
0313  110300        LXI     D,RTABS ; DE GET SIZE OF A TABLE ENTRY
0316            RGA05:
0316  7E            MOV     A,M     ; GET REGISTER IDENTIFIER
0317  B7            ORA     A       ; CHECK FOR TABLE END (IDENTIFIER IS 0)
0318  CA1202        JZ      ERROR   ; IF AT END OF TABLE, ARGUMENT IS ILLEGAL
031B  B9            CMP     C       ; ELSE, COMPARE TABLE ENTRY AND ARGUMENT
031C  CA2303        JZ      RGA10   ; IF EQUAL, WE'VE FOUND WHAT WE'RE LOOKING FOR
031F  19            DAD     D       ; ELSE, INCREMENT TABLE POINTER TO NEXT ENTRY
0320  C31603        JMP     RGA05   ; TRY AGAIN
0323            RGA10:
0323  23            INX     H       ; IF A MATCH, INCREMENT TABLE POINTER TO
0324  44            MOV     B,H     ; /SAVE LOCATION ADDRESS
0325  4D            MOV     C,L     ; RETURN THIS VALUE
0326  C9            RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: RSTTF
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: NOTHING
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: RSTTF RESTORES ALL CPU REGISTER, FLIP/FLOPS, STACK
                ;              POINTER AND PROGRAM COUNTER FROM THEIR RESPECTIVE
                ;              SAVE LOCATIONS IN MEMORY.  THE ROUTINE THEN TRANSFERS
```

```
                ;               CONTROL TO THE LOCATION SPECIFIED BY THE PROGRAM
                ;               COUNTER (I.E. THE RESTORED VALUE).  THE ROUTINE
                ;               EXITS WITH THE INTERRUPTS ENABLED.
                ;
0327            RSTTF:
0327  F3             DI              ; DISABLE INTERRUPTS WHILE RESTORING THINGS
0328  312E3C         LXI     SP,MSTAK        ; SET MONITOR STACK POINTER TO START
                                       ; /OF STACK
032B  D1             POP     D       ; START ALSO END OF REGISTER SAVE AREA
032C  C1             POP     B
032D  F1             POP     PSW
032E  2A383C         LHLD    SSAVE   ; RESTORE USER STACK POINTER
0331  F9             SPHL
0332  2A363C         LHLD    PSAVE
0335  E5             PUSH    H       ; PUT USER RETURN ADDRESS ON USER STACK
0336  2A343C         LHLD    LSAVE   ; RESTORE HL REGISTERS
0339  FB             EI              ; ENABLE INTERRUPTS NOW
033A  C9             RET             ; JUMP TO RESTORED PC LOCATION
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: SRET
                ; INPUTS: NONE
                ; OUTPUTS: CARRY = 1
                ; CALLS: NOTHING
                ; DESTROYS: CARRY
                ; DESCRIPTION: SRET IS JUMPED TO BY ROUTINES WISHING TO RETURN SUCCESS.
                ;               SRET SETS THE CARRY TRUE AND THEN RETURNS TO THE
                ;               CALLER OF THE ROUTINE INVOKING SRET.
                ;
033B            SRET:
033B  37             STC             ; SET CARRY TRUE
033C  C9             RET             ; RETURN APPROPRIATELY
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: STHF0
                ; INPUTS: DE - 16 BIT ADDRESS OF BYTE TO BE STORED INTO
                ; OUTPUTS: NONE
                ; CALLS: NOTHING
                ; DESTROYS: A,B,C,H,L,F/F'S
                ; DESCRIPTION: STHF0 CHECKS THE HALF BYTE FLAG IN TEMP TO SEE IF
                ;               IT IS SET TO LOWER.  IF SO, STHF0 STORES A 0 TO
                ;               PAD OUT THE LOWER HALF OF THE ADDRESSED BYTE;
                ;               OTHERWISE, THE ROUTINE TAKES NO ACTION.
                ;
033D            STHF0:
```

```
033D  3A3A3C        LDA     TEMP    ; GET HALF BYTE FLAG
0340  B7            ORA     A       ; SET F/F'S
0341  C0            RNZ             ; IF SET TO UPPER, DON'T DO ANYTHING
0342  0E00          MVI     C,0     ; ELSE, WANT TO STORE THE VALUE 0
0344  CD4803        CALL    STHLF   ; DO IT
0347  C9            RET
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION: STHLF
                    ; INPUTS: C - 4 BIT VALUE TO BE STORED IN HALF BYTE
                    ;         DE - 16 BIT ADDRESS OF BYTE TO BE STORED INTO
                    ; OUTPUTS: NONE
                    ; CALLS: NOTHING
                    ; DESTROYS: A,B,C,H,L,F/F'S
                    ; DESCRIPTION: STHLF TAKES THE 4 BIT VALUE IN C AND STORES IT IN
                    ;               HALF OF THE BYTE ADDRESSED BY REGISTERS DE.  THE
                    ;               HALF BYTE USED (EITHER UPPER OR LOWER) IS DENOTED
                    ;               BY THE VALUE OF THE FLAG IN TEMP.  STHLF ASSUMES
                    ;               THAT THIS FLAG HAS BEEN PREVIOUSLY SET
                    ;               (NOMINALLY BY ICMD).
                    ;
0348                STHLF:
0348  D5                  PUSH    D
0349  E1                  POP     H       ; MOVE ADDRESS OF BYTE INTO HL
034A  79                  MOV     A,C     ; GET VALUE
034B  E60F                ANI     0FH     ; FORCE TO 4 BIT LENGTH
034D  4F                  MOV     C,A     ; PUT VALUE BACK
034E  3A3A3C              LDA     TEMP    ; GET HALF BYTE FLAG
0351  B7                  ORA     A       ; CHECK FOR LOWER HALF
0352  C25B03              JNZ     STH05   ; BRANCH IF NOT
0355  7E                  MOV     A,M     ; ELSE, GET BYTE
0356  E6F0                ANI     0F0H    ; CLEAR LOWER 4 BITS
0358  B1                  ORA     C       ; OR IN VALUE
0359  77                  MOV     M,A     ; PUT BYTE BACK
035A  C9                  RET
035B                STH05:
035B  7E                  MOV     A,M     ; IF UPPER HALF, GET BYTE
035C  E60F                ANI     0FH     ; CLEAR UPPER 4 BITS
035E  47                  MOV     B,A     ; SAVE BYTE IN B
035F  79                  MOV     A,C     ; GET VALUE
0360  0F                  RRC
0361  0F                  RRC
0362  0F                  RRC
0363  0F                  RRC             ; ALIGN TO UPPER 4 BITS
0364  B0                  ORA     B       ; OR IN ORIGINAL LOWER 4 BITS
0365  77                  MOV     M,A     ; PUT NEW CONFIGURATION BACK
0366  C9                  RET
                    ;
```

```
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: VALDG
                        ; INPUTS: C - ASCII CHARACTER
                        ; OUTPUTS: CARRY - 1 IF CHARACTER REPRESENTS VALID HEX DIGIT
                        ;                 - 0 OTHERWISE
                        ; CALLS: NOTHING
                        ; DESTROYS: A,F/F'S
                        ; DESCRIPTION: VALDG RETURNS SUCCESS IF ITS INPUT ARGUMENT IS
                        ;              AN ASCII CHARACTER REPRESENTING A VALID HEX DIGIT
                        ;              (0-9,A-F), AND FAILURE OTHERWISE.
                        ;
0367                    VALDG:
0367  79                      MOV     A,C
0368  FE30                    CPI     '0'     ; TEST CHARACTER AGAINST '0'
036A  FA1D02                  JM      FRET    ; IF ASCII CODE LESS, CANNOT BE VALID DIGIT
036D  FE39                    CPI     '9'     ; ELSE, SEE IF IN RANGE '0'-'9'
036F  FA3B03                  JM      SRET    ; CODE BETWEEN '0' AND '9'
0372  CA3B03                  JZ      SRET    ; CODE EQUAL '9'
0375  FE41                    CPI     'A'     ; NOT A DIGIT - TRY FOR A LETTER
0377  FA1D02                  JM      FRET    ; NO - CODE BETWEEN '9' AND 'A'
037A  FE47                    CPI     'G'
037C  F21D02                  JP      FRET    ; NO - CODE GREATER THAN 'F'
037F  C33B03                  JMP     SRET    ; OKAY - CODE IS 'A' TO 'F', INCLUSIVE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: VALDL
                        ; INPUTS: C - CHARACTER
                        ; OUTPUTS: CARRY - 1 IF INPUT ARGUMENT VALID DELIMTER
                        ;                 - 0 OTHERWISE
                        ; CALLS: NOTHING
                        ; DESTROYS: A,F/F'S
                        ; DESCRIPTION: VALDL RETURNS SUCCESS IF ITS INPUT ARGUMENT IS A VALID
                        ;              DELIMITER CHARACTER (SPACE, COMMA, CARRIAGE RETURN) AND
                        ;              FAILURE OTHERWISE.
                        ;
0382                    VALDL:
0382  79                      MOV     A,C
0383  FE2C                    CPI     ','     ; CHECK FOR COMMA
0385  CA3B03                  JZ      SRET
0388  FE0D                    CPI     CR      ; CHECK FOR CARRIAGE RETURN
038A  CA3B03                  JZ      SRET
038D  FE20                    CPI     ' '     ; CHECK FOR SPACE
038F  CA3B03                  JZ      SRET
0392  C31D02                  JMP     FRET    ; ERROR IF NONE OF THE ABOVE
                        ;
```

```
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;                       MONITOR TABLES
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
0395                    SGNON:                          ; SIGNON MESSAGE
0395  0D0A3830                DB      CR,LF,'80/10 MONITOR',CR,LF
0399  2F313020
039D  4D4F4E49
03A1  544F520D
03A5  0A
0011                    LSGNON          EQU     $-SGNON ; LENGTH OF SIGNON MESSAGE
                        ;
03A6                    CADR:                   ; TABLE OF ADDRESSES OF COMMAND ROUTINES
03A6  0000                    DW        0       ; DUMMY
03A8  3301                    DW        XCMD
03AA  0F01                    DW        SCMD
03AC  EF00                    DW        MCMD
03AE  A900                    DW        ICMD
03B0  8B00                    DW        GCMD
03B2  5F00                    DW        DCMD
03B4  0604                    DW        RCMD
03B6  4104                    DW        WCMD
                        ;
03B8                    CTAB:                   ; TABLE OF VALID COMMAND CHARACTERS
03B8  57                      DB        'W'
03B9  52                      DB        'R'
03BA  44                      DB        'D'
03BB  47                      DB        'G'
03BC  49                      DB        'I'
03BD  4D                      DB        'M'
03BE  53                      DB        'S'
03BF  58                      DB        'X'
0008                    NCMDS EQU       $-CTAB  ; NUMBER OF VALID COMMANDS
                        ;
                        ;
03C0                    RTAB:                   ; TABLE OF REGISTER INFORMATION
03C0  41                      DB        'A'     ; REGISTER IDENTIFIER
03C1  33                      DB        ASAVE AND 0FFH  ; ADDRESS OF REGISTER SAVE LOCATION
03C2  00                      DB        0       ; LENGTH FLAG - 0=8 BITS, 1=16 BITS
0003                    RTABS EQU       $-RTAB  ; SIZE OF AN ENTRY IN THIS TABLE
03C3  42                      DB        'B'
03C4  31                      DB        BSAVE AND 0FFH
03C5  00                      DB        0
03C6  43                      DB        'C'
03C7  30                      DB        CSAVE AND 0FFH
```

```
03C8   00            DB      0
03C9   44            DB      'D'
03CA   2F            DB      DSAVE AND 0FFH
03CB   00            DB      0
03CC   45            DB      'E'
03CD   2E            DB      ESAVE AND 0FFH
03CE   00            DB      0
03CF   46            DB      'F'
03D0   32            DB      FSAVE AND 0FFH
03D1   00            DB      0
03D2   48            DB      'H'
03D3   35            DB      HSAVE AND 0FFH
03D4   00            DB      0
03D5   4C            DB      'L'
03D6   34            DB      LSAVE AND 0FFH
03D7   00            DB      0
03D8   4D            DB      'M'
03D9   35            DB      HSAVE AND 0FFH
03DA   01            DB      1
03DB   50            DB      'P'
03DC   37            DB      PSAVE+1 AND 0FFH
03DD   01            DB      1
03DE   53            DB      'S'
03DF   39            DB      SSAVE+1 AND 0FFH
03E0   01            DB      1
03E1   00            DB      0       ; END OF TABLE MARKERS
03E2   00            DB      0
                ;
                ;
                ;*****************************************************************
                ;
                ;
03E3            CPYRT:
03E3   28432920      DB      '(C) 1976 INTEL CORP'
03E7   31393736
03EB   20494E54
03EF   454C2043
03F3   4F5250
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;
03FA                ORG     BRTAB
                ;
03FA   C3E801       JMP     CO      ; BRANCH TABLE FOR USER ACCESSIBLE ROUTINES
03FD   C3D501       JMP     CI
0400   C31C05       JMP     RI
0403   C30F05       JMP     PO
                ;
```

```
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION RCMD
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: GETCH,ECHO,CO,RICH,BYTE
                        ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                        ; DESCRIPTION: RCMD IMPLEMENTS THE READ HEXADECIMAL TAPE (R)
                        ;              COMMAND.
                        ;
0406                    RCMD:
0406  CD2002                 CALL    GETCH   ; GET CARRIAGE RETURN CHARACTER
0409  CDF901                 CALL    ECHO    ; ECHO IT
040C  79                    MOV     A,C     ; MOVE IT TO A REGISTER
040D  FE0D                  CPI     CR      ; SEE IF CARRIAGE RETURN
040F  C21202                JNZ     ERROR   ; ERROR IF NOT PROPERLY TERMINATED
0412                    RCM05:
0412  CD1305                 CALL    RICH    ; READ CHARACTER FROM TAPE
0415  FE3A                  CPI     ':'     ; SEE IF RECORD MARK
0417  C21204                JNZ     RCM05   ; TRY AGAIN IF NOT MARK
041A  AF                    XRA     A       ; ZERO A REGISTER
041B  57                    MOV     D,A     ; INITIALIZE D FOR HOLDING THE CHECKSUM
041C  CD9604                CALL    BYTE    ; READ TWO CHARACTERS FROM TAPE
041F  CA2C00                JZ      GETCM   ; IF ZERO RECORD LENGTH, ALL DONE
0422  5F                    MOV     E,A     ; OTHERWISE, PUT THE RECORD LENGTH IN E
0423  CD9604                CALL    BYTE    ; GET MSB OF LOAD ADDRESS
0426  67                    MOV     H,A     ; MOVE TO H
0427  CD9604                CALL    BYTE    ; GET LSB OF LOAD ADDRESS
042A  6F                    MOV     L,A     ; MOVE TO L
042B  CD9604                CALL    BYTE    ; GET RECORD TYPE
042E  4B                    MOV     C,E     ; MOVE RECORD LENGTH TO C
042F                    RCM10:
042F  CD9604                CALL    BYTE    ; READ DATA FROM TAPE
0432  77                    MOV     M,A     ; PUT DATA INTO MEMORY
0433  23                    INX     H       ; INCREMENT HL FOR NEXT LOCATION
0434  1D                    DCR     E       ; DECREMENT RECORD LENGTH
0435  C22F04                JNZ     RCM10   ; LOOP UNTIL DONE
0438  CD9604                CALL    BYTE    ; READ CHECKSUM FROM TAPE
043B  C21202                JNZ     ERROR   ; CHECKSUM ERROR IF NOT ZERO
043E  C31204                JMP     RCM05   ; GET ANOTHER RECORD
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION WCMD
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: GETNM,LEAD,PO,PBYTE,PADR,PEOL,PEOF
```

```
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: WCMD IMPLEMENTS THE WRITE HEXADECIMAL TAPE (W)
                ;              COMMAND.
                ;
0441            WCMD:
0441  0E02           MVI     C,2
0443  CD5B02         CALL    GETNM   ; GET 2 NUMBERS FROM INPUT STREAM
0446  CDBA04         CALL    LEAD    ; PUNCH 60 NULL CHARACTERS FOR TAPE LEADER
0449  D1             POP     D       ; ENDING ADDRESS TO DE
044A  E1             POP     H       ; STARTING ADDRESS TO HL
044B            WCM05:
044B  7D             MOV     A,L     ; MOVE L TO A
044C  C610           ADI     16      ; INCREMENT THE LSB OF STARTING ADDRESS BY 16
044E  4F             MOV     C,A     ; MOVE RESULT TO C
044F  7C             MOV     A,H     ; MOVE H TO A
0450  CE00           ACI     0       ; ADD CARRY IN FROM PREVIOUS OPERATION
0452  47             MOV     B,A     ; SAVE RESULT IN B
0453  7B             MOV     A,E     ; NOW MOVE LSB OF ENDING ADDRESS TO A
0454  91             SUB     C       ; SUBTRACT LSB OF STARTING ADDRESS
0455  4F             MOV     C,A     ; SAVE IN C
0456  7A             MOV     A,D     ; NOW GET MSB OF ENDING ADDRESS IN A
0457  98             SBB     B       ; SUBTRACT MSB OF STARTING ADDRESS
0458  DA6004         JC      WCM10   ; BRANCH IF THE RECORD LENGTH IS NOT 16
045B  3E10           MVI     A,16    ; OTHERWISE SET A TO RECORD LENGTH OF 16
045D  C36304         JMP     WCM15   ; NOW BRANCH TO PUNCH THE RECORD
0460            WCM10:
0460  79             MOV     A,C     ; THIS IS THE LAST RECORD
0461  C611           ADI     17      ; SO SET A TO REMAINING DATA LENGTH
0463            WCM15:
0463  B7             ORA     A       ; CHECK FOR RECORD LENGTH OF ZERO
0464  CA9004         JZ      WCM25   ; IF IT IS, ALL DONE
0467  D5             PUSH    D       ; OTHERWISE, SAVE ENDING ADDRESS
0468  5F             MOV     E,A     ; PUT RECORD LENGTH IN E
0469  1600           MVI     D,0     ; INITIALIZE D FOR HOLDING CHECKSUM
046B  0E3A           MVI     C,':'
046D  CD0F05         CALL    PO      ; PUNCH RECORD MARK CHARACTER
0470  7B             MOV     A,E     ; PUT RECORD LENGTH IN A
0471  CDCF04         CALL    PBYTE   ; PUNCH RECORD LENGTH
0474  CDC604         CALL    PADR    ; PUNCH STARTING ADDRESS
0477  AF             XRA     A       ; ZERO A
0478  CDCF04         CALL    PBYTE   ; PUNCH RECORD TYPE
047B            WCM20:
047B  7E             MOV     A,M     ; CET DATA TO BE PUNCHED FROM MEMORY
047C  CDCF04         CALL    PBYTE   ; PUNCH IT
047F  23             INX     H       ; INCREMENT MEMORY ADDRESS
0480  1D             DCR     E       ; DECREMENT LENGTH COUNT
0481  C27B04         JNZ     WCM20   ; LOOP UNTIL ALL DATA PUNCHED
0484  AF             XRA     A
0485  92             SUB     D       ; PUNCH CHECKSUM
0486  CDCF04         CALL    PBYTE
0489  D1             POP     D       ; RESTORE ENDING ADDRESS
```

```
048A   CD0405          CALL    PECL    ; PUNCH CARRIAGE RETURN AND LINE FEED
048D   C34B04          JMP     WCM05
0490                WCM25:
0490   CDE604          CALL    PEOF    ; PUNCH END OF FILE RECORD
0493   C31702          JMP     EXIT    ; ALL DONE
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION BYTE
                    ; INPUTS: D - CURRENT VALUE OF CHECKSUM
                    ; OUTPUTS: A - HEXADECIMAL CHARACTER
                    ;          D - UPDATED VALUE OF CHECKSUM
                    ; CALLS: RICH,CNVBN
                    ; DESTROYS: A,B,C,D,F/F'S
                    ; DESCRIPTION: BYTE READS 2 ASCII CHARACTERS FROM THE TELETYPEWRITER
                    ;               AND CONVERTS THE CHARACTERS TO ONE HEXADECIMAL CHARACTER.
                    ;               THE A REGISTER CONTAINS THE FINAL CHARACTER AND THE
                    ;               D REGISTER CONTAINS THE UPDATED VALUE OF
                    ;               THE CHECKSUM.
                    ;
0496                BYTE:
0496   C5              PUSH    B       ; SAVE BC
0497   CD1305          CALL    RICH    ; READ ASCII CHARACTER FROM TAPE
049A   4F              MOV     C,A
049B   CDDF01          CALL    CNVBN   ; CONVERT CHARACTER TO HEXADECIMAL
049E   07              RLC             ; POSITION VALUE INTO UPPER 4 BITS
049F   07              RLC
04A0   07              RLC
04A1   07              RLC
04A2   47              MOV     B,A     ; SAVE RESULTS IN B
04A3   CD1305          CALL    RICH    ; GET ANOTHER CHARACTER FROM TAPE
04A6   4F              MOV     C,A
04A7   CDDF01          CALL    CNVBN   ; CONVERT IT
04AA   B0              ORA     B       ; OR IN THE UPPER 4 BITS
04AB   4F              MOV     C,A     ; SAVE
04AC   82              ADD     D       ; INCREMENT CHECKSUM
04AD   57              MOV     D,A
04AE   79              MOV     A,C     ; RESTORE HEX DATA TO A REGISTER
04AF   C1              POP     B       ; RESTORE BC
04B0   C9              RET
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION DELAY
                    ; INPUTS: NONE
                    ; OUTPUTS: NONE
                    ; CALLS: NOTHING
```

```
                ; DESTROYS: F/F'S
                ; DESCRIPTION: DELAY PROVIDES A PROGRAMMED DELAY OF 1 MILLISECOND
                ;              FOR TAPE READER OPERATION.
                ;
04B1            DELAY:
04B1  C5              PUSH    B       ; SAVE BC REGISTERS
04B2  0683            MVI     B,ONEMS ; LOAD 1 MILLISECOND CONSTANT
04B4            DEL1:
04B4  05              DCR     B       ; DECREMENT INNER COUNTER
04B5  C2B404          JNZ     DEL1    ; JUMP IF NOT DONE
04B8  C1              POP     B       ; RESTORE BC REGISTERS
04B9  C9              RET             ; RETURN TO CALLING ROUTINE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION LEAD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: PO
                ; DESTROYS: B,C,F/F'S
                ; DESCRIPTIOM: LEAD OUTPUTS 60 NULL CHARACTERS TO PAPER TAPE TO FORM A
                ;              LEADER.
                ;
04BA            LEAD:
04BA  063C            MVI     B,60    ; LOAD B WITH A COUNT OF 60
04BC            LE05:
04BC  0E00            MVI     C,0
04BE  CD0F05          CALL    PO      ; PUNCH NULL CHARACTER
04C1  05              DCR     B       ; DECREMENT COUNT
04C2  C2BC04          JNZ     LE05    ; DO IT AGAIN IF NOT DONE
04C5  C9              RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION PADR
                ; INPUTS: HL - ADDRESS TO BE PUNCHED
                ; OUTPUTS: NONE
                ; CALLS: PBYTE
                ; DESTROYS: A
                ; DESCRIPTION: PADR PUNCHES ON THE TELETYPEWRITER THE ADDRESS
                ;              CONTAINED IN THE H,L REGISTERS.
                ;
04C6            PADR:
04C6  7C              MOV     A,H     ; PUNCH FIRST HALF OF ADDRESS
04C7  CDCF04          CALL    PBYTE
04CA  7D              MOV     A,L     ; PUNCH SECOND HALF OF ADDRESS
```

```
04CB   CDCF04          CALL    PBYTE
04CE   C9              RET             ; RETURN TO CALLING ROUTINE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION PBYTE
                ; INPUTS: A - CHARACTER TO BE PUNCHED
                ;         D - CURRENT VALUE OF CHECKSUM
                ; OUTPUTS: D - UPDATED VALUE OF CHECKSUM
                ; CALLS: PRVAL,PO
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: PBYTE CONVERTS THE HEXADECIMAL VALUE IN THE A REGISTER
                ;              INTO TWO ASCII CHARACTERS AND PUNCHES THESE CHARACTERS
                ;              ON PAPER TAPE. THE CHECKSUM CONTAINED IN D IS UPDATED.
                ;
04CF            PBYTE:
04CF   F5              PUSH    PSW     ; SAVE A,F/F'S
04D0   0F              RRC             ; POSITION UPPER 4 BITS INTO LOWER 4 BITS
04D1   0F              RRC
04D2   0F              RRC
04D3   0F              RRC
04D4   CDD502          CALL    PRVAL   ; CONVERT UPPER 4 BITS JUST ROTATED TO ASCII
04D7   CD0F05          CALL    PO      ; PUNCH CHARACTER
04DA   F1              POP     PSW     ; RESTORE A,F/F'S
04DB   F5              PUSH    PSW     ; SAVE A AGAIN
04DC   CDD502          CALL    PRVAL   ; CONVERT LOWER 4 BITS TO ASCII CHARACTER
04DF   CD0F05          CALL    PO      ; PUNCH CHARACTER
04E2   F1              POP     PSW     ; RESTORE A
04E3   82              ADD     D       ; ADD VALUE TO CHECKSUM
04E4   57              MOV     D,A     ; UPDATE D REGISTER WITH NEW CHECKSUM
04E5   C9              RET             ; RETURN TO CALLING ROUTINE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION PEOF
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: PO,PBYTE,PADR,LEAD
                ; DESTROYS: A,C,D,H,L,F/F'S
                ; DESCRIPTON: PEOF PUNCHES THE END OF FILE RECORD CONSISTING OF A RECORD
                ;             MARK, A LOAD ADDRESS OF 0, THE RECORD TYPE, AND THE
                ;             RECORD CHECKSUM.
                ;
04E6            PEOF:
04E6   0E3A            MVI     C,':'
04E8   CD0F05          CALL    PO      ; PUNCH RECORD MARK
04EB   AF              XRA     A       ; ZERO CHECKSUM
```

```
04EC   57          MOV     D,A     ; SAVE IN D REGISTER
04ED   CDCF04      CALL    PBYTE   ; PUNCH RECORD LENGTH
04F0   210000      LXI     H,0     ; LOAD HL WITH ZERO ADDRESS
04F3   CDC604      CALL    PADR    ; PUNCH IT
04F6   3E01        MVI     A,1     ; LOAD A WITH RECORD TYPE
04F8   CDCF04      CALL    PBYTE   ; PUNCH IT
04FB   AF          XRA     A       ; ZERO A
04FC   92          SUB     D       ; COMPUTE CHECKSUM
04FD   CDCF04      CALL    PBYTE   ; PUNCH IT
0500   CDBA04      CALL    LEAD    ; PUNCH TRAILER
0503   C9          RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION PEOL
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: PO
                ; DESTROYS: C
                ; DESCRIPTION: PEOL PUNCHES A CARRIAGE RETURN AND LINE FEED ONTO
                ;              PAPER TAPE.
                ;
0504            PEOL:
0504   0E0D        MVI     C,CR
0506   CD0F05      CALL    PO      ; PUNCH CARRIAGE RETURN CHARACTER
0509   0E0A        MVI     C,LF
050B   CD0F05      CALL    PO      ; PUNCH LINE FEED CHARACTER
050E   C9          RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION PO
                ; INPUTS: C - CHARACTER TO BE PUNCHED
                ; OUTPUTS: NONE
                ; CALLS: CO
                ; DESTROYS: NOTHING
                ; DESCRIPTION: PO PUNCHES THE CHARACTER SUPPLIED IN THE C REGISTER TO
                ;              THE USER TELETYPEWRITER.
                ;
050F            PO:
050F   CDE801      CALL    CO      ; CALL CONSOLE OUT TO PERFORM CHARACTER OUTPUT
0512   C9          RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
```

```
                        ; FUNCTION RICH
                        ; INPUTS: NONE
                        ; OUTPUTS: A - ZERO, CARRY - 1 IF END OF FILE
                        ;          A - CHARACTER, CARRY  - 0 IF VALID CHARACTER
                        ; CALLS: RI
                        ; DESTROYS: A,F/F'S
                        ; DESCRIPTION: RICH TESTS FOR AN END OF FILE CONDITION.
                        ;
0513                    RICH:
0513  CD1C05                 CALL    RI      ; READ A CHARACTER FROM TAPE
0516  DA1202                 JC      ERROR   ; JUMP IF READER TIMEOUT ERROR
0519  E67F                   ANI     PRTY0   ; REMOVE PARITY BIT
051B  C9                     RET             ; RETURN TO CALLING ROUTINE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION RI
                        ; INPUTS: NONE
                        ; OUTPUTS: A - ZERO, CARRY - 1 IF END OF FILE
                        ;          A - CHARACTER, CARRY  - 0 IF VALID CHARACTER
                        ; CALLS: DELAY
                        ; DESTROYS: A,F/F'S
                        ; DESCRIPTION: RI READS A CHARACTER FROM THE TTY TAPE READER.
                        ;
051C                    RI:
051C  C5                     PUSH    B       ; SAVE BC
051D                    RI05:
051D  DBED                   IN      CNCTL   ; READ IN USART STATUS
051F  E604                   ANI     TXBE    ; CHECK FOR TRANSMITTER BUFFER EMPTY
0521  CA1D05                 JZ      RI05    ; TRY AGAIN IF NOT EMPTY
0524  3E27                   MVI     A,TTYADV        ; ADVANCE THE TAPE
0526  D3ED                   OUT     CNCTL   ; OUTPUT THE ADVANCE COMMAND
0528  0628                   MVI     B,40    ; INITIALIZE TIMER FOR 40 MS.
052A                    RI07:
052A  CDB104                 CALL    DELAY   ; DELAY FOR 1 MILLISECONDS
052D  05                     DCR     B       ; DECREMENT TIMER
052E  C22A05                 JNZ     RI07    ; JUMP IF TIMER NOT EXPIRED
0531  3E25                   MVI     A,CMD   ; STOP THE READER ADVANCE
0533  D3ED                   OUT     CNCTL   ; OUTPUT STOP COMMAND
0535  06FA                   MVI     B,250   ; INITIALIZE TIMER FOR 250 MS.
0537                    RI10:
0537  DBED                   IN      CONST   ; INPUT READER STATUS
0539  E602                   ANI     RBR     ; CHECK FOR RECEIVER BUFFER READY
053B  C24905                 JNZ     RI15    ; YES - DATA IS READY
053E  CDB104                 CALL    DELAY   ; DELAY 1 MS
0541  05                     DCR     B       ; DECREMENT TIMER
0542  C23705                 JNZ     RI10    ; JUMP IF TIMER NOT EXPIRED
0545  AF                     XRA     A       ; ZERO A
0546  37                     STC             ; SET CARRY INDICATING EOF
```

```
0547   C1              POP     B       ; RESTORE BC
0548   C9              RET             ; RETURN TO CALLING ROUTINE
0549             RI15:
0549   DBEC            IN      CNIN    ; INPUT DATA CHARACTER
054B   B7              ORA     A       ; CLEAR CARRY
054C   C1              POP     B       ; RESTORE BC
054D   C9              RET             ; RETURN TO CALLING ROUTINE
                 ;
                 ;
                 ;*****************************************************************
                 ;
                 ;
054E             COPYRT:
054E   28432920        DB      '(C) 1976 INTEL CORP'
0552   31393736
0556   20494E54
055A   454C2043
055E   4F5250
                 ;
                 ;
                 ;*****************************************************************
                 ;
                 ;
3C00                    ORG     DATA
3C2E                    ORG     REGS    ; ORG TO REGISTER SAVE - STACK GOES IN HERE
                 ;
3C2E             MSTAK          EQU     $       ; START OF MONITOR STACK
3C2E   00        ESAVE:         DB      0       ; E REGISTER SAVE LOCATION
3C2F   00        DSAVE:         DB      0       ; D REGISTER SAVE LOCATION
3C30   00        CSAVE:         DB      0       ; C REGISTER SAVE LOCATION
3C31   00        BSAVE:         DB      0       ; B REGISTER SAVE LOCATION
3C32   00        FSAVE:         DB      0       ; FLAGS SAVE LOCATION
3C33   00        ASAVE:         DB      0       ; A REGISTER SAVE LOCATION
3C34   00        LSAVE:         DB      0       ; L REGISTER SAVE LOCATION
3C35   00        HSAVE:         DB      0       ; H REGISTER SAVE LOCATION
3C36   0000      PSAVE:         DW      0       ; PGM COUNTER SAVE LOCATION
3C38   0000      SSAVE:         DW      0       ; USER STACK POINTER SAVE LOCATION
3C3A   00        TEMP:          DB      0       ; TEMPORARY MONITOR CELL
                 ;
3C3D                    ORG     BRLOC           ; ORG TO USER BRANCH LOCATION
                 ;
3C3D             USRBR:         DS      3       ; BRANCH GOES IN HERE
                 ;
                 ;
                        END
NO PROGRAM ERRORS
```

SYMBOL TABLE

* 01

A	0007	ADRD	01A8	ADROU	01B1	ASAVE	3C33
B	0000	BRCHR	001B	BREAK	01C2	BRLOC	3C3D
BRTAB	03FA	BSAVE	3C31	BYTE	0496	C	0001
CADR	03A6	CI	01D5	CMD	0025	CNCTL	00ED
CNIN	00EC	CNOUT	00EC	CNVBN	01DF	CO	01E8
CONST	00ED	COPYR	054E *	CPYRT	03E3 *	CR	000D
CROUT	01F3	CSAVE	3C30	CTAB	03B8	D	0002
DATA	3C00	DCM05	0066	DCM10	006C	DCMD	005F
DEL1	04B4	DELAY	04B1	DSAVE	3C2F	E	0003
ECH05	0202	ECH10	0210	ECHO	01F9	ERROR	0212
ESAVE	3C2E	ESC	001B	EXIT	0217	FALSE	0F9C
FRET	021D	FSAVE	3C32	GCM05	00A0	GCM10	00A6
GCMD	008B	GETCH	0220	GETCM	002C	GETHX	0227
GETNM	025B	GHX05	022D	GHX10	0245	GNM05	0262
GNM10	0277	GNM15	0285	GNM20	028A	GNM25	0295
GNM30	0299	GO	0008 *	GTC03	003C	GTC05	0049
GTC10	0055	H	0004	HCHAR	000F	HIL05	02AF
HILO	02A0	HSAVE	3C35	ICM05	00B4	ICM10	00DB
ICM20	00E3	ICM25	00E9	ICMD	00A9	INUST	02B2
INVRT	00FF	L	0005	LE05	04BC	LEAD	04BA
LF	000A	LSAVE	3C34	LSGNO	0011	M	0006
MCM05	00F7	MCMD	00EF	MODE	00CF	MSGL	0023
MSTAK	3C2E	NCMDS	0008	NEWLN	000F	NMOUT	02C2
ONEMS	0083	PADR	04C6	PBYTE	04CF	PEOF	04E6
PEOL	0504	PO	050F	PRTY0	007F	PRVAL	02D5
PSAVE	3C36	PSW	0006	RBR	0002	RCM05	0412
RCM10	042F	RCMD	0406	REG05	02E2	REG10	02EC
REG15	0307	REGDS	02DF	REGS	3C2E	RGA05	0316
RGA10	0323	RGADR	0310	RI	051C	RI05	051D
RI07	052A	RI10	0537	RI15	0549	RICH	0513
RSTTF	0327	RSTU	0038	RTAB	03C0	RTABS	0003
SCM05	0114	SCM10	011F	SCM15	012F	SCMD	010F
SGNON	0395	SOMSG	001E	SP	0006	SRET	033B
SSAVE	3C38	STH05	035B	STHF0	033D	STHLF	0348
TEMP	3C3A	TERM	001B	TRDY	0001	TRUE	0F9F
TTYAD	0027	TXBE	0004	UPPER	00FF	USRBR	3C3D
VALDG	0367	VALDL	0382	WCM05	044B	WCM10	0460
WCM15	0463	WCM20	047B	WCM25	0490	WCMD	0441
XCM05	0145	XCM10	0154	XCM15	0161	XCM20	017F
XCM25	0196	XCM27	0197	XCM30	019F	XCMD	0133

* 02

* 03

* 04

* 05

* 06

* 07

* 08

* 09

* 10

* 11

* 12

* 13

APPENDIX F

ASCII TABLE

The INTELLEC®MDS uses a 7-bit ASCII code, which is the normal 8-bit ASCII code with the parity (high order) bit always reset.

GRAPHIC OR CONTROL	ASCII (HEXADECIMAL)	GRAPHIC OR CONTROL	ASCII (HEXADECIMAL)
NULL	00	ACK	7C
SOM	01	Alt. Mode	7D
EOA	02	Rubout	7F
EOM	03	!	21
EOT	04	"	22
WRU	05	#	23
RU	06	$	24
BELL	07	%	25
FE	08	&	26
H. Tab	09	'	27
Line Feed	0A	(	28
V. Tab	0B	)	29
Form	0C	*	2A
Return	0D	+	2B
SO	0E	,	2C
SI	0F	–	2D
DCO	10	.	2E
X-On	11	/	2F
Tape Aux. On	12	:	3A
X-Off	13	;	3B
Tape Aux. Off	14	<	3C
Error	15	=	3D
Sync	16	>	3E
LEM	17	?	3F
SO	18	[	5B
S1	19	/	5C
S2	1A	]	5D
S3	1B	↑	5E
S4	1C	←	5F
S5	1D	@	40
S6	1E	blank	20
S7	1F	0	30

APPENDIX G

BINARY–DECIMAL–HEXADECIMAL CONVERSION TABLES

HEXADECIMAL ARITHMETIC

ADDITION TABLE

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10
2	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11
3	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12
4	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13
5	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14
6	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15
7	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16
8	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17
9	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18
A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19
B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A
C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

MULTIPLICATION TABLE

1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	04	06	08	0A	0C	0E	10	12	14	16	18	1A	1C	1E
3	06	09	0C	0F	12	15	18	1B	1E	21	24	27	2A	2D
4	08	0C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	0A	0F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	0C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	0E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69
8	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87
A	14	1E	28	32	30	46	50	5A	64	6E	78	82	8C	96
B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5
C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4
D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3
E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2
F	1E	2D	3C	48	5A	69	78	87	96	A5	B4	C3	D2	E1

POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 347 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 916 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 726 776	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5
2 199 023 255 552	41	0.000 000 000 000 454 747 350 886 464 118 957 519 521 25
4 398 046 511 104	42	0.000 000 000 000 227 373 675 443 232 059 478 759 765 625
8 796 093 022 208	43	0.000 000 000 000 113 688 837 721 616 029 739 379 882 812 5
17 592 186 044 416	44	0.000 000 000 000 056 843 418 860 808 014 869 941 406 25
35 184 372 088 832	45	0.000 000 000 000 028 421 709 430 404 007 434 844 970 703 125
70 368 744 177 664	46	0.000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5
140 737 488 355 328	47	0.000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25
281 474 976 710 656	48	0.000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625
562 940 953 421 213	49	0.000 000 000 000 001 776 866 839 499 259 464 677 810 668 945 312 5
1 125 899 906 842 624	50	0.000 000 000 000 000 888 178 419 700 125 232 338 905 334 472 656 25
2 251 799 813 685 248	51	0.000 000 000 000 000 444 089 209 850 062 616 169 452 667 236 328 125
4 503 599 627 370 496	52	0.000 000 000 000 000 222 044 604 925 031 308 084 726 333 618 164 062 5
9 007 199 254 740 992	53	0.000 000 000 000 000 111 022 302 462 515 654 042 363 166 809 082 031 25
18 014 398 509 481 984	54	0.000 000 000 000 000 055 511 151 231 257 827 021 181 583 404 541 015 625
36 028 797 018 963 968	55	0.000 000 000 000 000 027 755 575 615 628 913 510 590 791 702 270 507 812 5
72 057 594 037 927 936	56	0.000 000 000 000 000 013 877 787 807 814 456 755 295 395 851 135 253 906 25
144 115 188 075 855 872	57	0.000 000 000 000 000 006 938 893 903 907 228 377 647 697 925 567 676 950 125
288 230 376 151 711 744	58	0.000 000 000 000 000 003 469 446 951 953 614 188 823 848 962 783 813 476 562 5
576 460 752 303 423 488	59	0.000 000 000 000 000 001 734 723 475 976 807 094 411 924 481 391 906 738 281 25
1 152 921 504 606 846 976	60	0.000 000 000 000 000 000 867 361 737 988 403 547 205 962 240 695 953 369 140 625
2 305 843 009 213 693 952	61	0.000 000 000 000 000 000 433 680 868 994 201 773 602 981 120 347 976 684 570 312 5
4 611 686 018 427 387 904	62	0.000 000 000 000 000 000 216 840 434 497 100 886 801 490 560 173 988 342 285 156 25
9 223 372 036 854 775 808	63	0.000 000 000 000 000 000 108 420 217 248 550 443 400 745 280 086 994 171 142 578 125

TABLE OF POWERS OF SIXTEEN$_{10}$

16^n_n	n	16^{-n}
++1	0	0.10000 00000 00000 00000 × 10
16	1	0.62500 00000 00000 00000 × 10^{-1}
256	2	0.39062 50000 00000 00000 × 10^{-2}
4 096	3	0.24414 06250 00000 00000 × 10^{-3}
65 536	4	0.15258 78906 25000 00000 × 10^{-4}
1 048 576	5	0.95367 43164 06250 00000 × 10^{-6}
16 777 216	6	0.59604 64477 53906 25000 × 10^{-7}
268 435 456	7	0.37252 90298 46191 40625 × 10^{-8}
4 294 967 296	8	0.23283 06436 53869 62891 × 10^{-9}
68 719 476 736	9	0.14551 91522 83668 51807 × 10^{-10}
1 099 511 627 776	10	0.90949 47017 72928 23792 × 10^{-12}
17 592 186 044 416	11	0.56843 41886 08080 14870 × 10^{-13}
281 474 976 710 656	12	0.35527 13678 80050 09294 × 10^{-14}
4 503 599 627 370 496	13	0.22204 46049 25031 30808 × 10^{-15}
72 057 594 037 927 936	14	0.13877 78780 78144 56755 × 10^{-16}
1 152 921 504 606 846 976	15	0.86736 17379 88403 54721 × 10^{-18}

TABLE OF POWERS OF 10_{16}

10^n	n	10^{-n}
1	0	1.0000 0000 0000 0000
A	1	0.1999 9999 9999 999A
64	2	0.28F5 C28F 5C28 F5C3 × 16^{-1}
3E8	3	0.4189 374B C6A7 EF9E × 16^{-2}
2710	4	0.68DB 8BAC 710C B290 × 16^{-3}
1 86A0	5	0 A7C5 AC47 1B47 8423 × 16^{-4}
F 4240	6	0.10C7 F7A0 B5ED 8D37 × 16^{-4}
98 9680	7	0.1AD7 F29A BCAF 4858 × 16^{-5}
5F5 E100	8	0.2AF3 1DC4 6118 73BF × 16^{-6}
3B9A CA00	9	0.44B8 2FA0 9B5A 52CC × 16^{-7}
2 540B E400	10	0.6DF3 7F67 SEF0 EADF × 16^{-8}
17 4876 E800	11	0.AFEB FF0B CB24 AAFF × 16^{-9}
E8 D4A5 1000	12	0.1197 9981 2DEA 1119 × 16^{-9}
918 4E72 A000	13	0.1025 C268 4976 81C2 × 16^{-10}
5AF3 107A 4000	14	0.2D09 370D 4257 3604 × 16^{-11}
3 8D7E A4C6 3000	15	0.480E BE7B 9D58 566D × 16^{-12}
23 8652 6FC1 0000	16	0.734A CA5F 6226 F0AE × 16^{-13}
163 4578 5D8A 0000	17	0.B877 AA32 36A4 B449 × 16^{-14}
DE0 B6B3 A764 0000	18	0.1272 5DD1 D243 ABA1 × 16^{-15}
8AC7 2304 89E8 0000	19	0.1D83 C94F B6D2 AC35 × 16^{-15}

HEXADECIMAL-DECIMAL INTEGER CONVERSION

The table below provides for direct conversions between hexadecimal integers in the range 0—FFF and decimal integers in the range 0—4095. For conversions of larger integers, the table values may be added to the following figures:

HEXADECIMAL	DECIMAL	HEXADECIMAL	DECIMAL
01 000	4 096	20 000	131 072
02 000	8 192	30 000	196 608
03 000	12 288	40 000	262 144
04 000	16 384	50 000	327 680
05 000	20 480	60 000	393 216
06 000	24 576	70 000	458 752
07 000	28 672	80 000	524 288
08 000	32 768	90 000	589 824
09 000	36 864	A0 000	655 360
0A 000	40 960	B0 000	720 896
0B 000	45 056	C0 000	786 432
0C 000	49 152	D0 000	851 968
0D 000	53 248	E0 000	917 504
0E 000	57 344	F0 000	933 040
0F 000	61 440	100 000	1 048 576
10 000	65 536	200 000	2 097 152
11 000	69 632	300 000	3 145 728
12 000	73 728	400 000	4 194 304
13 000	77 824	500 000	5 242 880
14 000	81 920	600 000	6 291 456
15 000	86 016	700 000	7 340 032
16 000	90 112	800 000	8 388 608
17 000	94 208	900 000	9 437 184
18 000	98 304	A00 000	10 485 760
19 000	102 400	B00 000	11 534 336
1A 000	106 496	C00 000	12 582 912
1B 000	110 592	D00 000	13 631 489
1C 000	114 638	E00 000	14 680 064
1D 000	118 784	F00 000	15 728 640
1E 000	122 880	1 000 000	16 777 216
1F 000	126 976	2 000 000	33 554 432

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
010	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
020	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
030	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
040	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
050	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
060	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
070	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
080	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
090	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	1055	0156	0157	0158	0159
0A0	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0B0	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0C0	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	2007
0D0	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0E0	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0F0	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255

HEXADECIMAL-DECIMAL INTEGER CONVERSION (continued)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
100	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271
110	0272	0273	0274	0275	0276	0277	0278	0279	0280	0281	0282	0283	0284	0285	0286	0287
120	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	0301	0302	0303
130	0304	0305	0306	0307	0308	0309	0310	0311	0312	0313	0314	0315	0316	0317	0318	0319
140	0320	0321	0322	0323	0324	0325	0326	0327	0328	0329	0330	0331	0331	0333	0334	0335
150	0336	0337	0338	0339	0340	0341	0342	0343	0344	0345	0346	0347	0348	0349	0350	0351
160	0352	0353	0354	0355	0356	0357	0358	0359	0360	0361	0362	0363	0364	0365	0366	0367
170	0368	0369	0370	0371	0372	0373	0374	0375	0376	0377	0378	0379	0380	0381	0382	0383
180	0384	0385	0386	0387	0388	0389	0390	0391	0392	0393	0394	0395	0396	0397	0398	0399
190	0400	0401	0402	0403	0404	0405	0406	0407	0408	0409	0410	0411	0412	0413	0414	0415
1A0	0416	0417	0418	0419	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	0430	0431
1B0	0432	0433	0434	0435	0436	0437	0438	0439	0440	0441	0442	0443	0444	0445	0446	0447
1C0	0448	0449	0450	0451	0452	0453	0454	0455	0456	0457	0458	0459	0460	0461	0462	0463
1D0	0464	0465	0466	0467	0468	0469	0470	0471	0472	0473	0474	0475	0476	0477	0478	0479
1E0	0480	0481	0482	0483	0484	0485	0486	0487	0488	0489	0490	0491	0492	0493	0494	0495
1F0	0496	0497	0498	0499	0500	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511
200	0512	0513	0514	0515	0516	0517	0518	0519	0520	0521	0522	0523	0524	0525	0526	0527
210	0528	0529	0530	0531	0532	0533	0534	0535	0536	0537	0538	0539	0540	0541	0542	0543
220	0544	0545	0546	0547	0548	0549	0550	0551	0552	0553	0554	0555	0556	0557	0558	0559
230	0560	0561	0562	0563	0564	0565	0566	0567	0568	0569	0570	0571	0572	0573	0574	0575
240	0576	0577	0578	0579	0580	0581	0582	0583	0584	0585	0586	0587	0588	0589	0590	0591
250	0592	0593	0594	0595	0596	0597	0598	0599	0600	0601	0602	0603	0604	0605	0606	0607
260	0608	0609	0610	0611	0612	0613	0614	0615	0616	0617	0618	0619	0620	0621	0622	0623
270	0624	0625	0626	0627	0628	0629	0630	0631	0632	0633	0634	0635	0636	0637	0638	0639
280	0640	0641	0642	0643	0644	0645	0646	0647	0648	0649	0650	0651	0652	0653	0654	0655
290	0656	0657	0658	0659	0660	0661	0662	0663	0664	0665	0666	0667	0668	0669	0670	0671
2A0	0672	0673	0674	0675	0676	0677	0678	0679	0680	0681	0682	0683	0684	0685	0686	0687
2B0	0688	0689	0690	0691	0692	0693	0694	0695	0696	0697	0698	0699	0700	0701	0702	0703
2C0	0704	0705	0706	0707	0708	0709	0710	0711	0712	0713	0714	0715	0716	0717	0718	0719
2D0	0720	0721	0722	0723	0724	0725	0726	0727	0728	0729	0730	0731	0732	0733	0734	0735
2E0	0736	0738	0738	0739	0740	0741	0742	0743	0744	0745	0746	0747	0748	0749	0750	0751
2F0	0752	0753	0754	0755	0756	0757	0758	0759	0760	0761	0762	0763	0764	0765	0766	0767
300	0768	0769	0770	0771	0772	0773	0774	0775	0776	0777	0778	0779	0780	0781	0782	0783
310	0784	0785	0786	0787	0788	0789	0790	0791	0792	0793	0794	0795	0796	0797	0798	0799
320	0800	0801	0802	0803	0804	0805	0806	0807	0808	0809	0810	0811	0812	0813	0814	0815
330	0816	0817	0818	0819	0820	0821	0822	0823	0824	0825	0826	0827	0828	0829	0830	0831
340	0832	0833	0834	0835	0836	0837	0838	0839	0840	0841	0842	0843	0844	0845	0846	0847
350	0848	0849	0850	0851	0852	0853	0854	0855	0856	0857	0858	0859	0860	0861	0862	0863
360	0864	0865	0866	0867	0868	0869	0870	0871	0872	0873	0874	0875	0876	0877	0878	0879
370	0880	0881	0882	0883	0884	0885	0886	0887	0888	0889	0890	0891	0892	0893	0894	0895
380	0896	0897	0898	0899	0900	0901	0902	0903	0904	0905	0906	0907	0908	0909	0910	0911
390	0912	0913	0914	0915	0916	0917	0918	0919	0920	0921	0922	0923	0924	0925	0926	0927
3A0	0928	0929	0930	0931	0932	0933	0934	0935	0936	0937	0938	0939	0940	0941	0942	0943
3B0	0944	0945	0946	0947	0948	0949	0950	0951	0952	0953	0954	0955	0956	0957	0958	0959
3C0	0960	0961	0962	0963	0964	0965	0966	0967	0968	0969	0970	0971	0972	0973	0974	0975
3D0	0976	0977	0978	0979	0980	0981	0982	0983	0984	0985	0986	0987	0988	0989	0990	0991
3E0	0992	0993	0994	0995	0996	0997	0998	0999	1000	1001	1002	1003	1004	1005	1006	1007
3F0	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023

HEXADECIMAL-DECIMAL INTEGER CONVERSION (continued)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
400	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
410	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
420	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
430	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
440	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
450	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
460	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
470	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
480	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1163	1165	1166	1167
490	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A0	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B0	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C0	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D0	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E0	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1263	1263
4F0	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279
500	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
510	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
520	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
530	1382	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
540	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
550	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	11372	1373	1374	1375
560	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
570	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
580	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
590	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A0	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B0	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C0	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D0	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E0	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F0	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535
600	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
610	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
620	1568	1569	1570	1571	1572	1573	1574	1575	1576n	1577	1578	1579	1580	1581	1582	1583
630	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
640	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
650	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
660	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
670	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
680	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
690	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A0	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B0	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C0	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D0	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E0	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F0	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791

HEXADECIMAL-DECIMAL INTEGER CONVERSION (continued)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
700	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
710	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
720	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
730	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
740	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
750	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
760	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
770	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
780	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
790	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A0	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B0	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C0	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D0	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E0	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F0	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047
800	2048	2049	2050	2051	2502	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
810	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
820	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
830	2096	2097	2098	2099	2100	2101	2102	2103	2104	2015	2106	2107	2108	2109	2110	2111
840	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
850	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
860	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
870	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
880	2176	21777	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
890	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A0	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B0	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C0	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D0	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E0	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F0	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303
900	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
910	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
920	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
930	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
940	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
950	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
960	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
970	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
980	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
990	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A0	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B0	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C0	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D0	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E0	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F0	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559

HEXADECIMAL-DECIMAL INTEGER CONVERSION (continued)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A00	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
A10	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A20	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A30	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A40	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A50	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A60	2656	2657	2658	2659	2660	2661	2622	2663	2664	2665	2666	2667	2668	2669	2670	2671
A70	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687
A80	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A90	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA0	2720	2721	2722	2723	2724	2725	2726	27272	2728	2729	2730	2731	2732	2733	2734	2735
AB0	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC0	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD0	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE0	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF0	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815
B00	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B10	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B20	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B30	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B40	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B50	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B60	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
B70	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
B80	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B90	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA0	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB0	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC0	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD0	3024	3025	3026	3027	3028	3029	3030	3031	3032	3933	3034	3035	3036	3037	3038	3039
BE0	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF0	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071
C00	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C10	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C20	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C30	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C40	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C50	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C60	3168	3168	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C70	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C80	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C90	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA0	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB0	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
CC0	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
CD0	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE0	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF0	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327

HEXADECIMAL-DECIMAL INTEGER CONVERSION (continued)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
D00	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D10	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D20	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D30	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D40	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D50	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D60	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D70	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D80	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D90	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA0	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB0	3504	3505	3506	3507	3508	3509	3410	3511	3512	3513	3514	3515	1516	3517	3518	3519
DC0	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
CC0	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE0	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF0	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583
E00	3584	3585	3586	3587	3588	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
E10	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E20	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
E30	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E40	3648	3648	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E50	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E60	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E70	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3070	3708	3709	3710	3711
E80	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E90	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA0	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB0	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC0	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED0	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE0	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF0	3824	3825	3826	3827	3828	3829	3030	3831	3832	3833	3834	3835	3836	3837	3838	3839
F00	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F10	3856	3857	3858	3859	3860	3861	3862	3863	3865	3865	3866	3867	3868	3869	3870	3871
F20	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F30	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F40	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F50	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F60	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F70	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F80	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F90	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA0	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB0	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC0	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD0	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE0	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077	4078	4079
FF0	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095

APPENDIX H
TELETYPEWRITER MODIFICATIONS

H-1. INTRODUCTION

This appendix provides information required to modify a Model ASR-33 Teletypewriter for use with certain Intel SBC 80 computer systems.

H-2. INTERNAL MODIFICATIONS

WARNING

> Hazardous voltages are exposed when the top cover of the teletypewriter is removed. To prevent accidental shock, disconnect the teleprinter power cord before proceeding beyond this point.

Remove the top cover and modify the teletypewriter as follows:

a. Remove blue lead from 750-ohm tap on current source register; reconnect this lead to 1450-ohm tap. (Refer to figures H-1 and H-2.)

b. On terminal block, change two wires as follows to create an internal full-duplex loop (refer to figures H-1 and H-3):

   1. Remove brown/yellow lead from terminal 3; reconnect this lead to terminal 5.
   2. Remove white/blue lead from terminal 4; reconnect this lead to terminal 5.

c. On terminal block, remove violet lead from terminal 8; reconnect this lead to terminal 9. This changes the receiver current level from 60 mA to 20 mA.

A relay circuit card must be fabricated and connected to the paper tape reader driver circuit. The relay circuit card to be fabricated requires a relay, a diode, a thyractor, a small 'vector' board for mounting the components, and suitable hardware for mounting the assembled relay card.

A circuit diagram of the relay circuit card is included in figure H-4; this diagram also includes the part numbers of the relay, diode, and thyractor. (Note that a 470-ohm resistor and a 0.1 μF capacitor may be substituted for the thyractor.) After the relay circuit card has been assembled, mount it in position as shown in figure H-5. Secure the card to the base plate using two self-tapping screws. Connect the relay circuit to the distributor trip magnet and mode switch as follows:

a. Refer to figure H-4 and connect a wire (Wire 'A') from relay circuit card to terminal L2 on mode switch. (See figure H-6.)

b. Disconnect brown wire shown in figure H-7 from plastic connector. Connect this brown wire to terminal l2 on mode switch. (Brown wire will have to be extended.)

c. Refer to figure H-4 and connect a wire (Wire 'B') from relay circuit board to terminal L1 on mode switch.

H-3. EXTERNAL CONNECTIONS

Connect a two-wire receive loop, a two-wire send loop, and a two-wire tape reader control loop to the external device as shown in figure H-4. The external connector pin numbers shown in figure H-4 are for interface with an RS232C device.

H-4. SBC 530 TTY ADAPTER

The SBC 530, which converts RS232C signal levels to an optically isolated 20 mA current loop interface, provides signal translation for transmitted data, received data, and a paper tape reader relay. The SBC 530 interfaces an Intel SBC 80 computer system to a teletypewriter as shown in figure H-8.

The SBC 530 requires +12V at 98 mA and −12V at 98 mA. An auxiliary supply must be used if the SBC 80 system does not supply this power. A schematic diagram of the SBC 530 is supplied with the unit. The following auxiliary power connector (or equivalent) must be procured by the user:

Connector, Molex 09-50-7071
Pins, Molex 08-50-0106
Polarizing Key, Molex 15-04-0219

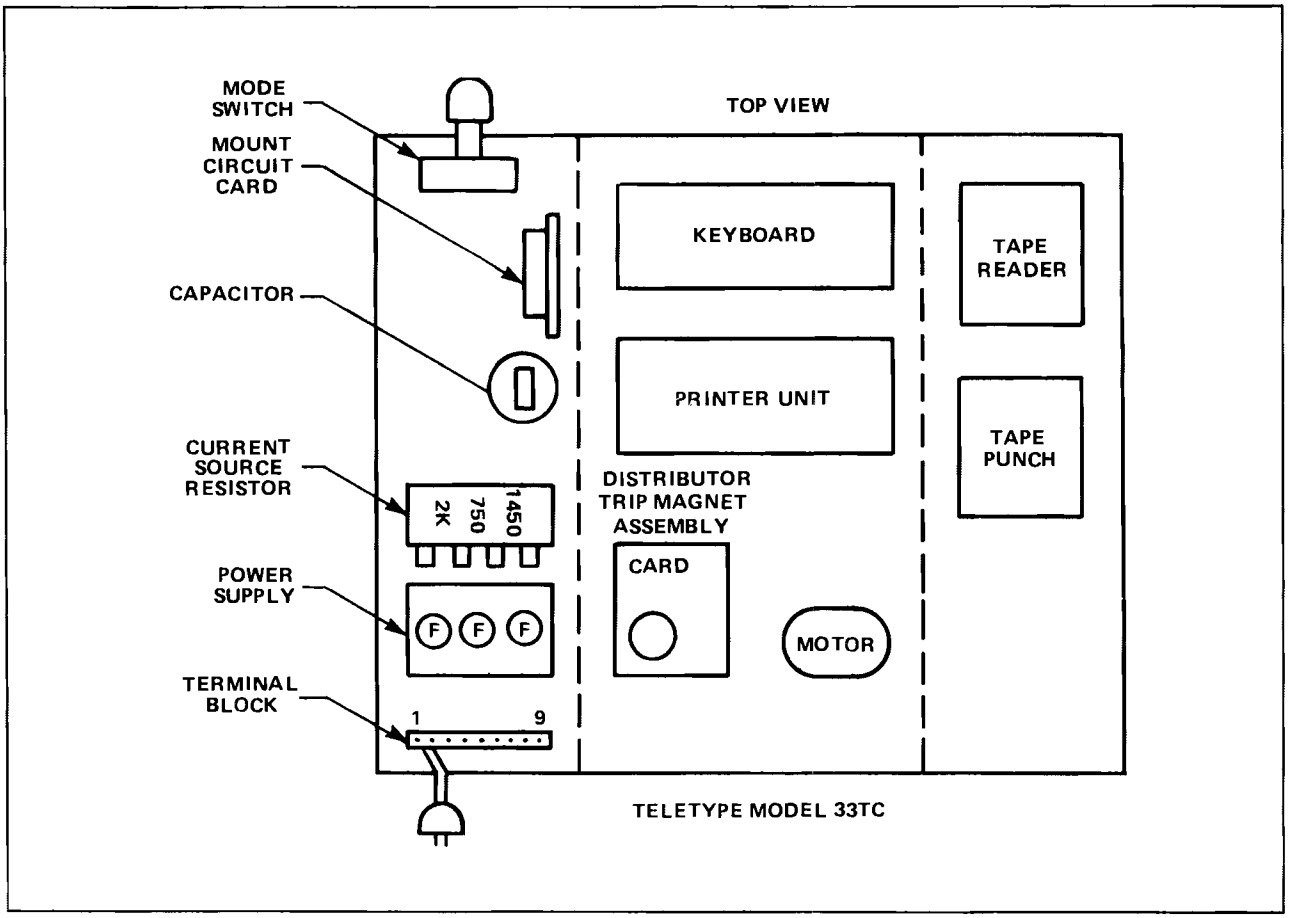


Figure H-1. Teletype Component Layout

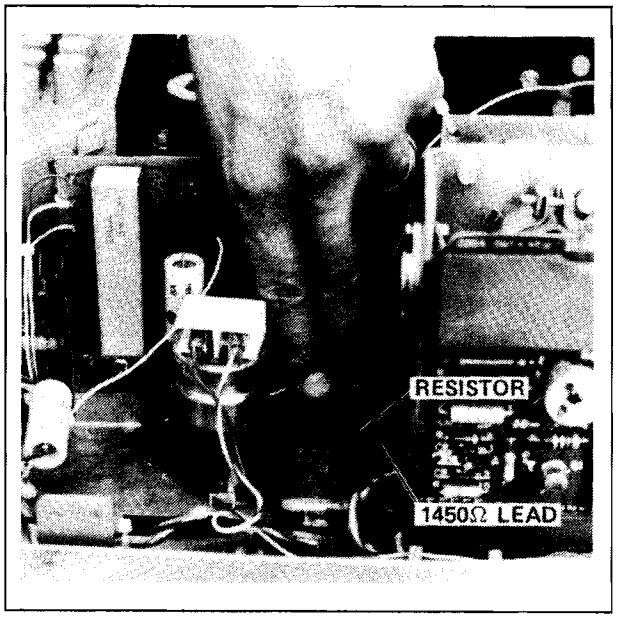


Figure H-2. Current Source Resistor

Figure H-3. Terminal Block

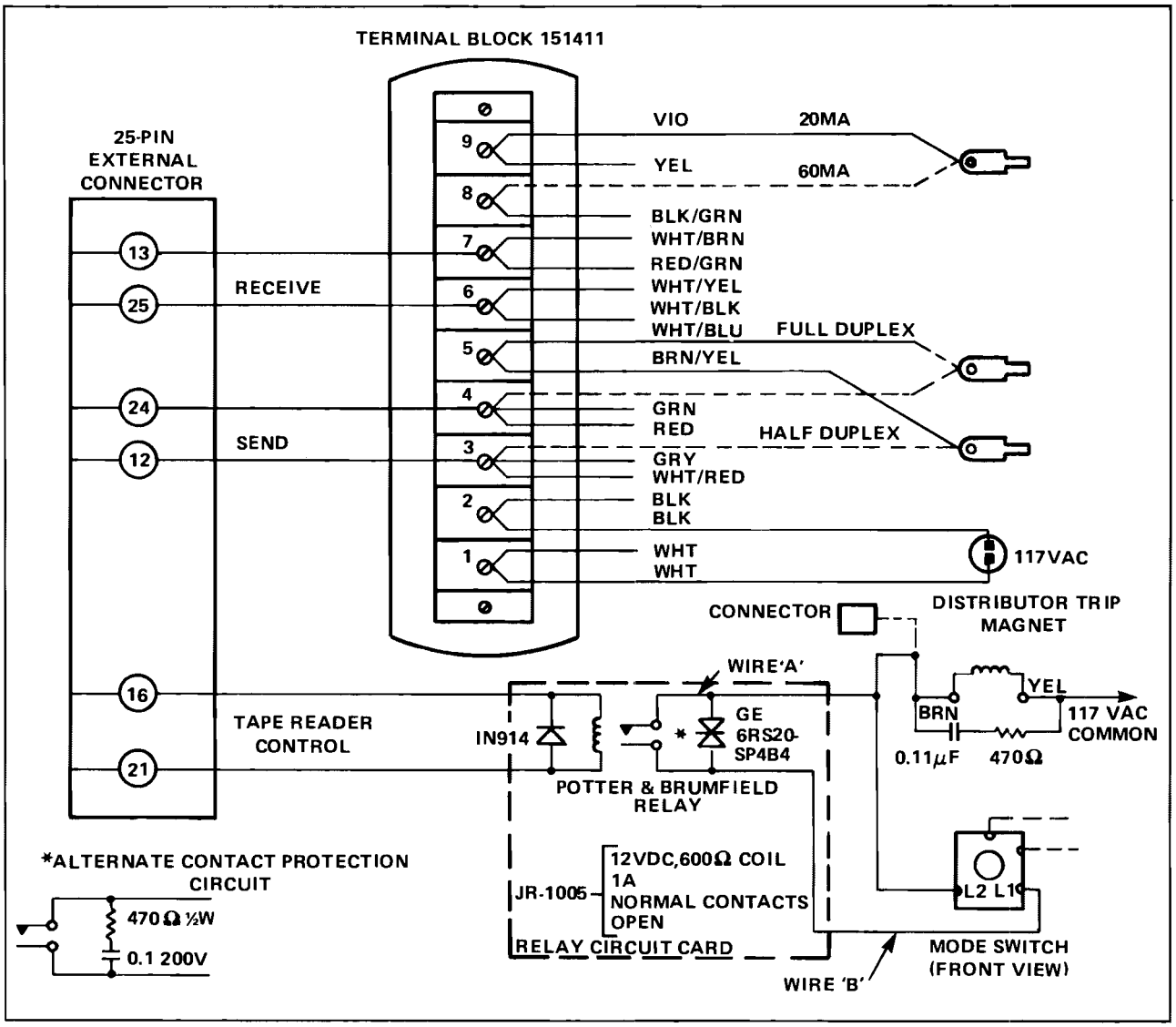


Figure H-4. Teletypewriter Modifications

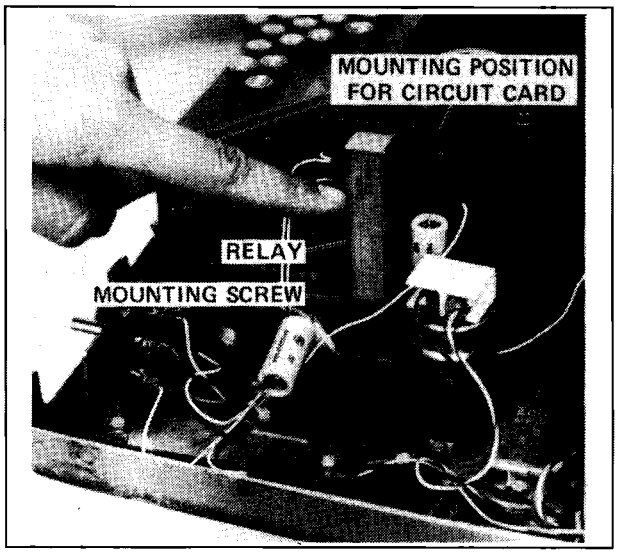


Figure H-5. Relay Circuit

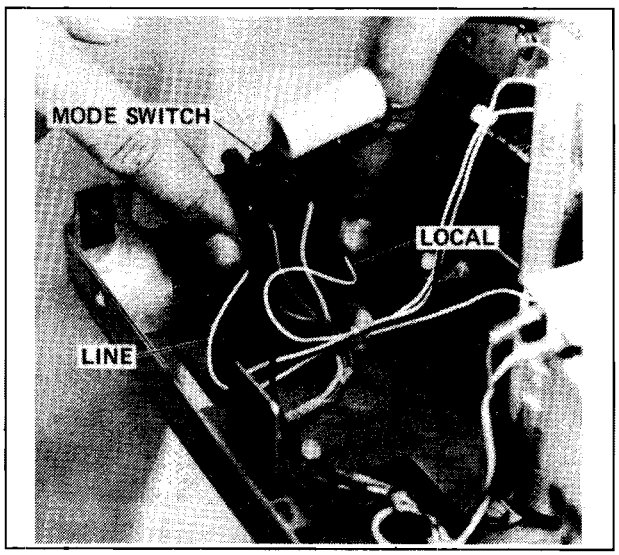


Figure H-6. Mode Switch

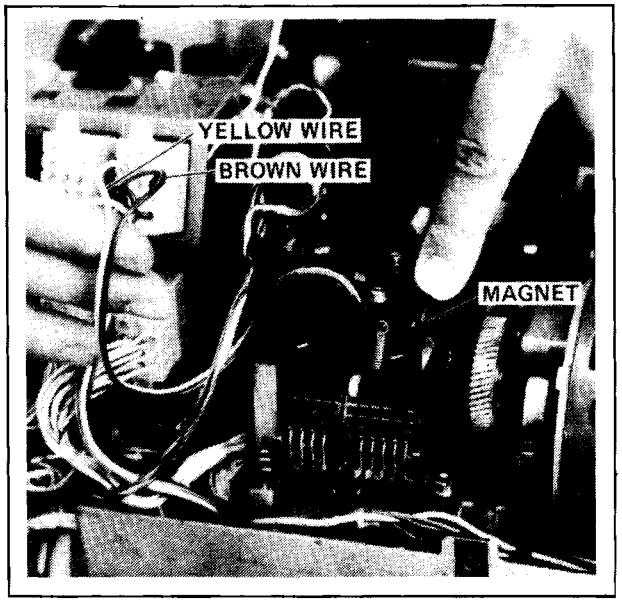


Figure H-7. Distributor Trip Magnet

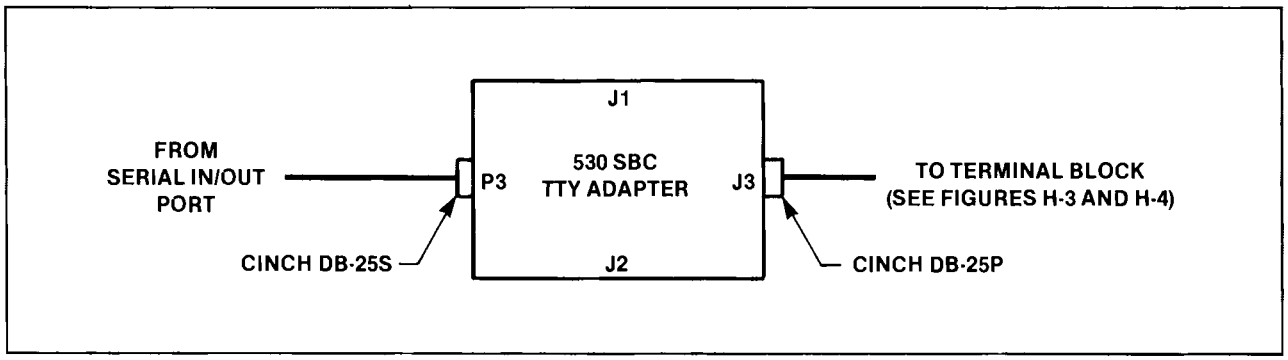


Figure H-8. TTY Adapter Cabling

REQUEST FOR READER'S COMMENTS

The Microcomputer Division Technical Publications Department attempts to provide documents that meet the needs of all Intel product users. This form lets you participate directly in the documentation process.

Please restrict your comments to the usability, accuracy, readability, organization, and completeness of this document.

1. Please specify by page any errors you found in this manual.

2. Does the document cover the information you expected or required? Please make suggestions for improvement.

3. Is this the right type of document for your needs? Is it at the right level? What other types of documents are needed?

4. Did you have any difficulty understanding descriptions or wording? Where?

5. Please rate this document on a scale of 1 to 10 with 10 being the best rating. _____

NAME _____ DATE _____
TITLE _____
COMPANY NAME/DEPARTMENT _____
ADDRESS _____
CITY _____ STATE _____ ZIP CODE _____

Please check here if you require a written reply. ☐

WE'D LIKE YOUR COMMENTS . . .

This document is one of a series describing Intel products. Your comments on the back of this form will help us produce better manuals. Each reply will be carefully reviewed by the responsible person. All comments and suggestions become the property of Intel Corporation.

First Class
Permit No. 1040
Santa Clara, CA

BUSINESS REPLY MAIL

No Postage Stamp Necessary if Mailed in U.S.A.

Postage will be paid by:

Intel Corporation
3065 Bowers Avenue
Santa Clara, CA 95051

Attention: MCD Technical Publications