The 8255A Programmable Peripheral Interface\(^4\) provides programmable I/O ports exclusively.\([3][4]\) It can be used with the 8085A or any of a number of other microprocessors. The 8255A contains a control register, a status register, and three 8-bit I/O ports: A, B, and C, as shown in Fig. 8.6-6. Port C is actually two separately programmable ports: C-upper (C\(_4\)–C\(_7\)) and C-lower (C\(_0\)–C\(_3\)). An 8-bit data bus buffer transfers data between the external data bus and the control register, status register, or one of the I/O ports.

The 8255A is selected by a low signal at its chip select input, CS. When not selected, the data bus buffers which connect the 8255A to the system data bus are floated. The source of the CS signal depends on whether isolated or memory mapped I/O is used. For isolated I/O, bits A\(_2\) to A\(_7\) are decoded to provide CS, and bits A\(_1\) and A\(_0\) are used for selecting the control register, status register, or one of the ports. If the six address bits, A\(_2\)–A\(_7\), are exhaustively decoded, as many as sixty-four 8255As can be used in a system. With isolated I/O, selection of the 8255A is further conditioned on \(IO/M = 1\).

Linear selection saves decoders and can select six 8255As, using isolated I/O with device and port selection as shown below.

```
  A\(_{15}\) A\(_{14}\) A\(_{13}\) A\(_{12}\) A\(_{11}\) A\(_{10}\) A\(_9\) A\(_8\)
  A\(_7\) A\(_6\) A\(_5\) A\(_4\) A\(_3\) A\(_2\) A\(_1\) A\(_0\)
  PORT SELECTS
-----------
8255A DEVICE SELECTS
```

However, in order to condition the selection to isolated I/O, \(\overline{I/OR}\) is connected to the RD input of the 8255A, and \(\overline{I/O\bar{W}}\) to the WR input of the 8255A.

With memory mapped I/O, \(\overline{MEMR}\) is connected to the RD input of the 8255A and \(\overline{MEMW}\) is connected to the WR input of the 8255A. And the address bits A\(_2\)–A\(_{15}\) are decoded to provide the CS signal.

In both the isolated I/O and memory mapped schemes, when the 8255A is selected, inputs A\(_0\) and A\(_1\), in turn, select the control register or one of the ports (A, B, or C) for the data transfer (see Table 8.6-1).

At system power up, a reset signal applied to the 8255A floats all 24 pins associated with the three I/O ports. The 8255A stays in this condition until the

\(^4\)This overview of the 8255A provides an insight into the capability and flexibility of programmable LSI ports. Consult the manufacturer's literature for more details.
application program writes a word into the control register which defines the 8255A's subsequent mode of operation. The three basic modes of operation are:

1. Mode 0: basic input-output.

The mode definition format of the control word is shown in Fig. 8.6-7.

Mode 0 provides two 8-bit ports (A and B) and two 4-bit ports (C-upper and C-lower). Any port can be input or output; outputs are latched, inputs are not. There are 16 possible input-output configurations in this mode. For example, the control word 8AH sets port A for output, port C-upper for input, port C-lower for output, and port B for input. An 8255A used for isolated I/O and selected when A₂ to A₇ = 0, is initialized by the following instructions to the above configuration.

```
MVI A,8AH  ; load A with control word
OUT 03H    ; write control word into control
            ; register of 8255A
```
Mode 1 also provides two 8-bit ports, A and B, but here both inputs and outputs are latched. The two 4-bit ports (C) provide handshaking for ports A and B. Figure 8.6-8 shows the configuration of the 8255A for input and output in mode 1 operation along with the necessary control word.

For input in mode 1, port A is an input port, and bits C₃, C₄, and C₅ are used for associated handshaking. Port B is an input port with C₀, C₁, and C₂ used for handshaking. C₆ and C₇ can be used as input or output ports. The input device places 8 bits of data at A₀–A₇ (or B₀–B₇), then generates an active low strobe, STB, which loads data into the input latch. This makes the input buffer full signal, IBF, logic 1. The microprocessor reads port C and checks the IBF signal to determine whether data is available for input to the microprocessor. The mode 1 status word format is shown in Fig. 8.6-9. If IBF is logic 1, the microprocessor reads port A (or B), which inputs the data and resets the IBF flag.

For output in mode 1, the microprocessor writes data to port A (or B), and the output buffer full flag, OBF, goes low to indicate this. The output device monitors OBF to determine when output data is available. And, it acknowledges acceptance of the data by bringing the acknowledge input, ACK, low, thus clearing the output buffer full flag.

Mode 2 provides a single 8-bit bidirectional bus: port A. Five bits of port C are used for status and control of port A, thus providing a handshaking
Figure 8.6-7. Mode definition format of control word for 8255A. (Courtesy of Intel Corp.)

capability similar to that of mode 1. Figure 8.6-10 illustrates the control word and the configuration of the 8255A in mode 2. The format of the status word when the 8255A is in mode 2 is shown in Fig. 8.6-11.

Various combinations of mode operation are possible. For example, while port A and C5 to C7 are used for bidirectional data transfer with handshaking in mode 2, port B can be used for input in mode 0.

The 8255A also has a bit set/reset capability for port C. When bit 7 of the control word is 0, the control word is interpreted by the 8255A as a port C bit set/reset command. Any bit of port C can be set or cleared. The bit set/reset format is shown in Fig. 8.6-12. The ability to directly set or reset a single bit is advantageous in applications where individual bits control separate external functions.
Figure 8.6-8. Mode 1 (a) input and (b) output configuration for an 8255A. (Courtesy of Intel Corp.)

Figure 8.6-9. Mode 1 status word for 8255A. (Courtesy of Intel Corp.)

Figure 8.6-10. Mode 2 bidirectional port configuration for an 8255A. (Courtesy of Intel Corp.)
8.7 SERIAL TRANSFER OF INFORMATION

Although the data bus of a microprocessor is designed to transfer data to and from I/O devices in parallel—all bits of a data word being transferred simultaneously—there are cases when it is preferable to transfer data serially (1 bit at a time). Data transferred serially is often sent in groups of bits which constitute a character or word. Frequently, the characters are coded in ASCII. **Serial transfer** requires only one signal line or communications channel and is appropriate when:

1. The I/O device to/from which the data is transferred is inherently serial in operation.
2. The distance between the microprocessor and the I/O device is great.

Many I/O devices such as Teletypes and magnetic tape cartridges and cassettes are constrained by their design to receive or transmit data a bit at a time and thus operate in a serial fashion.

As the distance between a microprocessor and an I/O device increases, the cost differential between running a cable with a number of conductors equal to the data bus width and running a single cable becomes very significant. Not only is the multiple conductor cable more costly, but line drivers and receivers are necessary. A point is reached beyond which it becomes more economical to
use serial data transfer, even though it requires additional hardware and/or software, than to use a multiple conductor cable. In other applications, the distance may be so great that common carrier facilities such as telephone lines are required, and thus the data must be transmitted in serial form. In general, an interface is required between the microprocessor and an I/O device for serial data transfer (see Fig. 8.7-1). The interface provides two functions:

1. The logical formatting of data, including serial-to-parallel/parallel-to-serial conversion.

2. Translation of logic signals to the electrical signals appropriate for transmitting data over the communications channel connecting the microprocessor and I/O device.

Voltage and current levels used for data communication are seldom T²L compatible. Electrical signal translation is implemented by hardware. Logical formatting of data, however, may be implemented by software or hardware or by a combination of the two.

Serial data transfer systems are simplex, half duplex, or full duplex. In a simplex system, data is transferred only in one direction. In a half duplex system, it is transmitted in either direction, but in only one direction at a time. In a full duplex system, data is transferred in both directions simultaneously.

Consider the example of simplex serial data transfer shown in Fig. 8.7-2. A microprocessor transfers data to a shift register in an I/O device. The shift register is within several feet of the microprocessor, and the communications channel is a twisted pair of wires. The formatting of data is done by software. Data input to the shift register is connected to bit D₀ of the data bus, and the clock input of the shift register is driven by logic which generates an output device select pulse. Assuming that the data byte to be transferred is in A, the following subroutine formats the data and implements the transfer, least significant bit first:

```assembly
SO8:   MVI C,8
LOOP:  OUT SR
       RAR
       DCR C
       JNZ LOOP
       RET
```
This example illustrates an important aspect of serial data transfer: The receiver has to have some means of determining when its data input should be sampled, thus defining the occurrence of a new data bit. In this example a clock signal from the transmitter tells the receiving device when it should sample the data input line, thus synchronizing the data transfer. Use of the device select pulse for synchronization ensures that the data on the interconnecting line is sampled at the proper time. However, transferring a separate clock signal from the transmitter to the receiver for synchronization requires an additional connection between the devices.\(^5\)

The \textit{baud rate} is the rate at which data is transferred. It denotes the number of signal changes per second. When the signals being transmitted are binary, the baud rate is equivalent to the number of bits per second. Communications channels are rated by baud rate. If data is transmitted on a channel at a baud rate beyond the channel’s capacity, the error rate of the transmission is unacceptably high.

Serial transfer of data is asynchronous or synchronous. In asynchronous transmission, a character is sent whenever it is available. Thus, the time interval between two characters is variable; however, the time interval between bits in a single character is fixed. When no character is available for transmission, the line is idle. With synchronous character transmission, one character is followed immediately by another. Whenever another data character is not immediately ready for transmission, the transmitter repeatedly sends a special SYNC character until it can transmit the next data character.

\subsection{8.7.1 Asynchronous Serial Character Transfer}

Asynchronous data transfer is used for low speed, low data rate transfers: Such transfers typically occur at 110, 300, 600, 1200, or 2400 baud, values commonly used by manufacturers of commercial communications equipment. The transfer of information between a teletypewriter and microprocessor is asynchronous. For instance Teletype’s Models 33 and 35, transmit and receive data at 110 baud

\(^5\)Techniques for the serial transfer of data which do not require a separate clock line from transmitter to receiver are considered in the following sections.
or 110 bits per second. At this rate, each bit duration or \textit{bit time} is 9.09 ms. Data is transferred as a group of serial bits constituting a character. The character is coded in ASCII as 7 bits, or \textit{seven level code}, and transmitted in the format shown in Fig. 8.7-3.

Even though the receiver and transmitter in an asynchronous data transfer are not synchronized with respect to the time at which a character is transmitted, once the transmitter starts to send a character, the receiver synchronizes itself with the bit times of the character in order to sample them at the correct time.

A \textit{start bit} synchronizes the transmitter and receiver. It is the first bit of any character sent and is logic 0. When no character is being sent, the transmitter’s output is logic 1, and the data line is \textit{idle} or \textit{marking}. The receiver synchronizes its operation with the transmitter on the 1 to 0 transition of the data line. It waits one-half a bit time, checks the input to make sure it is still logic 0— and therefore a valid start bit—and begins sampling the data line at intervals equal to one bit time. The baud rates of the transmitter and receiver are set to the same value, and the data line is sampled at the center of each transmitted bit. This eliminates errors which might occur if sampling takes place at the beginning of each bit time since the leading or trailing edges of transitions on the data line are distorted in transmission. Following the start bit, the seven data bits of the ASCII character are transmitted—least significant bit first—followed by a parity bit, which is set or cleared to provide even or odd parity. For odd (even) parity, the parity bit is set to make the total number of bits (data bits and parity bit) in the ASCII character odd (even). Finally, two stop bits are transmitted. The start bit, logic 0, and two stop bits, both logic 1, \textit{frame} the ASCII character.

On the microprocessor side, the logical formatting of data for serial transfer can be implemented in software, including the 9.09 ms delay and the other features required for transmitting or receiving. Hardware only provides electrical interfacing between the microprocessor and Teletype signals.

The disadvantage of software formatting and timing of a serial transfer is that the microprocessor is completely tied up during each character transfer. It takes 100 ms to transfer a single character at 110 baud; in this period of time, assuming an average instruction execution time of 2 \( \mu \text{S} \), 50,000 instructions could be executed.

Parallel-to-serial conversion for transmitting and serial-to-parallel conversion for receiving and formatting with hardware use the microprocessor’s time
more effectively. The microprocessor transfers data in parallel to external hardware, which provides the necessary formatting and parallel-to-serial conversion. This hardware also receives serial data, removes the parity and framing information, and supplies the data in parallel to the microprocessor.

The Universal Asynchronous Receiver Transmitter, UAR/T [5], a full duplex device in a single 40-pin package, provides all the logic for asynchronous data transfer. It supplies logical formatting; its input and output signals are T/L levels; and its receiver outputs are three-state. Additional circuitry may be necessary for the electrical interface, but no common clock signal is required between the UAR/T and the device with which it communicates. UAR/Ts which operate at rates up to 200 K baud are available (Intersil IM6402/6403). The UAR/T has become a de facto industry standard with several manufacturers providing pin compatible devices. Typically both the microprocessor and I/O device are connected in parallel to UAR/Ts which are interconnected to provide serial communication (see Fig. 8.7-4). A single UAR/T contains both a transmitter and a receiver. The transmitter and receiver portions of the UAR/T are independent except for the formatting of the words transmitted and received. The UAR/T inputs which program the device for number of data bits (5 to 8), parity (even, odd, none), and number of stop bits (1 or 2) apply to both transmitter and receiver portions of the system. The baud rate can be different for the receiver and transmitter and is set by external clocks which run at 16 times the desired baud rate and which are connected to the separate receiver and transmitter clock inputs.

Operation of the UAR/T in a microprocessor based system is simple. To transmit, data is output by the microprocessor in parallel to the UAR/T, and a device select pulse is sent to the data strobe input, DS, which loads the data into the UAR/T and starts the serial transmission. The UAR/T is double buffered [see Fig. 8.7-4(a)], so a new character can be loaded as soon as the previous one starts transmission. An output from the UAR/T, transmitter buffer empty (TBMT), indicates that the UAR/T's data bits holding register is ready to accept the next character. This output can be tested by the microprocessor to provide handshaking. Or, the transfer from the microprocessor to the UAR/T can be unconditional, in which case, whenever the microprocessor has a character to send, it simply transfers it to the UAR/T. When an unconditional mode of operation is used, the average rate of data transfer to the UAR/T must be less than the word transmission rate of the UAR/T.

In the receiver portion of the UAR/T [Fig. 8.7-4(b)] a clock, running at 16 times the baud rate, operates logic which provides noise immunity. When a 1 to 0 transition is detected at the serial input, the input is tested half a bit time later to verify validity of the transition. The detection of a 1 to 0 transition following an idle period is used by the UAR/T to synchronize itself with the incoming character for the duration of the character. Continued sampling of the data input in the center of a bit time determines the remaining bit values.

An output from the receiver—data available, DAV—goes high, indicating that a character has been completely received. In addition to the parallel data
Figure 8.7-4. Block diagram of the internal structure of a UAR/T: (a) transmitter block diagram; (b) receiver block diagram. (Courtesy of General Instrument Corp.)
outputs which provide the received character, outputs are also available that indicate whether a parity, framing, or overrun error has occurred. When the receiver is operated in a handshaking mode, the microprocessor tests the data available flag, and, if a character is available, inputs it. If a second word is received before the previous word has been read, an overrun error occurs.

The microprocessor can also input error information. The parallel data outputs and error outputs are three-state and can be connected in parallel to the data bus and enabled separately. The received data enable, RDE, and the status word enable, SWE, inputs are controlled by device select pulses for this purpose. A strobe, reset data available, RDAV, resets the UAR/T, indicating that the previous word loaded in the output buffer has been read before the next word is received. This strobe is provided by the microprocessor via the input device select pulse in the handshaking mode or by delaying the character received pulse by external hardware and using it to reset the UAR/T.

If the control bits holding register is connected to the data bus, it can be loaded under program control. The UAR/T is then operated as a programmable device. Further programmability is available if an LSI baud rate generator provides the receiver and transmitter clock frequencies. Baud rate generators provide a receiver and transmitter clock for UAR/Ts, and require only a crystal. Receiver and transmitter frequencies typically are externally selectable from 16 possible values by binary codes written into one of two 4-bit latches on the baud rate generator.

### 8.7.2 Synchronous Serial Character Transfer

Synchronous character transmission eliminates the noninformation-carrying start and stop bits associated with asynchronous transfers and allows faster data transmission. It usually occurs at rates of 3,800 and 9,600 baud. Synchronization between the receiver and transmitter is provided by one or two (bisync) synchronization characters. When transmitting in ASCII, for example, the SYN character is used.

In synchronous transmission, the data received is a continuous stream of bits with no indication of character boundaries. The receiver operates in a hunt mode—making a bit-by-bit comparison of the input stream with the stored values of the desired sync character—until it detects the sync character(s). Once the desired sync character(s) is detected, the receiver treats each subsequent group of \( n \) bits as a character. The transmitter continues to send characters to maintain the synchronization, even if the source of data characters does not have data ready for transmission. In this case, the transmitter sends the sync code or the code for a null character, and thus the time interval between two characters is fixed. The clocks in the transmitter and receiver operate at exactly the same frequency and must be very stable to maintain synchronization for a long period of time. Typically, thousands of blocks of characters can be sent without resynchronizing the receiver.
Single-chip LSI devices are available which provide the required logic for a bidirectional synchronous serial interface in a single package. The functional configurations of the device are programmed by writing information into the control registers. Devices, such as the Intel 8251 Programmable Communication Interface, provide both a synchronous and asynchronous interface in a single package.

### 8.8 DIRECT MICROPROCESSOR SERIAL I/O PINS

Some microprocessors have one or more pins for serial input and output of data. The 8085A, for instance, has a single serial input data, SID, pin and a single serial output data, SOD, pin. Data is input to the microprocessor at the SID pin by the RIM instruction and output from the SOD pin by the SIM instruction. The RIM instruction reads the data state of the SID input into bit 7 of the accumulator. The remaining six bits of the accumulator provide data on the status of the microprocessor's interrupt system (see Chapter 9).

The SIM instruction loads the contents of bit 7 of the accumulator into the SOD latch, if bit 6 of the accumulator is logic 1. If bit 6 is logic 0, the SOD latch is unaffected. When the 8085A is reset, the SOD latch is set to logic 0, and the other six bits of the accumulator set the interrupt masks of the 8085A (see Chapter 9). Appropriate program control of the SOD pin provides serial data directly to an output device. A subroutine for asynchronous serial data transfer using the SOD pin of the 8085A is given in Fig. 8.8-1. The subroutine is called with the 7-bit ASCII character and the parity bit in register B. The parity bit is bit 7 of register B.

```assembly
SRLDO:   MVI C, 100     ; INITIALIZE BIT COUNTER
         MVI A, SOD0    ; SET BIT 7 OF A TO 0 (SOD0 = 01XXXXXX)
         SIM           ; OUTPUT START BIT
LOOP:    CALL DELAY    ; DELAY ONE BIT TIME
         MOV A, B      ; LOAD A WITH DATA AND PARITY
         STC           ; SET CARRY FOR STOP BITS
         RAR           ; PUT BIT TO BE SENT IN CY
         MOV B, A      ; SAVE REMAINING BITS
         RAR           ; PUT BIT TO BE SENT IN A7
         ANI 80H       ; ZERO ALL BUT BIT 7 OF A
         ORI SOD0      ; OR REMAIN BITS REQUIRED FOR SIM
         SIM           ; OUTPUT BIT 7
         DCR C         ; DECREMENT BIT COUNTER
         JNZ LOOP
         RET
```

**Figure 8.8-1.** Subroutine for asynchronous serial data transmission using the SOD pin of the 8085A.
8.9 ELECTRICAL CHARACTERISTICS

As the physical distance between an I/O port and its associated I/O device increases, special consideration must be given to the electrical characteristics of the interconnection in order to minimize the error rate of the data received. As the propagation delay of the interconnection increases relative to the rise and fall time of the signal, interconnection lines cease to respond like simple interconnections and take on the aspects of transmission lines.

Three conditions produce a voltage at the receiving end significantly different from that at the transmitting end, causing received data to be invalid:

1. A noise voltage can be induced into the interconnection via capacitive and inductive coupling, or electrical noise sources, such as motors, in the environment.
2. The transmitting and receiving ends may have different ground connections between which a ground shift voltage exists.
3. The interconnection may act as a transmission line, and reflections of the transmitted voltage may occur on the line.

Typically, standard $T^2L$ gates are restricted to driving lines of a maximum 2 feet in length. In electrically noisy environments, the small noise margin of $T^2L$ voltage levels can result in seriously degraded performance.

8.9.1 Line Drivers and Receivers

Line driver ICs are available which convert $T^2L$ levels into signals for driving transmission lines. Line receiver circuits convert these signals back to $T^2L$ levels. There are three kinds of line driver and receiver interconnections: single-ended, balanced differential, or unbalanced differential (see Fig. 8.9-1).

A single-ended circuit uses one signal line and a common ground return to transmit the signal $V_O$. Although advantageous in that only one signal wire is required per data channel, the performance of the circuit can be degraded by a noise voltage, $V_N$, induced by inductive or capacitive coupling from adjacent signal lines or noise generators such as motors. There may be a voltage between the receiver and transmitter grounds due to a finite resistance between the driver and receiver ground points. The return signal current and possibly other system currents cause a voltage drop, $V_G$, across the resistance between the two ground points. The voltage at the receiver, $V_R$, equals $V_O + V_N + V_G$. The receiver cannot distinguish what portion of the signal is $V_O$ and may assign it an incorrect logic value.

A balanced differential system uses a differential driver and a differential receiver. The driver produces the logic value of the data to be transmitted on one output and its complement on the other. The receiver converts the differential signal into a $T^2L$ level at its output.
Figure 8.9-1. Line driver and receiver interconnections: (a) single-ended; (b) balanced differential; and (c) unbalanced differential.

The driver and receiver are connected by a twisted pair of wires which cancel magnetically induced currents. Electrostatically coupled noise equally affects both lines of the twisted pair and thus appears at both inputs of the differential receiver. This noise voltage, common to both inputs, is referred to as a common mode signal. The ground potential voltage also appears to the receiver as a common mode signal. The voltage at the plus input of the differential receiver is $V_O \pm V_N \pm V_G$, and at the minus input terminal is $\pm V_N \pm V_G$. The differential receiver takes the difference of the signals at its plus and minus inputs, eliminates the noise and ground shift voltage, and leaves $V_O$, which it converts to a T$^2$L logic level, at the output.

In the unbalanced differential method, the minus input of the differential amplifier is connected to the ground return, eliminating one wire. However, performance is diminished because inductive coupling is increased by the use of the common ground return.

8.9.2 Termination

Improperly terminated transmission lines are subject to errors from reflections of the transmitted signals. When data bit duration is long in comparison with
the propagation delay of the line, the effects of the reflection die away in a relatively short period of time. However, when the data bit duration is short in comparison with the propagation time, the line must be properly terminated.

The characteristic impedance of a transmission line, $R_0$—which is a function of its geometry and dimension—typically falls between 50 and 200 ohms. To preclude reflections, a transmission line must be terminated by a resistance equal to its characteristic impedance. Line receivers have a high input impedance and thus require the addition of resistors for proper termination.

### 8.9.3 Standards

Commercial I/O devices, particularly data communications devices, are frequently designed to meet the requirements of one or more formal interface standards. Such standards specify the electrical characteristics and the protocol for transferring data between devices adhering to the standard. A commonly used standard is the RS 232C [6]. The electrical interface is simplex, single-ended, and unterminated. Line length and slew rate limit control reflections. The recommended maximum line length is 50 feet, and the maximum data rate is 20 K baud; logic 1 is $-3$ to $-25$ V, and logic 0 is $+3$ to $+25$ V. IC line drivers and receivers are available which meet the electrical requirements of this standard.

More recent standards allow longer line lengths and higher data rates. Standard RS422 [7] covers the electrical characteristics of a balanced differential interface. This standard allows line lengths of 1,200 meters (4,000 feet) and data rates of 10 M baud. Standard RS423 [8] pertains to unbalanced differential circuits and allows a line length of 1,200 meters (4,000 feet) and data rates of 100 K baud. IC drivers and receivers are also available which meet these standards.

ANSI/IEEE Std. 488 [9] is a standard for interconnecting electronic instrumentation. As many as 15 instruments can be connected with a total cable length of 20 meters. When fewer instruments are used, the maximum cable length is 2 meters of cable per instrument. An 8-bit data bus and a number of control signals provide handshaking, and data transfer occurs at 250 K bytes per second. The protocol of the standard allows a device connected to the bus to function as a listener, which only receives data; as a talker, which only transmits data; or as a controller, which manages the operation of the bus. Many instruments are designed with this interface; thus, automatic test systems are easily configured. The instrument is connected to the bus, and a microprocessor or even a programmable calculator can be the controller.

### REFERENCES


**BIBLIOGRAPHY**


**PROBLEMS**

8-1. List the eight unique hexadecimal port addresses possible using isolated I/O and linear selection. Assume that the address line associated with each port is logic 1 when the port is selected.
8-2. Repeat problem 8-1 assuming that the address line associated with each port is logic 0 when the port is selected.

8-3. List all the instructions which will create each of the following device select pulses with the hardware of Fig. 8.2-4. Specify the instructions' addresses in hexadecimal.
   a. $\text{IDSP00H}$
   b. $\text{IDSP05H}$
   c. $\text{0DSP03H}$
   d. $\text{0DSP06H}$

8-4. Using three 8205s, NAND gates, and inverters, draw the logic diagram of a circuit that generates 24 output device select pulses, $\text{0DSP00H}$ to $\text{0DSP17H}$.

8-5. Design an 8-bit extended address register, as in Fig. 8.2-6, using an 8212 and NAND gates. Make the address of the extended address register 00H. Show all the connections required to an 8085A system bus.

8-6. Using 8205s and common gates, design the logic required to generate 24 input device select pulses, $\text{IDSP08H}$ to $\text{IDSP1FH}$.

8-7. Design the logic necessary to generate eight device select pulses for input and eight device select pulses for output using memory mapped I/O. Make the port addresses FFF0H to FFF7H. Use 8205s, NAND gates, and inverters.

8-8. List the advantages and disadvantages of isolated and memory mapped I/O.

8-9. Draw the logic diagram of the interconnection as output ports of (a) a 74LS373 octal latch and (b) a 74LS374 octal flip-flop (see Fig. 8.4-1) to an 8085A system bus. Assume the existence of address decoding logic to generate the necessary output device select pulse. Show the connection of the output device select pulse to the output port.

8-10. Repeat problem 8-9 using the 74LS373 and 74LS374 to create input ports.

8-11. Draw the logic diagram to interface a 9334 addressable latch, Fig. 8.4-2, to an 8085A system data bus to provide eight single bit output ports.

8-12. Draw a logic diagram to show how a 4-bit extended address register is placed in an 8085A system and utilized. Compute the maximum number of output ports used for data transfer with this technique. Give the instruction sequence used to output data. Also, give the preferred order of addressing these ports to minimize software.

8-13. An input device is interfaced to an 8085A microprocessor. The device has one input, RUN, which, when high, operates the device. The device then outputs bytes of data asynchronously on its eight data lines. It has one data valid, DAV, status output, which indicates valid data each time it makes a 1 to 0 transition. The RUN, DAV, and eight data outputs are the only connections to the device. These inputs and outputs are $T^2L$. Design the hardware and draw the logic diagram for this interface with the following port assignments:

<table>
<thead>
<tr>
<th>Port</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUN</td>
<td>bit 7</td>
</tr>
<tr>
<td>DAV</td>
<td>bit 7</td>
</tr>
<tr>
<td>Data (8 bits)</td>
<td>input port 0</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Write a subroutine that starts the device, inputs 64 bytes of data, stores them in a memory buffer, BUFF, and then turns the device off.
8-14. There are certain types of registers which are common to all programmable devices, including the 8155, 8255, and UAR/T. Describe, in general, these registers and their functions. For the 8155 describe the function and/or information provided by these registers in detail.

8-15. Design the interface between seven input devices and an 8085A microprocessor system which transfers data using programmed I/O. The devices are numbered 1 through 7; device number 1 has the highest priority. Use a 74148 priority encoder to speed up priority arbitration. The 74148 has eight active low inputs and three outputs. Input 7 of this encoder has the highest priority. The output of the priority encoder is the complement of the binary equivalent of the number of its highest priority active input.
   a. Draw a logic diagram of the interconnection of the data available flip-flops to the priority encoder and a single input port which provides the microprocessor with the status of all the input devices. Also show the logic necessary to clear the data available flip-flops.
   b. Write a program which checks the status and jumps to the appropriate service routine via a jump table.

8-16. Modify subroutine PIN, shown in Fig. 8.5-2, for conditional data transfer to provide a controlled timeout feature. If the input device does not respond in 100 mS with each byte of data, the subroutine, PIN, should be returned from, with the carry cleared. If all bytes of data are transferred successfully, the subroutine returns with the carry set.

8-17. An 8155 is used with port A as output, port B as input, and with port C used for handshaking for ports A and B. The 8155 is enabled when A15 to A8 = 00110XXX. Write the instruction sequence to program the 8155 for this mode of operation.

8-18. For the 8155 of problem 8-17, write a subroutine which outputs 40 bytes of data from a memory buffer labeled LINE, to port A using handshaking. The subroutine should check the 8155's status word to determine when the output device has accepted the data.

8-19. Determine the control words necessary to put an 8255 in mode 0 with the following configuration:
   a. port A, input; ports B and C output
   b. ports A and C lower, output; ports B and C upper, inputs
   c. ports A and C upper, input; ports B and C lower, output
   d. ports A and B, input; port C, output