EE 3232 DIGITAL SYSTEMS III
6.2 PARALLEL HANDSHAKING WITH A PRINTER

March 2000 CPD

OBJECTIVE

To learn how the Centronics parallel IO interface is implemented using a TL16C452 peripheral, to verify the Centronics handshaking protocol and to develop and debug software that writes a string of characters to a parallel printer using polled handshaking I/O.

REFERENCES

1. "Embedded Microprocessor System Design", K.L. Short, Chapter 12, pp. 397 - 400.
2. "Microprocessors and Interfacing - Programming and Hardware", D.V. Hall, 2nd Edition, pp. 252-258.
3. "The 80C186XL/80C188XL Microprocessor Users Manual", available in the lab, also at http://developer.intel.com/design/intarch/manuals/
4. Section 8.2, "The TL16C452 Communications Peripheral", also at http://www.ti.com/sc/ (Then search for TL16C452)
5. Section 7.1 - 7.2, "Sheets 1/11 and 2/11 of the SBC188 Schematics".
6 "The SBC188 Users Manual", available in the lab.
7. "Paradigm Debug/RT - 186 Users Guide", available in the lab.
8. "Paradigm Locate Reference Manual", available in the lab.

EQUIPMENT

1. SBC188 board.
2. Host PC.
3. IO interface board.
4. 26-pin ribbon cable.
5. 10-pin ribbon cable.
6. Parallel printer and printer cable.

BACKGROUND: The Centronics Parallel Interface

The Centronics interface is a standard interface for writing characters to a parallel printer. The SBC188 uses the TL16C452 peripheral device as an interface between the printer connector and the SBC188 system bus. Consult sheet 1/11 and 2/11 of SBC188 schematics in Sections 7.1 - 7.2 and identify the following:

- The 80C188XL microprocessor.
- The 74HCT573 latch for demultiplexing the address/data pins.
- The PA7104 programmable gate array that implements address and control signal decoding for various chip selects.
- The TL16C452 peripheral.

- The parallel printer connector, J201.

The TL16C452 is a multi-function peripheral that interfaces with a Centronics parallel port and two serial communication channels. In this lab you will only use the built-in parallel port functions of the TL16C452. By examining sheet 2/11 of the SBC188 schematics identify the interface of the following TL16C452 pins with the SBC188 system bus: DB0-DB7, A0-A3, /CS2, /IOR, /IOW and INT2. You should also identify the interface of the following pins with the printer connector: /STB PD0-PD7, /ACK, BUSY, PE, SELECT, /ERROR, /AUTOFD, /INIT and /SELIN. Consult pages 4-5 of the TL16C452 Data Sheets in the APPENDIX for a description of the function of each of these pins.

By consulting the IO Port Map section in your Lab Manual, you should verify that the Centronics ports in the TL16C452 are IO-mapped in the SBC188 system as follows.

Port 0x120 ***Write/read printer data***.
Port 0x122 ***Read printer status***.
Port 0x124 ***Write/read printer control signals***.

Read page 22 of the TL16C452 Data Sheets in Section 8.2 to verify the bit assignments of the parallel printer port as follows.

PRINTER PORT DETAILS

Port 0x120: PDR (Printer Data Port - Write Only)

Writing to this port transfers the contents of the data bus to the printer data pins. The printers used in this experiment accept 8-bit data with ASCII representation.

Port 0x122: PSR (Printer Status Register - Read Only)

Reading this port retrieves printer and interrupt status as follows,

PSR0: Always read as '1'.
PSR1: Always read as a '1'.
PSR2: /INTP (An interrupt caused by the falling edge of /ACK is pending. PSR2 is deactivated to '1' after PSR is read).
PSR3: /ERROR
PSR4: SELECT
PSR5: PE
PSR6: /ACK
PSR7: /BUSY

Port 0x124: PCR (Printer Control Port - Read and Write)

PCR0: STB
PCR1: AUTOFD
PCR2: /INIT

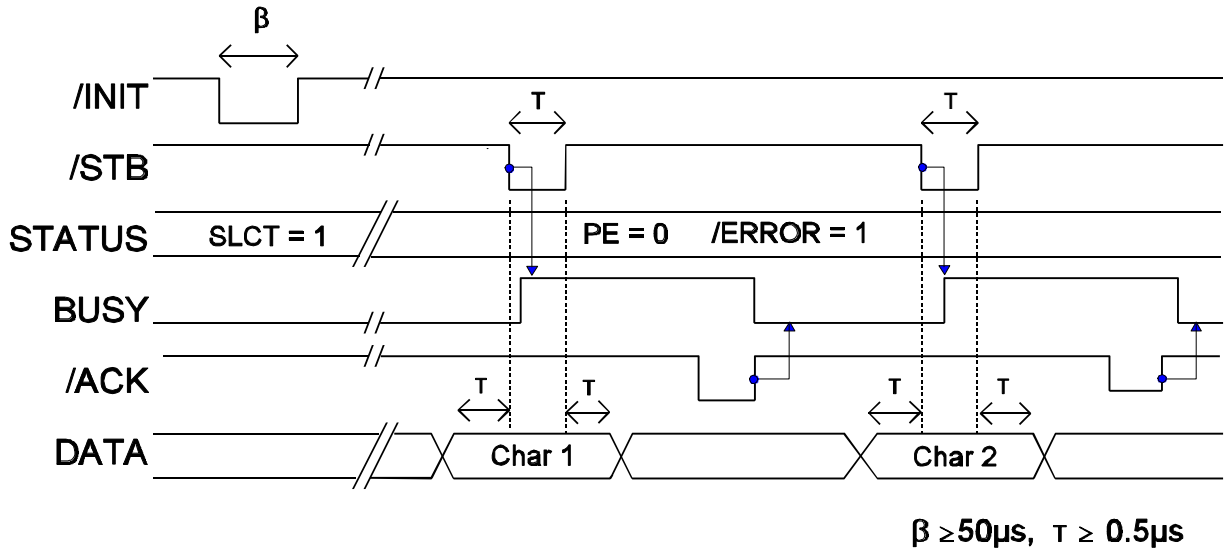
PCR3: SLCTIN
PCR4: INTP_ENABLE ('1' = INTP interrupt output is enabled, '0' = disabled).
PCR5: '1'
PCR6: '1'
PCR7: '1'

Before the printer can be used it must first be initialized. Upon initialization, the printer supplies the status bits SELECT ('1' = printer selected), PE ('0' = not out of paper) and /ERROR ('1' = no printer error). Once initialized any number of ASCII characters may be written to the printer using the Centronics handshake. The initialization sequence and a typical Centronics handshake appears in the table below.

The initialization sequence for the printer.	**A typical Centronics polled handshake.**

Deactivate /STB high.
Activate /INIT for ≥ 50 µsecs.
If ((/ERROR != 1) | PE != 0) | (SLCT != 1)) {
Signal a printer error;
}
	While(/BUSY == 0) { }
	Write character to data port.
	Wait for ≥ 0.5 µs.
	Activate /STB ≥ 0.5 µs.
	Deactivate /STB ≥ 0.5 µs.

A timing diagram associated with the Centronics interface follows,



PREPARATION: Software Development

1. Write a procedure **unsigned PRN_INIT(void)** that initializes the parallel printer as follows. INIT_PRN() activates /INIT (lo) for a minimum of 50 microseconds and then deactivates /INIT (hi). The other bits should be set as follows: /STB inactive (hi), /AUTOFD inactive (hi),

/SLCTIN active (lo). The procedure must then check to determine if there are printer errors, (i.e., PE SELECT /ERROR ≠ 011B). The procedure should return an error code as follows:

0 - no errors
1 - if /ERROR = 0
2 - if SELECT = 0
3 - if PE = 1.

2. Write a procedure, **void PRN_BUSY(void)** that polls the /BUSY status signal until the printer is not busy, i.e., waits until /BUSY != 0.

3. Create a procedure **void PRN_WRITE_CHAR(char A)** that writes the character (passed as A) to the printer data pins, waits for ≥ 0.5 microseconds, activates /STB (lo), waits for ≥ 0.5 microseconds, deactivates /STB (hi) and waits for ≥ 0.5 microseconds before returning. NOTE: /INIT and /AUTOFD must remain inactive (hi) and /SLCTIN must remain active (lo) during the write operation.

4. Using PRN_BUSY() and PRN_WRITE_CHAR(), write a procedure **void PRN_PUT_CHAR(char A)** that waits until the printer is ready, i.e., waits until /BUSY != 0, and then prints the ASCII character passed as A by calling PRN_WRITE_CHAR().

5. Using PRN_PUT_CHAR() write a procedure **void PRN_PUT_STRING(char *PTR_STRING)** that prints a string of characters. The starting address of the string is passed as PTR_STRING, and the string terminates with the sentinel character, NULL = 0x00 = '/0'.

6. * In this advanced section you will use the procedure PRN_PUT_STRING() to display the value of a byte array in HEX. First you will develop the procedure BIN_TO_HEX() that converts a byte variable, X, into its 2-digit HEX representation in ASCII.

void BIN_TO_HEX(char X, char *PTR_ASCII) : Converts the 8-bit binary number passed as X into an 2-byte ***hex*** representation in ASCII. The two ASCII bytes are to be returned in two consecutive memory locations pointed to by PTR_ASCII, (hi-order byte first then lo-order byte) as illustrated in the Table below.

X	ASCII Values		Printed Value
01010111 B	'5' = 35H '7' = 37H	lo-order hi-order	57
11001001 B	'C' = 43H '9' = 09H	lo-order hi-order	C9
10101111 B	'A' = 41H 'F' = 46H	lo-order hi-order	AF

Using BIN_TO_HEX() write a procedure, **void PRINTF(char *PTR_X)**, that prints in HEX

the elements of the byte array whose address is passed as PTR_X. Assume the printed characters begin on a new line in the left most column and that each character is separated by an ASCII space, SP = 20H. To start printing on a new line the ASCII characters, LF = 0AH, and CR = 0DH must first be written to the printer.

7. * In this advanced section you will use the procedure PRN_PUT_STRING() to display the value of an unsigned byte array in DECIMAL. First you will develop the procedure BIN_TO_DEC() that converts a byte variable, X, into its 3-digit DECIMAL representation in ASCII.

void BIN_TO_DEC(char X, char *PTR_ASCII) : Converts the unsigned 8-bit binary number passed as X into a 3-byte *decimal* representation in ASCII. The three ASCII bytes are to be returned in two consecutive memory locations pointed to by PTR_ASCII, (hi-order byte first and lo-order byte last) as illustrated in the Table below.

X	Decimal Values in ASCII	Printed Value
01010111 B	' ' = SP '8' = 38H '7' = 37H	87
11001001 B	'2' = 42H '1' = 41H '1' = 41H	211
10001111 B	'1' = 41H '4' = 44H '3' = 43H	143

Using BIN_TO_DEC() modify the procedure, PRINTF() as follows, **void PRINTF(char *PTR_X, char FORMAT)**, prints the elements of the byte array whose address is passed as PTR_X. The array is printed in HEX if FORMAT = 'H' or in decimal if FORMAT = 'D'.

EXPERIMENT

1. **Hardware Debugging**

Connect the 26-pin cable between header J-201 on the SBC188 and the 26-pin Centronics header on the IO Board. Hook up the printer to the Centronics port on the IO Interface Board and ensure the printer is on-line. Connect a 10-pin cable between 8-LED's and the printer status bits so these can be displayed. The status bits are displayed as follows.

SLCT	PE	BUSY	/ACK	/INIT	/ERROR	/AUTOFD	/STB

(MSB) (LSB)

Use the debug monitor I/O command to read the printer status, initialize the printer and then write a single ASCII character to the printer data port using the same sequence as the procedures PRN_INIT() and PRN_PUT_CHAR(). Proceed to debug any hardware errors.

2. **Software Debugging**

   Load the program containing INIT_PRN() and PUT_CHAR_PRN(). Trace through the INIT_PRN() being careful to verify all IO operations. Also trace through the PRN_PUT_CHAR() being careful to verify its operation.

3. Debug and execute the program developed in the preparation that prints a string of characters.

4. * Using procedure PRINTF() debug and execute a program that prints the contents of an array stored in memory.

   *** ADVANCED SECTIONS**

SUMMARY

1. List the port names and the addresses of the TL16C452 peripheral that you have used in this experiment.

2. List the limitations imposed by the Centronics handshake and the use of polled handshaking.

3. Explain how would you modify your program such that binary values are printed instead of hex or decimal values.

EXTENSIONS : An Interrupt Driven Printer Interface

The following describes how you may modify the polled handshaking design so that the printer interface is interrupt driven. What are the advantages and disadvantages of an interrupt driven printer interface?

In addition to polled handshaking, the TL16C452 also allows the printer to interrupt the 80C188XL microprocessor at the INT3 interrupt pin as shown in Figure 6.5.1. A printer interrupt is activated whenever the printer activates the acknowledge, /ACK, i.e., whenever the printer is not busy. The interrupt type associated with the INT3 hardware interrupt (of the 80C188XL) is 15 (decimal). Therefore the interrupt vector table for a type 15 interrupt (i.e., the printer interrupt) must be initialized before printer interrupts are turned on.

Printer interrupt generation by the TL16C452 is enabled when the INTP bit of the printer control port, PCR, is set by writing '1' in bit-4. The other bits of the printer control port, i.e., /SLCTIN, /INIT, /AUTOFD and /STB must also be appropriately initialized. NOTE: Interrupts should not be turned on until after /BUSY = '1'.

The 80C188XL interrupt controller also needs to be initialized for accepting hardware interrupts at INT3 by writing 0x0000 to the internal 16-bit port, I3CON (address 0xFF3E).

Within the printer interrupt handler, one character is written to the printer using PRN_WRITE_CHAR(). If the last character has been printed then interrupts are turned off by masking interrupts at INT3 (set bit-3 of the 16-bit internal 80C188XL port I3CON, address 0xFF3E) and resetting bit-4, INTP = '0' in the TL16C452 printer control port. Next an EOI (end of interrupt command) is sent to the 80C188XL interrupt controller (by writing 15 decimal to the 16-bit port 0x0FF22).

If a string of characters is to be printed using interrupt driven handshaking, then a pointer to the string will need to be initialized. Each time a printer interrupt is invoked the pointer may be incremented to point to the next character to be printed within the interrupt handler.