PARALLEL IO and HANDSHAKING

EE3232 DIGITAL SYSTEMS III
CLASS NOTES CHAPTER 5

Department of Electrical Engineering
University of New Brunswick

 January 5, 2000

SUMMARY

OBJECTIVES

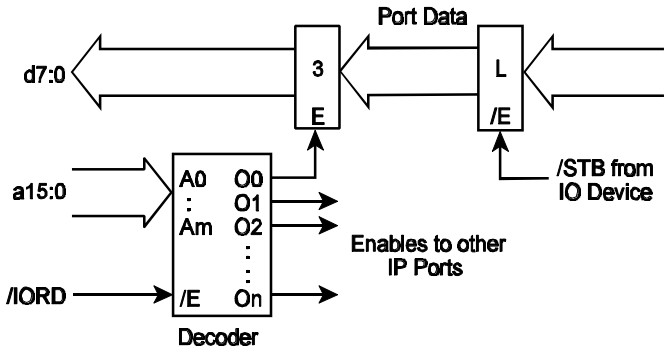
- To show how simple IO ports may be used to build a parallel peripheral interface, PPI.
- To explain the need for handshaking IO and to design handshaking IO interfaces - hardware and software.
- To investigate the operation of the 82C55A PPI.
- To investigate the operation of the TL16C452 Integrated Peripheral and the Centronics parallel interface.

INTRODUCTION

- IO transfers may be classified as:
  - ***Unconditional*** : A transfer proceeds whenever data is available, (e.g., switches and LED's).
  - ***Conditional*** : A transfer proceeds only when the IO device is ready - synchronization is needed (e.g., UART's, ADC's, MODEM's, printers, ...).
- ***Handshaking*** : is a conditional IO transfer which uses separate control signals for achieving synchronization.

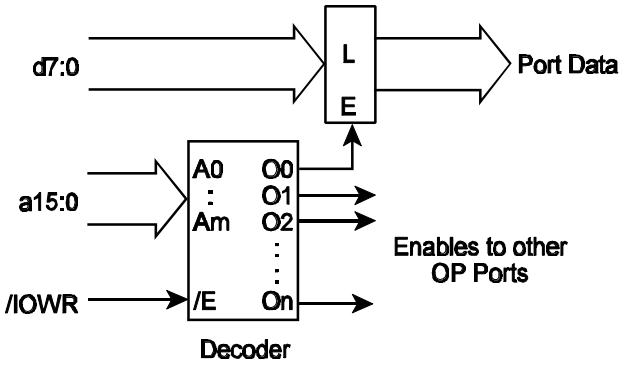
INTERFACING SIMPLE IP PORTS

- Port data is gated onto the DATA bus only when the selected input port is being read.
- Hardware consists of : an address decoder, three state buffer and (optional) latch to hold the data while /IORD = '0'.



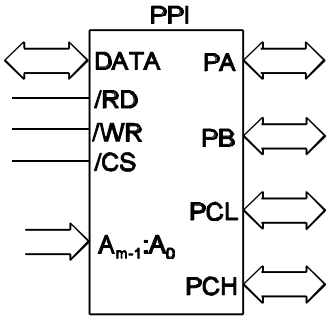
INTERFACING SIMPLE OP PORTS

- Data remains at the OP Port until a new value is written.
- A simple OP port consists of an address decoder and a latch to ***capture and hold*** the data.

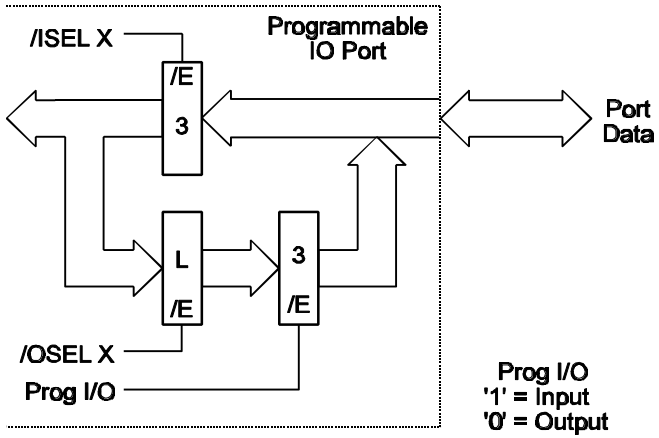


ARCHITECTURE OF A PPI

- Common IO pins for data.
- Separate /RD and /WR control pins.
- m-address pins: selects 1 of 2^m ports or registers.
- One or more chip selects, /CS's.
- One or more 8-bit parallel IO ports.



- The IO ports may be programmed as input or output.

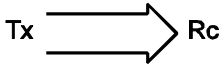


- Programmable devices usually contain,
  - ***A Control Register*** : to program mode of operation.
  - ***A Status Register*** : to read operating status.
- In a PPI, the Control Register is first written (after a reset) to configure selected ports as IP or OP.
- The PPI interfaces to a system bus like memory :
  - Connect the LSB's of the ADDR bus to the address pins of the PPI,
  - Decode the unused address lines to generate the device /CS for a specified base or starting address.
  - Connect the /IORD, /IOWR and any other required system bus signals.

HANDSHAKING

- Information is transmitted to a receiver in byte-serial format, i.e., bytes are sent one after the other.
- The issues,
  1. How does the receiver, Rc, know when valid data is available ?
  2. How does the transmitter, Tx, know when new data can be sent ?

Polled Handshaking Analogy #1

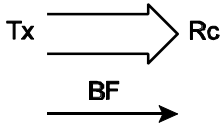


- **Tx** : Write first/next byte ;
- **Rc** : Read first/next byte ;

- **Problem** : the Rc does not know ***when*** valid data is available to be read.

- **Solution** : the Rc waits until the Tx activates a separate handshake signal, buffer full, BF.

Polled Handshaking Analogy #2



- **Tx** : Write next byte to Rc ;
  Activate BF ;
- **Rc** : do { Read BF ; }
  while (BF is NOT active) ;
  Read data supplied by Tx ;

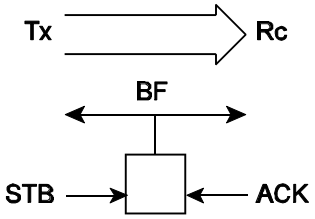
- **Problem** : the Tx may send new information before the Rc has processed the old information!

- **Solution** : the Tx waits until the Rc deactivates the handshake signal, buffer full, BF.

Polled Handshaking Analogy #3

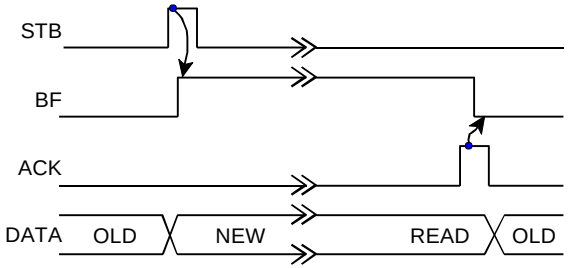
- **Tx** : do { Read BF ; } while (BF is active) ;
  Write next byte to Rc ;
  Activate BF ;

- **Rc** : do { Read BF ; } while (BF is NOT active) ;
  Read data supplied by Tx ;
  Deactivate BF ;



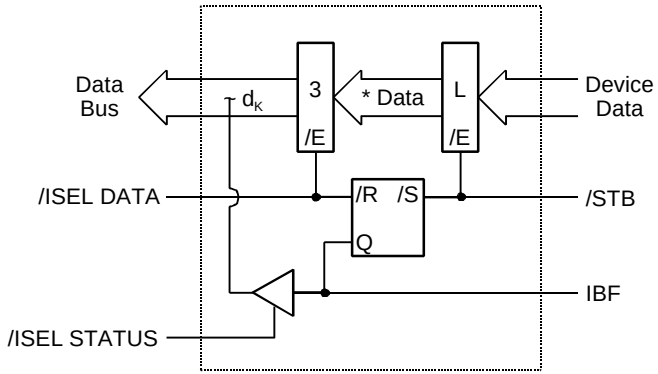
HANDSHAKING TIMING

- Draw a timing diagram that shows STB, ACK, BF and DATA when a byte is transferred from Tx to Rc using handshaking IO.



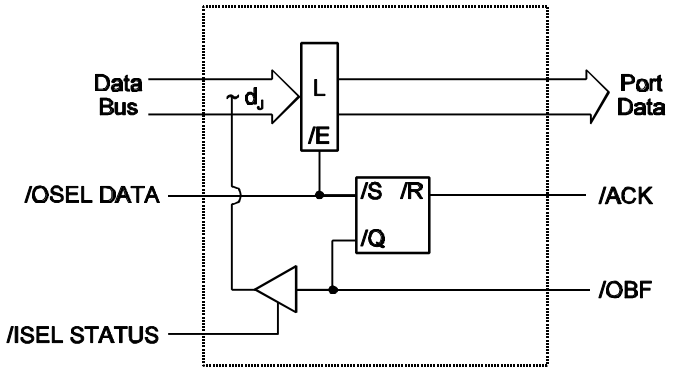
HANDSHAKING INPUT H/W AND S/W INTERFACE

```
do { X = inportb(STATUS) ; }
while ( (X & MASK) == 0 ) ;
INFO = inportb(DATA) ;    // This deactivates IBF
                          // automatically.
```



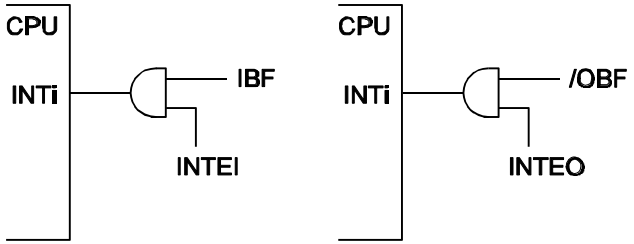
HANDSHAKING OUTPUT H/W AND S/W INTERFACE

```
do { X = inportb(STATUS) ; }
while ( (X & MASK) == 0 ) ;
outportb(DATA, INFO) ;     // This activates /OBF
                           // automatically.
```



INTERRUPT DRIVEN HANDSHAKING

- Polling the status bits, IBF or /OBF, to synchronize IO operations is inefficient.
- Alternatively : connect IBF or /OBF to a CPU interrupt !
- When the IO device is ready, an interrupt is generated !
- This transfers control to an interrupt handler.
- A ***single*** IO transfer now takes place inside the interrupt handler ***without*** the need for polling!

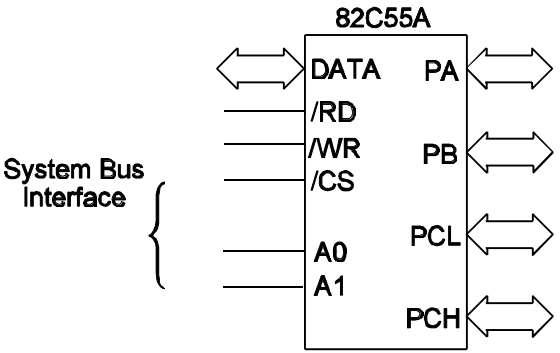


THE 82C55A PARALLEL PERIPHERAL INTERFACE OR PPI

- 24-bits of programmable IO :
  - PA (8-bits)
  - PB (8-bits)
  - PCL (4-bits)
  - PCH (4-bits)
- Operating Modes :
  - ***MODE 0*** : simple IP or simple OP,
  - ***MODE 1*** : handshaking IO in one direction with PA and/or PB
  - ***MODE 2*** : handshaking IO in two directions with PA.

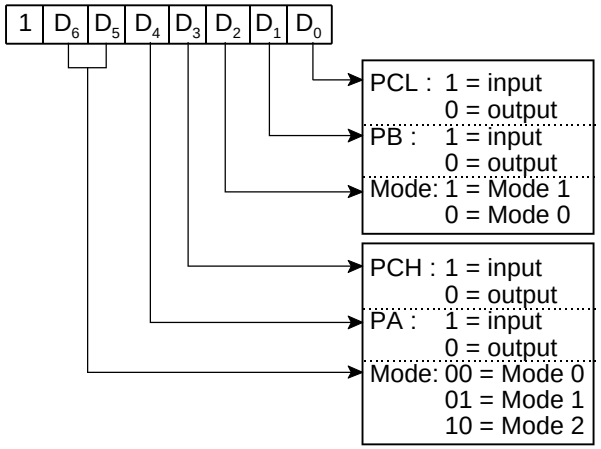
THE 82C55A READ/WRITE OPERATION TABLE

/CS	/RD	/WR	A1	A0	OPERATION
1	x	x	x	x	(Data) ← Hi-Z
0	0	1	0	0	(Data) ← (PA)
0	0	1	0	1	(Data) ← (PB)
0	0	1	1	0	(Data) ← (PC)
0	0	1	1	1	Invalid
0	1	0	0	0	(Data) → (PA)
0	1	0	0	1	(Data) → (PB)
0	1	0	1	0	(Data) → (PC)
0	1	0	1	1	(Data) → (Control)



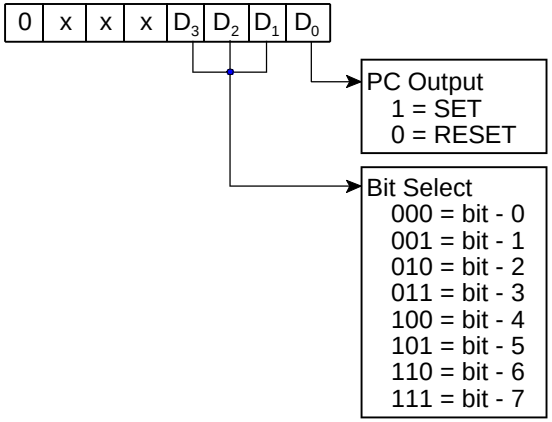
THE 82C55A CONTROL WORD FORMATS

- The 8255A Mode Set Control Word Format



THE 82C55A CONTROL WORD FORMATS

- The Bit Set-Reset Control Word Format for PC Only.



82C55A INTERFACE EXAMPLES - MODE 0

- ***Hardware Interface*** : an 82C55A is to be interfaced to a system bus such that the base address is 0x200.
- ***Software Interface*** : the ports are configured with PA-output, PB-input, PCL-output and PCH-input.

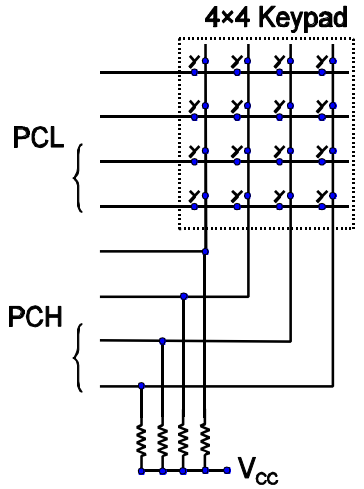
```
# define  PortA    0x200    // 82C55A port addresses.
# define  PortB    0x201
# define  PortC    0x202
# define  CTRL     0x203
# define  MODE     0x8A     // 82C55A Mode word = 10001010B.

void main(void)
{
      char X;
      outportb(CTRL, MODE);   // Initialize 82C55A PPI.
      outportb(PortA, 0xFF);  // Write 11111111B to PA.
      X = inportb(PortB);     // Read PB.
      outportb(PortC, X);     // Write 4-LSB's of X to PCL.
      X = inportb(PortC);     // Read PCH into 4-MSB's of X.
      X = X & 0xF0;
}
```

KEYPAD INTERFACE EXAMPLE

The 82C55A is interfaced to a keypad matrix as shown. Give hi-level code for a keyboard scan:

1. Wait for any key to be pressed ;
2. Identify the row and column of the pressed key ;
3. Ensure a valid key press ;
4. Wait for the key to be released ;
5. Encode the pressed key ;



KEYPAD INTERFACE EXAMPLE (CONT'D)

```
char KEY_PAD(void) {

ALL_ROWS ← 0 ;
do {
      do { Read columns ;}              // Wait for a key to
      while (No key is pressed) ;       //  be pressed.
      // Identify the row and column of the pressed key.
      do {First/next row ← '0', all other rows ← '1' ;
            scan_code = Value read at rows and columns ;
            }
      while (A pressed key is not detected) ;
      Delay for 10's of milliseconds;  // S/W debouncing
      }
// Keep scanning if glitching or bouncing occurs.
while (scan_code != Value read at rows and columns) ;

// Wait for the key to be released.
ALL_ROWS ← 0 ;
do { Read columns ; }                     // Wait for (all) key(s) to
while (Any key is pressed) ;              // be released.

KEY = ENCODE(scan_code) ;         // Encode pressed key in ASCII and
return KEY ;                      // pass back as the parameter.
}
```

BIT SET-RESET OPERATION OF PC OF THE 82C55A

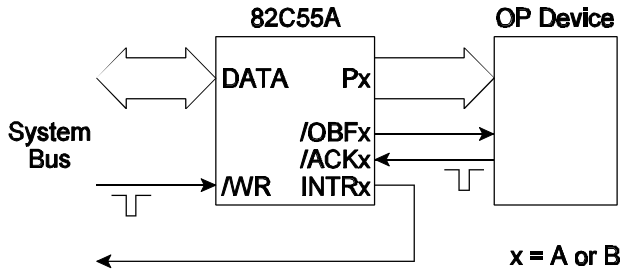
- Sets or resets individual bits of PC.
- Useful for control of on-off devices such as motors, pumps, alarms and solenoid valves ...

```
outportb(CTRL, 0x80);      // Initialize PC as output.

outportb(CTRL, 0x00);      // Reset PC0 to 0.
outportb(CTRL, 0x04);      // Reset PC2 to 0.
outportb(CTRL, 0x0F);      // Set PC7 to 1.
```

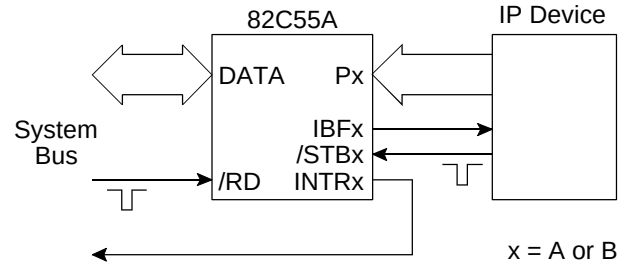
HANDSHAKING (MODE-1) OPERATION OF THE 82C55A

- PA and/or PB may be programmed for handshaking, i.e., MODE-1 operation.
- In MODE-1 certain bits of port PC automatically default as the handshaking bits, /OBFx, /STBx, IBFx and INTRx.



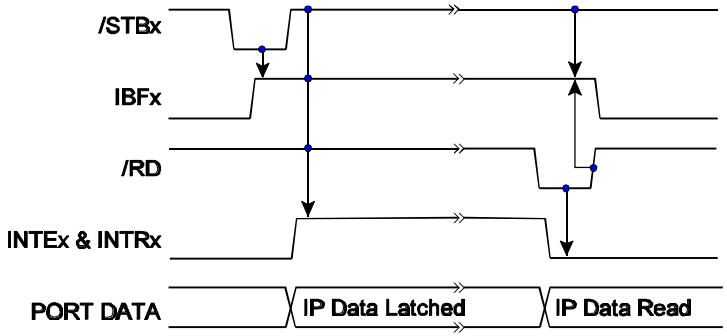
- /WR from CPU to Px activates /OBFx.
- /ACKx pulse from OP device (to 82C55A) deactivates /OBFx.

MODE-1 OPERATION OF THE 82C55A (CONT'D)

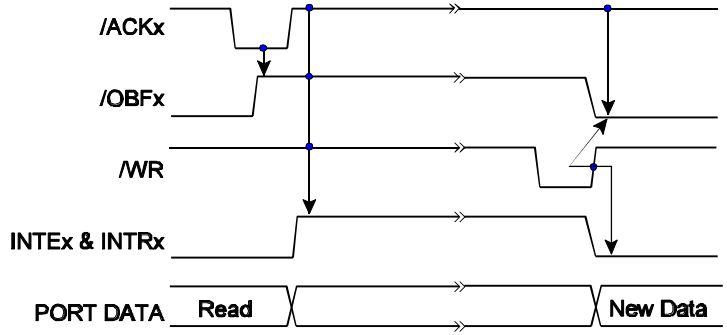


- /STBx pulse from IP device (to 82C55A) activates IBFx.
- /RD of Px (by CPU) deactivates IBFx.

MODE 1: 82C55A HANDSHAKING INPUT PORT TIMING



MODE 1: 82C55A HANDSHAKING OUTPUT PORT TIMING



MODE-1 PC BIT ASSIGNMENTS FOR THE 82C55A

- The 82C55A PC Pin Configurations for Mode 1 - Handshaking I/O.

	Mode 1 - IP	Mode 1 - OP
PC_0	INTRB	INTRB
PC_1	IBFB	/OBFB
PC_2	/STBB	/ACKB
PC_3	INTRA	INTRA
PC_4	/STBA	IO
PC_5	IBFA	IO
PC_6	IO	/ACKA
PC_7	IO	/OBFA

MODE-1 STATUS READ OF THE 82C55A (CONT'D)

- When programmed in MODE 1 (or 2) read PC (A1 = 1, A0 = 0) for the handshaking status.
- The 82C55A handshaking status format for MODE 1 (or 2) is,

	PA - Mode 1					PB - Mode 1		
IP	IO	IO	IBFA	INTEA	INTRA	INTEB	IBFB	INTRB
OP	OBFA	INTEA	IO	IO	INTRA	INTEB	OBFB	INTRB

- In MODE 1 (or 2), interrupt generation by the 82C55A is enabled by writing '1' to the INTEx bit in PC.

AN 82C55A INTERFACE EXAMPLE - MODE 1

- ***Hardware Interface*** : an 82C55A with base address = 0x0600 interfaced to a handshaking OP device using /OBFB and /ACKB.

- ***Software Interface*** : A program to initialize PB as a mode-1 output port and writes the elements of an array to PB using polled handshaking.

```
# define MODE   0x84      // Mode word = 1xxxx10xB.
# define MASK   0x02      // /OBFB mask = 00000010B.

void main(void)  {
char X = { 0, 2, 4, 6, 8, 10 } ;
outportb(CTRL, MODE) ;  // Initialize PB as a MODE-1
i = 0 ;                 // (handshaking) OP port.
while (i < M) {
      while ((MASK & inportb(PC)) != 0) { }     // Poll OBF
      outportb(PortB, X[i]) ;
      i = i + 1;
      }
}
```

THE CENTRONICS PARALLEL INTERFACE

- The Centronics interface is a standard interface for parallel printers. 36-pins are defined.
- The handshake synchronizes each character to be printed.

PD7:0	8-bit printer input data.
/INIT	Printer input, forces the printer to perform an internal initialization.
PE	Printer output. Out of paper.
SLCT	Printer output. Printer selected.
/ERR	Printer output. Internal printer error
BUSY	Printer handshaking output - level.
/ACK	Printer handshaking output - edge.
/STB	Printer input. Data strobe.
/AFD	Printer input. Auto-feed paper.
/SLCTIN	Printer input. Printer selected.

CENTRONICS INITIALIZATION AND HANDSHAKE

- The Centronics interface ***initialization*** sequence:

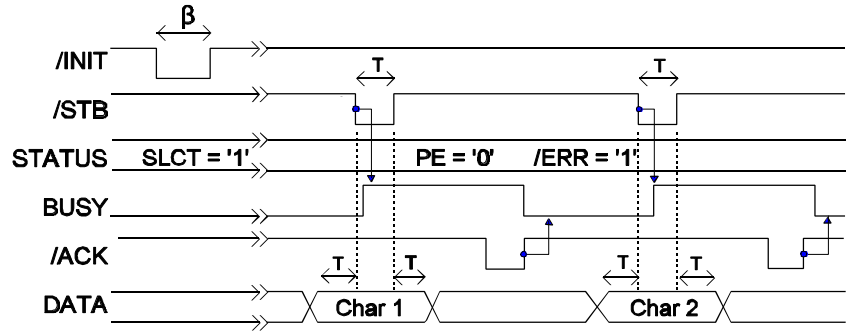
1. Deactivate /STB, activate /SLCTIN, deactivate /AFD.
2. Activate /INIT for at least 50 microseconds.
3. If (/ERROR 1 or PE 0 or SLCT 1) {
   Signal a printer error;
   }
   Else { Proceed ; }

- A typical Centronics polled ***handshake***,

```
While (BUSY != 0) { }        // Poll until BUSY = 0.
Write the next character to the PD pins ;
Wait for    0.5 microseconds ;
Activate /STB for    0.5 microseconds ;
Deactivate /STB for    0.5 microseconds ;
```

CENTRONICS INTERFACE TIMING

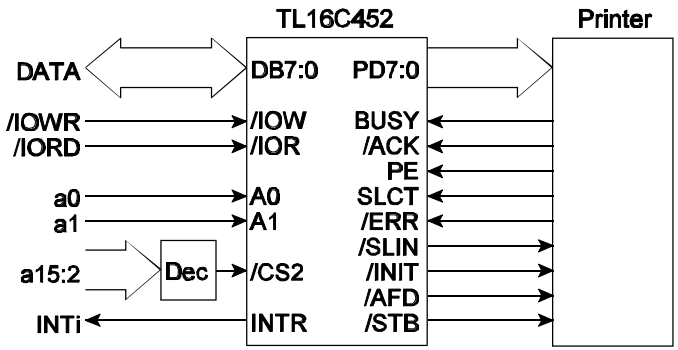
- The timing associated with the Centronics interface is shown below.



β ≥50µs, τ ≥ 0.5µs

IMPLEMENTING THE CENTRONICS INTERFACE USING A TL16C452 INTEGRATED PERIPHERAL

- Consult the TL16C452 data sheets in your Lab manual.



THE READ/WRITE OPERATION TABLE FOR THE CENTRONICS INTERFACE OF THE TL16C452

/CS2	/IOR	/IOW	A1	A0	Operation
1	x	x	x	x	DATA ← Hi-Z
0	0	1	0	0	DATA ← PD
0	0	1	0	1	DATA ← STATUS
0	0	1	1	0	DATA ← CONTROL
0	0	1	1	1	Invalid
0	1	0	0	0	PD ← DATA
0	1	0	0	1	Invalid
0	1	0	1	0	CONTROL ← DATA
0	1	0	1	1	Invalid

- ***Control & Status Register*** : bit formats.

Status	/BUSY	/ACK	PE	SLCT	/ERR	/INT	1	1
Control	1	1	1	IRQEN	SLIN	/INIT	AFD	STB