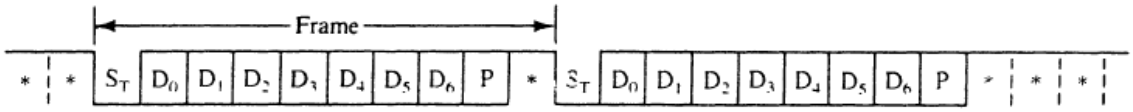
11. I16550 serial communications interface adapter.
12. I8255A programmable parallel interface.

16550 PROGRAMMABLE COMMUNICATIONS INTERFACE

The National Semiconductor Corporation's PC16550D is a programmable communications interface designed to connect to virtually any type of serial interface. The 16550 is a universal asynchronous receiver/transmitter (UART) that is fully compatible with the Intel microprocessors. The 16550 is capable of operating at 0–1.5 M Baud. Baud rate is the number of bits transferred per second, including start, stop, data, and parity. The 16550 also includes a programmable Baud rate generator and separate FIFOs for input and output data to ease the load on the microprocessor. Each FIFO contains 16 bytes of storage. This is the most common communications interface found in modern microprocessor-based equipment, including the personal computer and many modems.
Asynchronous serial data are transmitted and received without a clock or timing signal. Figure 10–42 illustrates two frames of asynchronous serial data. Each frame contains a start bit, seven data bits, parity, and one stop bit. In this figure, a frame, which contains one ASCII character, has 10-bits. Most dial-up communications systems, such as CompuServe, Prodigy, and America



* = STOP
ST = START

FIGURE 10–42 Asynchronous serial data

The 16550 is able to function in simplex, half-duplex, or full-duplex modes. One of the main features of the 16550 is its internal receiver and transmitter FIFO (first-in, first-out) memories. Because each is 16 bytes deep, the UART only requires attention from the microprocessor after receiving 16 bytes of data. It also holds 16 bytes before the microprocessor must wait for the transmitter. The FIFO makes this UART ideal when interfacing to high-speed systems, because less time is required to service it.

An example **simplex** system is where the transmitter or receiver is used by itself, such as in an FM (frequency modulation) radio station. An example half-duplex system is a CB (citizens band) radio where we transmit and receive, but not both at the same time. The full-duplex system allows transmission and reception in both directions simultaneously. An example **full-duplex** system is the telephone.

The 16550 can control a **modem** (modulator/demodulator), which is a device that converts TTL levels of serial data into audio tones that can pass through the telephone system. Six pins on the 16650 are devoted to modem control: $\overline{\text{DSR}}$ **(data set ready),** $\overline{\text{DTR}}$ **(data terminal ready),** $\overline{\text{CTS}}$ **(clear-to-send),** $\overline{\text{RTS}}$ **(request-to-send),** $\overline{\text{RI}}$ **(ring indicator),** and $\overline{\text{DCD}}$ **(data carrier detect).** The modem is referred to as the *data set*, and the 16550 is referred to as the *data terminal.*

16550 Pin Functions

A_0, A_1, A_2 — The address inputs are used to select an internal register for programming and also for data transfer. Refer to Table 10–7 for a list of each combination of the address inputs and the registers selected.

$\overline{\text{ADS}}$ The **address strobe** input is used to latch the address lines and chip select lines. If not needed (as in the Intel system), connect this pin to ground. The $\overline{\text{ADS}}$ pin is designed for use with Motorola microprocessors.

$\overline{\text{BAUDOUT}}$ The **Baud out** pin is where the clock signal generated by the Baud rate generator from the transmitter section is made available. It is most often connected to the RCLK input to generate a receiver clock that is equal to the transmitter clock.

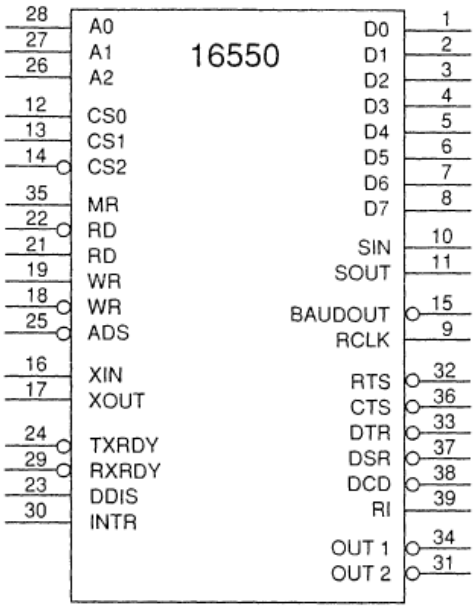


FIGURE 10–43 The pin-out of the 16550 UART

CS0, CS1, $\overline{\text{CS2}}$ The **chip select** inputs must all be active to enable the 16550 UART.

$\overline{\text{CTS}}$ The **clear to send** (if low) indicates that the modem or data set is ready to exchange information. This pin is often used in a half-duplex system to turn the line around.

D7–D0 The **data bus** pins are connected to the microprocessor data bus.

$\overline{\text{DCD}}$ The **data carrier detect** input is used by the modem to signal the 16550 that a carrier is present.

DDIS The **disable driver** output becomes a logic 0 to indicate that the microprocessor is reading data from the UART. DDIS can be used to change the direction of data flow through a buffer.

$\overline{\text{DSR}}$ **Data set ready** is an input to the 16550 that indicates the modem or data set is ready to operate.

$\overline{\text{DTR}}$ **Data terminal ready** is an output that indicates that the data terminal (16550) is ready to function.

INTR **Interrupt request** is an output to the microprocessor used to request an interrupt (INTR = 1) whenever the 16550 has a receiver error, has received data, and if the transmitter is empty.

MR **Master reset** initializes the 16550 and should be connected to the system RESET signal.

$\overline{\text{OUT1}}$, $\overline{\text{OUT2}}$ These are user-defined output pins that can provide signals to a modem

or any other device as needed in a system.

RCLK — **Receiver clock** is the clock input to the receiver section of the UART. This input is always 16 × the desired receiver Baud rate.

RD, $\overline{RD}$ — **Read** inputs (either may be used) cause data to be read from the register specified by the address inputs to the UART.

$\overline{RI}$ — The **ring indicator** input is placed at the logic 0 level by the modem to indicate that the telephone is ringing.

$\overline{RTS}$ — **Request-to-send** is a signal to the modem indicating that the UART wishes to send data.

SIN, SOUT — These are the **serial data** pins. SIN accepts serial data and SOUT transmits serial data.

$\overline{RXRDY}$ — **Receiver ready** is a signal used to transfer received data via DMA techniques (see text).

$\overline{TXRDY}$ — **Transmitter ready** is a signal used to transfer transmitter data via DMA techniques (see text).

WR, $\overline{WR}$ — **Write** (either may be used) connects (either) to the microprocessor write signal to transfer commands and data to the 16550.

XIN, XOUT — These are the **main clock connections.** A crystal is connected across these pins to form a crystal oscillator, or XIN is connected to an external timing source.

TABLE 10–7 The registers selected by A_0, A_1, and A_2

A_2	A_1	A_0	*Register*
0	0	0	Receiver buffer (read) and transmitter holding (write)
0	0	1	Interrupt enable
0	1	0	Interrupt identification (read) and FIFO control (write)
0	1	1	Line control
1	0	0	Modem control
1	0	1	Line status
1	1	0	Modem status
1	1	1	Scratch

Initializing the 16550. Initialization dialog, which occurs after a hardware or software reset, consists of two parts: programming the line control register and the Baud rate generator. The line control register selects the number of data bits, number of stop bits, and parity (whether it is even or odd or if parity is sent as a one or a zero). The Baud rate generator is programmed with a divisor that determines the Baud rate of the transmitter section.

FIGURE 10–44 The contents of the 16550 line control register

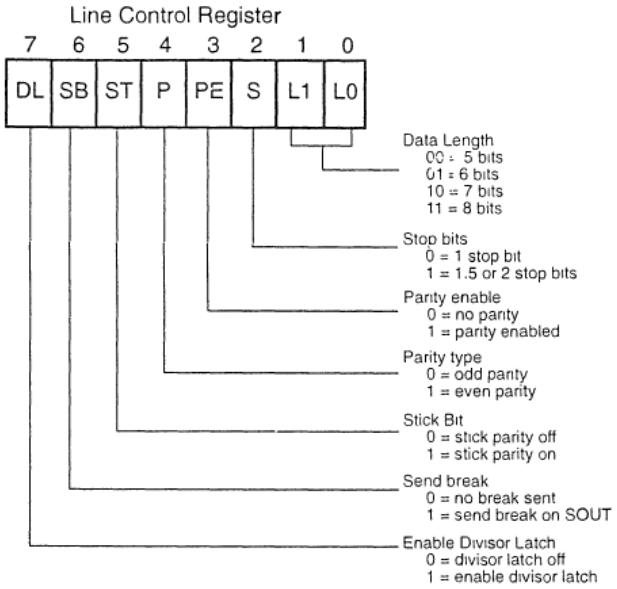


Figure 10–44 illustrates the line control register. The line control register is programmed by outputting information to I/O port 011 (A_2, A_1, A_0). The rightmost two bits of the line control register select the number of transmitted data bits (5, 6, 7, or 8). The number of stop bits are selected by S in the line control register. If S = 0, one stop bit is used; if S = 1, 1.5 stop bits are used for 5 data bits and 2 stop bits are used with 6, 7, or 8 data bits.

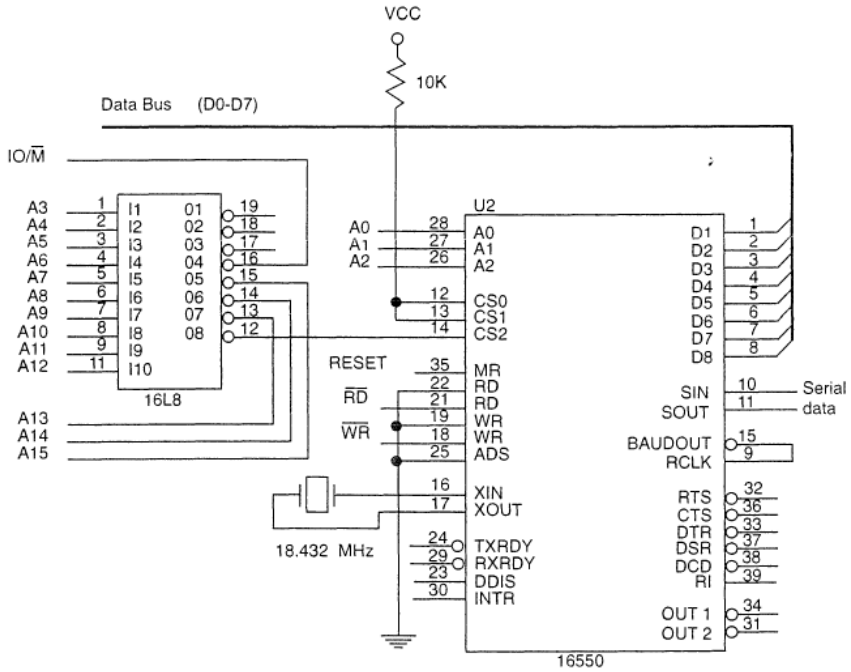
The next three bits are used together to send even or odd parity, to send no parity, or to send a 1 or a 0 in the parity bit position. To send even or odd parity, the ST *(stick)* bit must be placed at a logic 0 level and parity enable must be a logic 1. The value of the parity bit then determines even or odd parity. To send no parity (common in Internet connections), ST = 0 as well as the parity enable bit. This sends and receives data without parity. Finally, if a 1 or a 0 must be sent and received in the parity bit position for all data, ST = 1, with a 1 in parity enable. To send a 1 in the parity bit position, place a 0 in the parity bit; to send a 0, place a 1 in the parity bit.

The remaining bits in the line control register are used to send a break and to select programming for the Baud rate divisor. If bit position 6 of the line control register is a logic 1, a break is transmitted. As long as this bit is a 1, the break is sent from the SOUT pin. A break by definition is at least two frames of logic 0 data. The software in the system is responsible for timing the transmission of the break. To end the break, bit position 6 or the line control register is returned to a logic 0 level. The Baud rate divisor is only programmable when bit position 7 of the line control register is a logic 1.

Programming the Baud Rate. The Baud rates obtains if a 18.432 MHz crystal is used as a timing source. It also shows the divisor values programmed into the Baud rate generator to obtain these Baud rates. The actual number programmed into the Baud rate generator causes it to produce a clock that is 16 times the desired Baud rate. For example, if 240 is programmed into the Baud rate divisor, the Baud rate is $^{18.432MHz}/_{16 \times 240}$ = 4800 Baud.

Fig. 10-45 shows the interface between 8088 microprocessor using a PAL16L8 to decode 8b port address F0h and F7H.

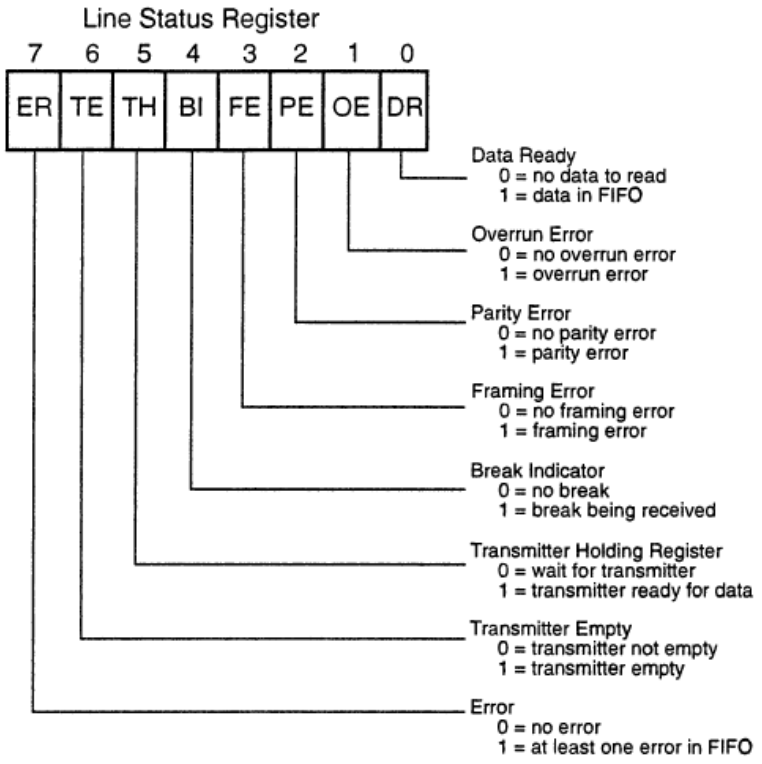
FIGURE 10–45 The 16550 interfaced to the 8088 microprocessor at ports 00F0H–00F7H



After the programming the line control register and Baud rate divisor, we must program FIFO control register.

Sending Serial Data. Before serial data can be sent or received through the 16550, we need to know the function of the line status register (see Figure 10–47). The line status register contains information about error conditions and the state of the transmitter and receiver. This register is tested before a byte is transmitted or can be received.

FIGURE 10–47 The contents of the line status register of the 16550 UART



Receiving Serial Data. To read received information from the 16550, we test the DR bit of the line status register. Example 10–26 lists a procedure that tests the DR bit to decide if the 16550 has received any data. Upon reception of data, the procedure tests for errors. If an error is detected, the procedure returns with AL equal to an ASCII '?'. If no error has occurred, then the procedure returns with AL equal to the received character.

EXAMPLE 10–26

```
                        ;A procedure that receives data from the 16550 UART
                        ;and returns it in AL.
                        ;
= 00F5                  LSTAT EQU   0F5H            ;line status port
= 00F0                  DATA  EQU   0F0H            ;data port

0000                    RECV  PROC  NEAR

0000  E4 F5                   IN    AL,LSTAT        ;get line status register
0002  A8 01                   TEST  AL,1            ;test DR bit
0004  74 FA                   JZ    RECV            ;if no data in receiver

0006  A8 0E                   TEST  AL,0EH          ;test all 3 error bits
0008  75 03                   JNZ   ERR             ;for an error

000A  E4 F0                   IN    AL,DATA         ;read data from 16550
000C  C3                      RET
000D   ERR:
000D  B0 3F                   MOV   AL,'?'          ;get question mark
000F  C3                      RET

0010   REVC   ENDP
```

EXAMPLE 10–25

```
                        ;A Procedure that transmits AH via the 16550 UART.
                        ;
= 00F5                  LSTAT EQU   0F5H          ;line status port
= 00F0                  DATA  EQU   0F0H          ;data port

0000                    SEND  PROC  NEAR

0000  50                      PUSH  AX            ;save AX
0001  E4 F5                   IN    AL,LSTAT      ;get line status register
0003  A8 20                   TEST  AL,20H        ;test TH bit
0005  74 FA                   JZ    SEND          ;if transmitter not ready

0007  8A C4                   MOV   AL,AH         ;get data
0009  E6 F0                   OUT   DATA,AL       ;transmit data
000B  58                      POP   AX            ;restore AX
000C  C3                      RET

000D                    SEND  ENDP
```

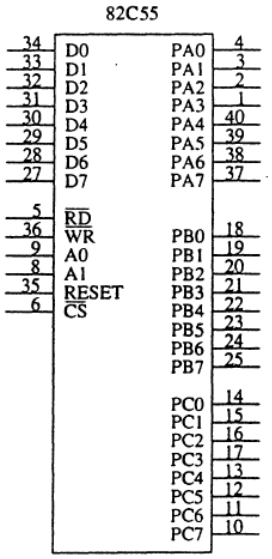
I8255A programmable parallel interface

The 82C55 **programmable peripheral interface** (PPI) is a very popular low-cost interfacing component found in many applications. The PPI has 24 pins for I/O, programmable in groups of 12 pins, that are used in three separate modes of operation. The 82C55A can interface any TTL-compatible I/O device to the microprocessor. The 82C55A (CMOS version) requires the insertion wait states if operated with a microprocessor using higher than an 8 MHz clock. It also provides at least 2.5 mA of sink (logic 0) current at each output with a maximum of 4.0 mA. Because I/O devices are inherently slow, wait states used during I/O transfers do not impact significantly upon the speed of the system. The 82C55 still finds application (compatible for programming, although it may not appear in the system as a discrete 82C55) even in the latest 80486- or Pentium Pro-based computer system. The 82C55 is used for interface to the keyboard and the parallel printer port in many of these personal computers. It also controls the timer and reads data from the keyboard interface.

Figure 10–13 illustrates the pin-out diagram of the 82C55. Its three I/O ports (labeled A, B, and C) are programmed in groups of 12 pins. Group A connections consist of port A (PA_7–PA_0) and the upper half of port C (PC_7–PC_4), and group B consists of port B (PB_7–PB_0) and the lower half of port C (PC_3–PC_0). The 82C55 is selected by its CS pin for programming and for reading or writing to a port. Register selection is accomplished through the A_1 and A_0 input pins, which select an internal register for programming or operation. Table 10–2 shows the I/O port assignments used for programming and access to the I/O ports. In the personal computer, an 82C55 or its equivalent is decoded at I/O ports 60H–63H.

Initialization.

FIGURE 10–13 The pin-out of the 82C55 peripheral interface adapter (PPI)



The 82C55 is a fairly simple device to interface to the microprocessor and program. For the 82C55 to be read or written, the $\overline{CS}$ input must be a logic 0 and the correct I/O address must be applied to the A_1 and A_0 pins. The remaining port address pins are don't cares as far as the 82C55 is concerned, and are externally decoded to select the 82C55.

Figure 10–14 shows an 82C55 connected to the 80386SX so that it functions at 8-bit I/O port addresses C0H (port A), C2H (port B), C4H (port C), and C6H (command register). This interface uses the low bank of the 80386SX I/O map. Notice from this interface that all of the 82C55 pins are direct connections to the 80386SX, except for the $\overline{CS}$ pin. The $\overline{CS}$ pin is decoded and selected by a 74ALS138 decoder.

TABLE 10–2 I/O port assignments for the 8255

A_1	A_0	*Function*
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Command Register

The RESET input to the 82C55 initializes the device whenever the microprocessor is reset. A RESET input to the 82C55 causes all ports to be set up as simple input ports using mode 0 operation. Because the port pins are internally programmed as input pins on a reset, damage is prevented when the power is first applied to the system. After a RESET, no other commands are needed to program the 82C55 as long as it is used as an input device at all three ports. Note that an 82C55 is interfaced to the personal computer at port addresses 60H–63H for keyboard control and also for controlling the speaker, timer, and other internal devices such as memory expansion. This is true for any AT or earlier style of personal computer system.

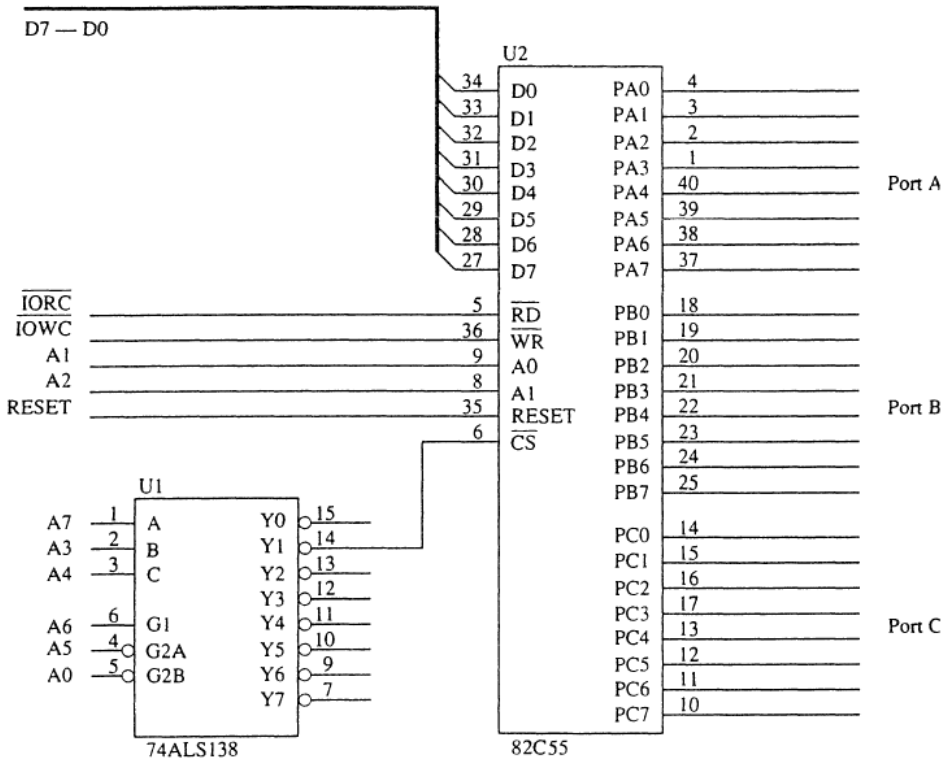


FIGURE 10–14 The 82C55 interfaced to the low bank of the 80386SX microprocessor

Programming the 82C55

The 82C55 is easy to program because it contains only two internal command registers, as illustrated in Figure 10–15. Notice that bit position 7 selects either command byte A or command byte B. Command byte A programs the function of group A and B, while command byte B sets (1) or resets (0) bits of port C only if the 82C55 is programmed in mode 1 or 2.

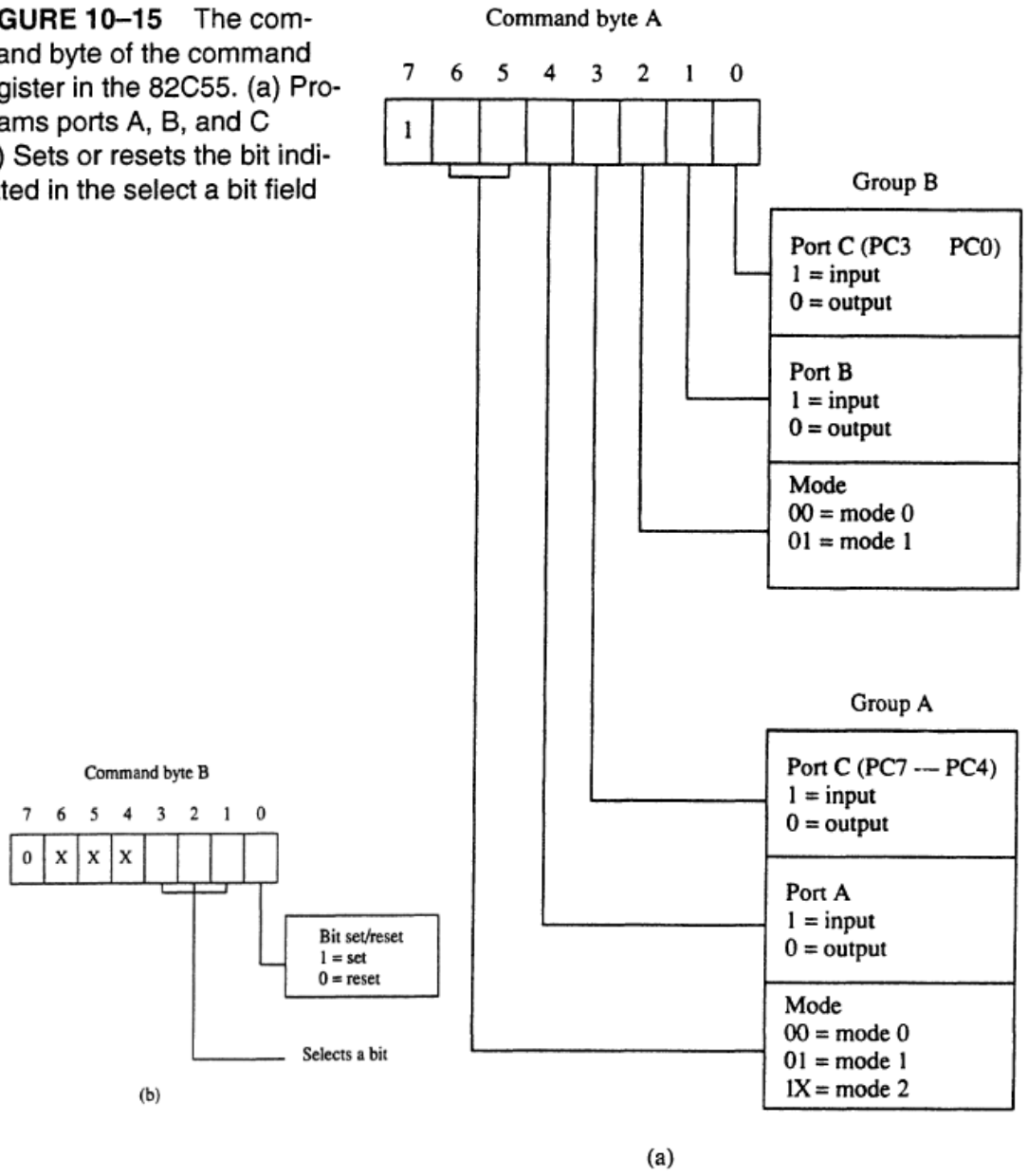


FIGURE 10–15 The command byte of the command register in the 82C55. (a) Programs ports A, B, and C (b) Sets or resets the bit indicated in the select a bit field

Group B pins (port B and the lower part of port C) are programmed as either input or output pins. Group B can operate in either mode 0 or mode 1. Mode 0 is the basic input/output mode that allows the pins of group B to be programmed as simple input and latched output connections. Mode 1 operation is the strobed operation for group B connections, where data are transferred through port B and handshaking signals are provided by port C.

Group A pins (port A and the upper part of port C) are also programmed as either input or output pins. The difference is that group A can operate in modes 0, 1, and 2. Mode 2 operation is a bi-directional mode of operation for port A.

If a 0 is placed in bit position 7 of the command byte, command byte B is selected. This command allows any bit of port C to be set (1) or reset (0) if the 82C55 is operated in either mode 1 or 2. Otherwise, this command byte is not used for programming. We often use the bit set/reset function in control system to set or clear a control bit at port C.

Mode 0 Operation

Mode 0 operation causes the 82C55 to function as either a buffered input device or as a latched output device. These are the same as the basic input and output circuits discussed in the first section .

How to program in this mode port A and port B as outputs, is shown at Example 10-8.

EXAMPLE 10–8

```
                    ;programming the 82C55 PIA
                    ;
0000   B0 80              MOV    AL,10000000B
0002   BA 0703            MOV    DX,703H          ;address command
0005   EE                 OUT    DX,AL            ;program 82C55
```

Mode 1 Strobed Input

Mode 1 operation causes port A and/or port B to function as latching input devices. This allows external data to be stored into the port until the microprocessor is ready to retrieve it. Port C is also used in mode 1 operation, not for data, but for control or handshaking signals that help operate either or both port A and port B as strobed input ports. Figure 10–21 shows how both ports are structured for mode 1 strobed input operation and also the timing diagram.

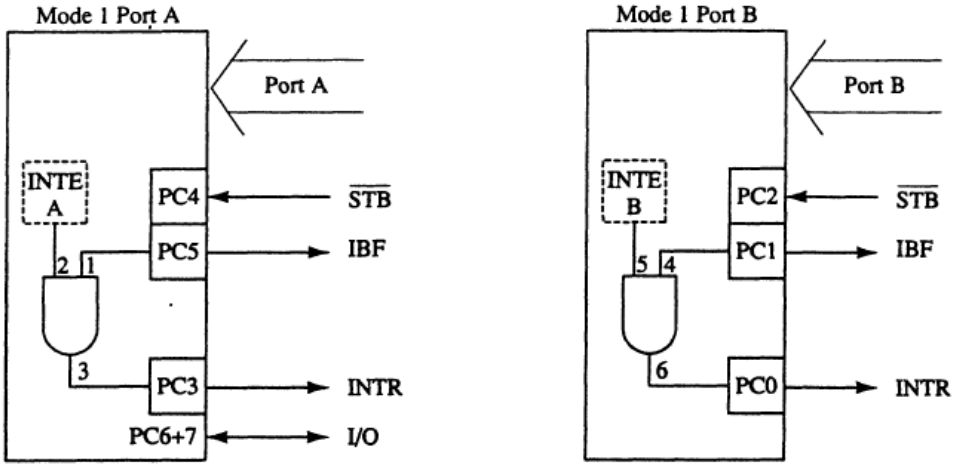
The strobed input port captures data from the port pins when the strobe ($\overline{STB}$) is activated. Note that strobe captures the port data on the 0-to-1 transition. The $\overline{STB}$ signal causes data to be captured in the port and also activates the IBF **(input buffer full)** and INTR **(interrupt request)** signals. Once the microprocessor, through software (IBF) or hardware (INTR), notices that data are strobed into the port, it executes an IN instruction to read the port ($\overline{RD}$). The act of reading the port restores both IBF and INTR to their inactive states until the next datum is strobed into the port.

Signal Definitions for Mode 1 Strobed Input

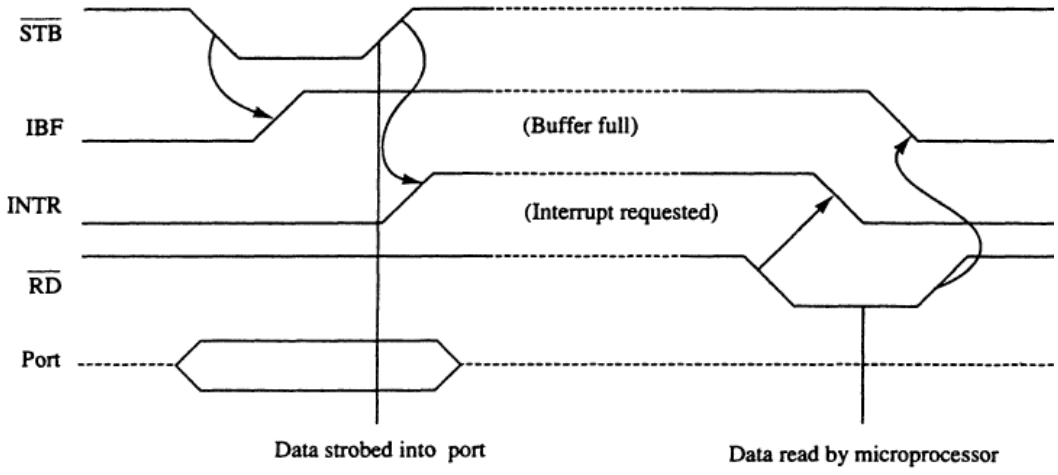
$\overline{STB}$ The **strobe** input loads data into the port latch, which holds the information until it is input to the microprocessor via the IN instruction.

IBF **Input buffer full** is an output that indicates the input latch contains information.

INTR **Interrupt request** is an output that requests an interrupt. The INTR pin becomes a logic 1 when the $\overline{STB}$ input returns to a logic 1 and is cleared when the data are input from the port by the microprocessor.



(a)



(b)

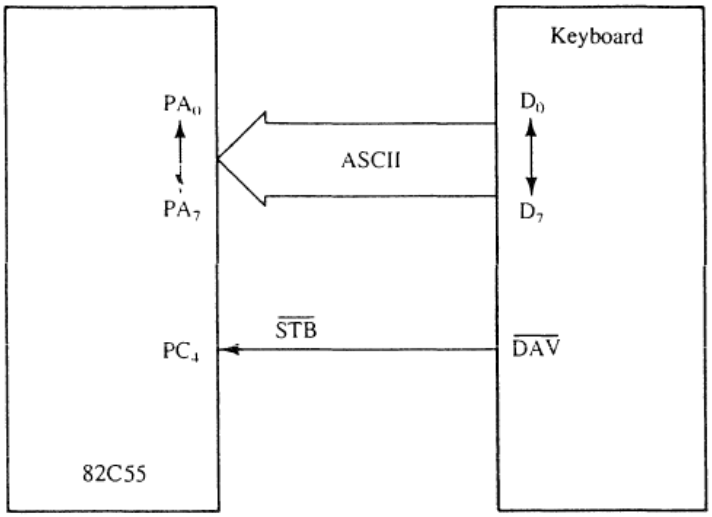
FIGURE 10–21 Strobed input operation (mode 1) of the 82C55. (a) Internal structure, and (b) timing diagram

INTE The **interrupt enable** signal is neither an input nor an output, but an internal bit programmed via the port PC4 (port A) or PC2 (port B) bit position.

PC_7, PC_6 The port C pins 7 and 6 are general purpose I/O pins that are available for any purpose.

Strobed Input Example. An excellent example of a strobed input device is a keyboard. The keyboard encoder de-bounces the key-switches and provides a strobe signal whenever a key is depressed and the data output contain the ASCII-coded key code. Figure 10–22 illustrates a keyboard connected to strobed input port A. Here $\overline{DAV}$ **(data available)** is activated for 1 μs each time that a key is typed on the keyboard. This causes data to be strobed into port A because $\overline{DAV}$ is connected to the $\overline{STB}$ input of port A. Each time a key is typed, therefore, it is stored into port A of the 82C55. The $\overline{STB}$ input also activates the IBF signal, indicating that data are in port A.

FIGURE 10–22 Using the 82C55 for strobed input operation of a keyboard



Mode 1 Strobed Output

Figure 10–23 illustrates the internal configuration and timing diagram of the 82C55 when it is operated as a strobed output device under mode 1. Strobed output operation is similar to mode 0 output, except that control signals are included to provide handshaking.

Whenever data are written to a port programmed as a strobed output port, the $\overline{\text{OBF}}$ **(output buffer full)** signal becomes a logic 0 to indicate that data are present in the port latch. This signal indicates that data are available to an external I/O device that removes the data by strobing the $\overline{\text{ACK}}$ **(acknowledge)** input to the port. The $\overline{\text{ACK}}$ signal returns the $\overline{\text{OBF}}$ signal to a logic 1, indicating that the buffer is not full.

Signal Definitions for Mode 1 Strobed Output

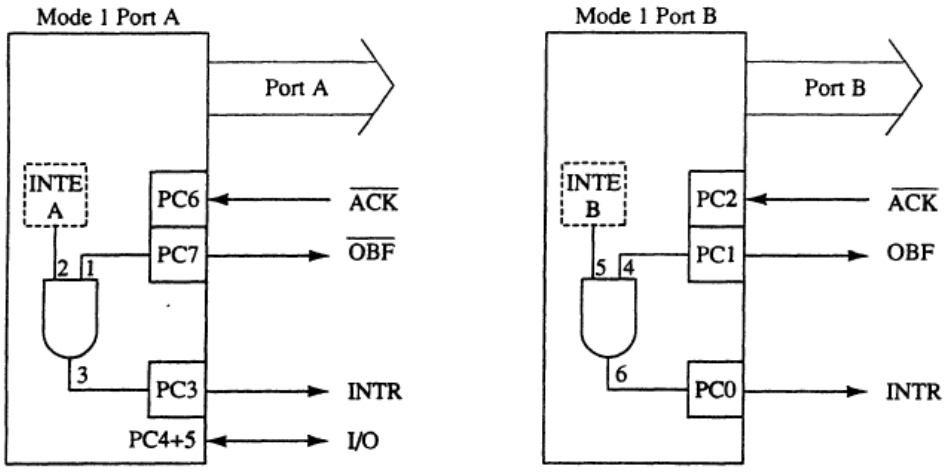
$\overline{\text{OBF}}$ — **Output buffer full** is an output that goes low whenever data are output (OUT) to the port A or port B latch. This signal is set to a logic 1 whenever the $\overline{\text{ACK}}$ pulse returns from the external device.

$\overline{\text{ACK}}$ — The **acknowledge** signal causes the $\overline{\text{OBF}}$ pin to return to a logic 1 level. The $\overline{\text{ACK}}$ is a response from an external device that indicates it has received the data from the 82C55 port.

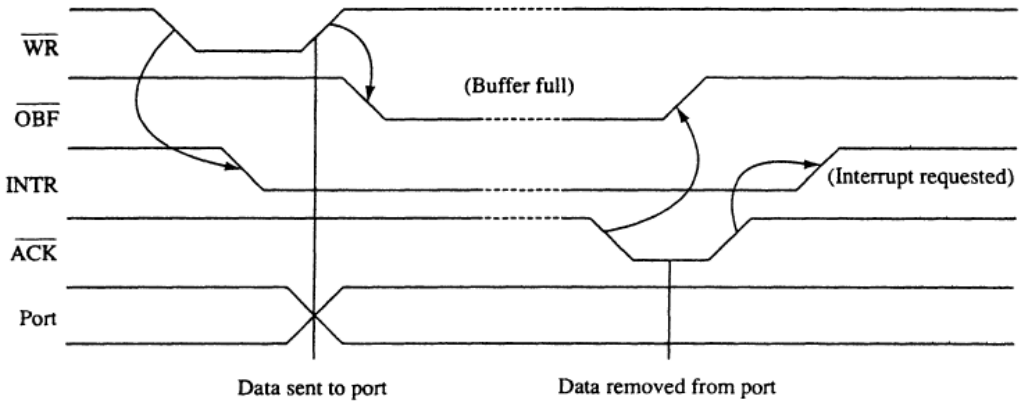
INTR — **Interrupt request** is a signal that often interrupts the microprocessor when the external device receives the data via the $\overline{\text{ACK}}$ signal. This pin is qualified by the internal INTE (interrupt enable) bit.

INTE — **Interrupt enable** is neither an input nor an output, but an internal bit programmed to enable or disable the INTR pin. The INTE A bit is programmed as PC_6, and INTE B is PC_2.

PC5, PC4 — **Port C bits 5 and 4** are general purpose I/O pins. The bit set and reset command may be used to set or reset these two pins.



(a)



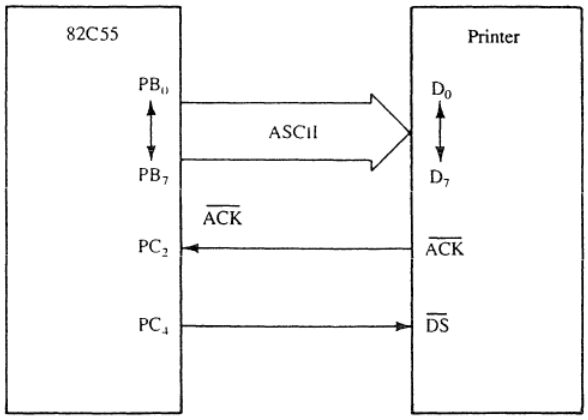
(b)

FIGURE 10–23 Strobed output operation (mode 1) of the 82C55. (a) Internal structure, and (b) timing diagram

Figure 10–24 illustrates port B connected to a parallel printer with eight data inputs for receiving ASCII-coded data, a $\overline{DS}$ (**data strobe**) input to strobe data into the printer, and an $\overline{ACK}$ output to acknowledge the receipt of the ASCII character.

In this circuit, there is no signal to generate the $\overline{DS}$ signal to the printer, so PC_4 is used with software that generates the $\overline{DS}$ signal. The $\overline{ACK}$ signal that is returned from the printer acknowledges the receipt of the data and is connected to the $\overline{ACK}$ input of the 82C55.

FIGURE 10–24 The 82C55 connected to a parallel printer interface that illustrates the strobed output mode of operation for the 82C55



Mode 2 Bidirectional Operation

In mode 2, which is allowed with group A only, port A becomes bi-directional, allowing data to be transmitted and received over the same eight wires. Bi-directional bused data are useful when interfacing two computers. It is also used for the IEEE-488 parallel high speed GPIB (**general purpose instrumentation bus**) interface standard. Figure 10–25 shows the internal structure and timing diagram for mode 2 bi-directional operation.

Signal Definitions for Bi-directional Mode 2

INTR **Interrupt request** is an output used to interrupt the microprocessor for both input and output conditions.

$\overline{\text{OBF}}$ **Output buffer full** is an output that indicates that the output buffer contains data for the bi-directional bus.

$\overline{\text{ACK}}$ **Acknowledge** is an input that enables the three-state buffers so that data can appear on port A. If $\overline{\text{ACK}}$ is a logic 1, the output buffers of port A are at their high-impedance state.

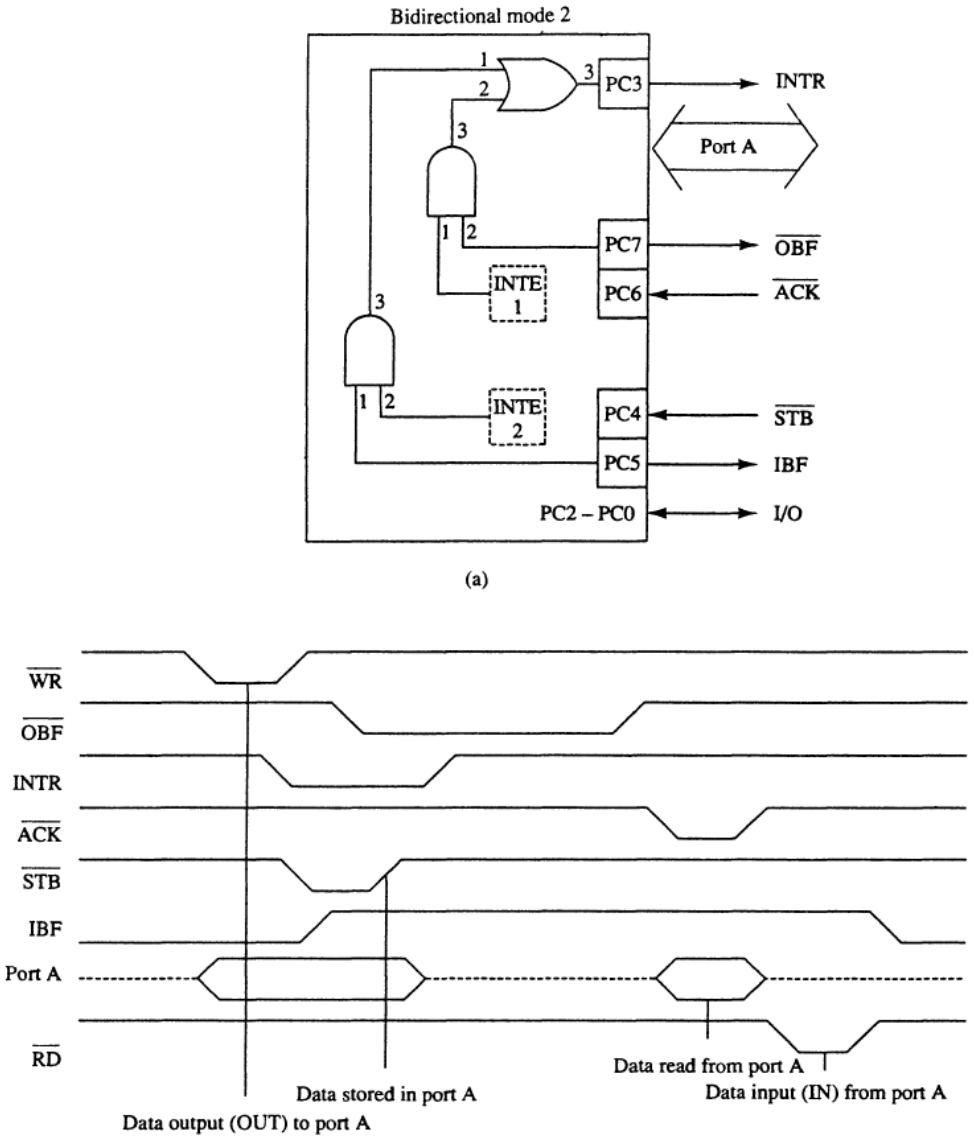


FIGURE 10–25 Mode 2 operation of the 82C55. (a) Internal structure, and (b) timing diagram

$\overline{\text{STB}}$	The **strobe** input loads the port A input latch with external data from the bi-directional port A bus.
IBF	**Input buffer full** is an output used to signal that the input buffer contains data for the external bi-directional bus.
INTE	**Interrupt enable bits** are internal (INTE1 and INTE2) and enab·e the INTR pin. The state of the INTR pin is controlled through port C bits PC_6 (INTE1) and PC_4 (INTE2).
PC2, PC1, and PC0	These bits are general-purpose I/O pins in mode 2 controlled by the bit set and reset command.

The Bi-directional Bus. The bi-directional bus is used by referencing port A with the IN and OUT instructions. To transmit data through the bi-directional bus, the program first tests the $\overline{\text{OBF}}$ signal to determine whether the output buffer is empty. If it is, then data are sent to the output buffer via the OUT instruction. The external circuitry also monitors the $\overline{\text{OBF}}$ signal to decide if the microprocessor has sent data to the bus. As soon as the output circuitry sees a logic 0 on $\overline{\text{OBF}}$, it sends back the $\overline{\text{ACK}}$ signal to remove it from the output buffer. The $\overline{\text{ACK}}$ signal sets the $\overline{\text{OBF}}$ bit and also enables the three-state output buffers so that data may be read.

82C55 Mode Summary

Figure 10–26 shows a graphical summary of the three modes of operation for the 82C55. Mode 0 provides simple I/O, mode 1 provides strobed I/O, and mode 2 provides bi-directional I/O. As mentioned, these modes are selected through the command register of the 82C55.

FIGURE 10–26 A summary of the port connections for the 82C55 PIA

		Mode 0		Mode 1		Mode 2
Port A		IN	OUT	IN	OUT	I/O
Port B		IN	OUT	IN	OUT	Not used
Port C	0	IN	OUT	$INTR_B$	$INTR_B$	I/O
	1			IBF_B	$\overline{OBF}_B$	I/O
	2			$\overline{STB}_B$	$\overline{ACK}_B$	I/O
	3			$INTR_A$	$INTR_A$	INTR
	4			$\overline{STB}_A$	I/O	$\overline{STB}$
	5			IBF_A	I/O	IBF
	6			I/O	$\overline{ACK}_A$	$\overline{ACK}$
	7			I/O	$\overline{OBF}_A$	$\overline{OBF}$