

Programmable I/O Interface

The 16550 Programmable Communications Interface

Microprocessor System Design, Chapter 11-3

Slide 1

The 16550 UART

- The National Semiconductor Corporation's PC16550D is a **universal asynchronous receiver/transmitter (UART)** designed to connect to virtually any type of serial interface.

The pin-out of the 16550 UART

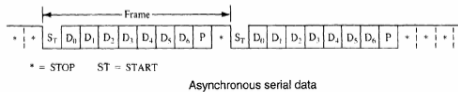
16550		
28	A0	D0
27	A1	D1
26	A2	D2
		D3
		D4
		D5
		D6
		D7
22	CS0	SIN
12	CS1	SOUT
13	CS2	BAUDOUT
14		RCLK
35	WR	RTS
22	RD	CTS
21	RD	DTR
19	WR	DSR
18	WR	DCD
29	ADS	RI
16	XIN	OUT 1
17	XOUT	OUT 2
24	TXRDY	
25	RXRDY	
22	DDIS	
30	INTR	

Microprocessor System Design, Chapter 11-3

Slide 2

Asynchronous Serial Data

- Asynchronous serial data** are transmitted and received **without a clock or timing signal**.



- **Start bit:** 1 bit, logic 0
- **Data bit:** 5-8 bits
- **Parity bit:** even, odd or no parity bit
- **Stop bit:** 1, 1.5 or 2 bits
- **Idle state:** logic 1

Microprocessor System Design, Chapter 11-3

Slide 3

The 16550 UART

- Two completely separate sections are responsible for data communications: the **transmitter** and the **receiver**.
- Since the transmitter and the receiver are independent of each other, the 16550 can function in **simplex**, **half-duplex** and **full-duplex** modes.

Microprocessor System Design, Chapter 11-3

Slide 4

The 16550 UART

Register Addresses			Register
A ₂	A ₁	A ₀	
0	0	0	Receiver Buffer (read), Transmitter Holding Register (write)
0	0	1	Interrupt Enable
0	1	0	Interrupt Identification (read)
0	1	0	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
0	0	0	Divisor Latch (least significant byte)
0	0	1	Divisor Latch (most significant byte)

Microprocessor System Design, Chapter 11-3

Slide 5

The 16550 UART

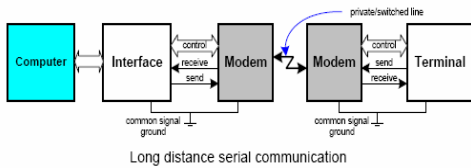
- The 16550
 - has **two 16-byte FIFO buffers** to reduce the attention of the microprocessor required to service it.
 - can **control a modem with 6 signal pins** which are devoted to modem control: **DSR**(data set ready), **DTR**(data terminal ready), **CTS**(clear-to-send), **RTS**(request-to-send), **RI**(ring indicator), **DCD**(data carrier detect).

Microprocessor System Design, Chapter 11-3

Slide 6

Long Distance Serial Communication

- The modem is referred to as the data set.
- The 16550 is referred to as the data terminal.



Microprocessor System Design, Chapter 11-3

Slide 7

Initializing the 16550 - Line Control Register

- Programming the **line control register** (A=011) and the **baud rate generator** (A=000/001).
- The **line control register** selects
 - the number of data bits,
 - number of stop bits, and
 - parity.

Microprocessor System Design, Chapter 11-3

Slide 8

Initializing the 16550 - Line Control Register

7	6	5	4	3	2	1	0
DL	SB	ST	P	PE	S	L1	L0

Data Length
 00 = 5 bits
 01 = 6 bits
 10 = 7 bits
 11 = 8 bits

Stop bits
 0 = 1 stop bit
 1 = 1.5 or 2 stop bits

Parity enable
 0 = no parity
 1 = parity enabled

Parity type
 0 = odd parity
 1 = even parity

Stick Bit
 0 = stick parity off
 1 = stick parity on

Send break
 0 = no break sent
 1 = send break on SOUT

Enable Divisor Latch
 0 = divisor latch off
 1 = enable divisor latch

The operation of the ST and parity bits

ST	P	PE	Function
0	0	0	No parity
0	0	1	Odd parity
0	1	0	No parity
0	1	1	Even parity
1	0	0	Undefined
1	0	1	Send/receive 1
1	1	0	Undefined
1	1	1	Send/receive 0

A break by definition is at least two frames of logic 0.

Microprocessor System Design, Chapter 11-3

Slide 9

Initializing the 16550 - The Baud rate Generator

- The **baud rate generator** is programmed with a divisor that determines the baud rate of the transmitter section.
- **Baud rate** is the number of bits transferred per second, including the start, data, parity and stop bits.
- The baud rate divisor is only programmable when bit position 7 of the line control register is a logic 1.

Microprocessor System Design, Chapter 11-3

Slide 10

Initializing the 16550 - The Baud rate Generator

- The baud rate generator is programmed at I/O address 000 and 001.
 - Ports 000 and 001 respectively hold the least significant byte and most significant byte of the 16-bit divisor.
- The actual number programmed into the baud rate generator causes it to produce a clock that is 16 times the desired baud rate. (e.g. A divisor value of 240 makes the baud rate be 18.432 MHz/16x240=4800 baud if a 18.432 MHz crystal is used as a timing source.)

Microprocessor System Design, Chapter 11-3

Slide 11

Initializing the 16550 - The Baud rate Generator

The divisor used with the Baud rate generator for an 18.432 MHz crystal illustrating common Baud rates	Baud rate	Divisor value
	110	10,473
	300	3,840
	1,200	920
	2,400	480
	4,800	240
	9,600	120
	19,200	60
	38,400	30
	57,600	20
	115,200	10

Microprocessor System Design, Chapter 11-3

Slide 12

Initializing the 16550 - Example

Suppose that an asynchronous system requires 7 data bits, odd parity, a baud rate of 9600 and 1 stop bit.

```

EXAMPLE
;Initialization Using the Figure 10-45.
;Baud rate 9600, 7 data, odd parity, one stop
= 00F2 LINES EQU 0F0H
= 00F0 LSR EQU 0F0H
= 00F8 MSR EQU 0F0H
= 00F2 FIFO EQU 0F0H
0000 START PROC NEAR
0000 50 6A MOV AL,10001010 ;enable baud divisor
0002 88 F3 OUT LINE,AL
0004 80 78 MOV AL,120 ;program baud rate
0006 88 F0 OUT LSR,AL
0008 80 50 MOV AL,0
000A 88 F3 OUT MSR,AL
000C 80 6A MOV AL,00001100 ;program 7-data, odd
000E 88 23 OUT LSR,AL ;parity, one stop
0010 80 C7 MOV AL,00001110 ;enable transmitter and
0012 88 F2 OUT FIFO,AL ;receiver
0014 C3 RET
0015 START ENDP
    
```

Microprocessor System Design, Chapter 11-3 Slide 13

Programming the 16550 – FIFO Control Register

- The **FIFO control register (A=010)**
 - enables the transmitter and receiver,
 - clears the transmitter and receiver FIFOs, and
 - provides control for the 16550 interrupts.
- The transmitter and receiver can be independently controlled.

Microprocessor System Design, Chapter 11-3 Slide 14

Programming the 16550 – FIFO Control Register

The FIFO control register of the 16550 UART

Microprocessor System Design, Chapter 11-3 Slide 15

Programming the 16550 – Line Status Register

Sending serial data:

- The **line status register (A=101)** contains information about **error conditions** and **the state of the transmitter and receiver**.
- Before serial data can be sent or received through the 16550, the line status register must be tested.

Microprocessor System Design, Chapter 11-3 Slide 16

Programming the 16550 – Line Status Register

The contents of the line status register of the 16550 UART

Microprocessor System Design, Chapter 11-3 Slide 17

Example – Sending Data

```

;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
    
```

Microprocessor System Design, Chapter 11-3 Slide 18

Example – Receiving Data

- Before we read the received information from the 16550, the DR bit of the line status register should be tested.

EXAMPLE

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

Microprocessor System Design, Chapter 11-3

Slide 19

UART Errors

- The types of errors detected by the 16550 are parity error, framing error and overrun error.
- A **parity error** indicates the received data contained the wrong parity.
- A **framing error** indicates that the start bit and stop bits are not in their proper places.
- An **overrun error** indicates that data have overrun the internal receiver FIFO buffer.

Microprocessor System Design, Chapter 11-3

Slide 20