SECTION 14: SERIAL I/O

FUNCTION DESCRIPTION

The Secure Microcontroller, like the 8051, includes a powerful Serial I/O (UART) port capable of both synchronous and asynchronous communication. The baud rate and time–base source is fully programmable. The serial port uses P3.0 as Receive Data (RXD) and P3.1 Transmit Data (TXD). Note that no special action other than enabling the function (i.e. writing a logic 1 to the corresponding port pins) is required to make these pins become the serial port. The serial port is capable of full duplex operation in asynchronous mode and half–duplex operation in synchronous mode.

The serial port consists of a receive shift register, receive buffer, transmit shift register and control logic. An incoming serial word from an external source is shifted into the receive shift register one bit at a time. Bits are shifted at the baud rate, which is programmable. The baud rate must be programmed by user's software to match the incoming frequency or the serial data will be unintelligible. Once the word is received, the serial port transfers it into the receive buffer. At this time, the serial port can receive another byte into its shift register. Once a word is received, the software should read the receive buffer before another word is completely received. The serial port will automatically transfer the new word into the receive buffer regardless of whether software has read the old value. This destroys the data that had been present from the previous word. At 9600 baud, receiving an asynchronous word takes 1.04 ms. Thus software must read a received word within 1.04 ms or it may be overwritten by another incoming word.

The transmit shift register has no buffer. Software writes into the shift register and the word is immediately shifted out. Thus software must wait until the serial word is shifted out before writing another to transmit. Both the receive buffer and the transmit shift register are located in the SFR map. Furthermore, they reside at the same address called SBUF (99h). Reading SBUF automatically transfers the word out of the serial receive buffer. Writing to SBUF automatically transfers a byte into the transmit shift register. Serial Port operation is controlled via the SCON (98h) register.

Each serial port function (receive and transmit) is capable of generating an interrupt. If enabled, the receive function interrupts the CPU when a word has been shifted in. This indicates that software should read the receive buffer. The serial port will set the RI flag bit in the SCON.0 location to indicate the source of the interrupt. The serial port will also generate an interrupt when it has completely shifted out a word. This indicates that another word can be transmitted. The serial port will set the TI flag bit at SCON.1 to indicate the source of the interrupt. Remember that the Serial Interrupt vectors to location 23h regardless of the source. The ISR must determine the cause of the interrupt from the flags mentioned above.

SERIAL PORT OPERATING MODES Table 14–1

MODE	SYNC/ASYNC	BAUD CLOCK	DATA BITS	START/STOP	9TH DATA BIT FUNCTION
MODE 0	SYNC	12 t_{CLK}	8	None	None
MODE 1	ASYNC	Timer 1 Overflow	8	1 Start 1 Stop	None
MODE 2	ASYNC	32 t_{CLK} or 64 t_{CLK}	9	1 Start 1 Stop	0, 1, or parity
MODE 3	ASYNC	Timer 1 Overflow	9	1 Start 1 Stop	0, 1, or parity

The serial port has four modes (Mode 0–3) of operation as shown in Table 14–1. Mode 0, is a synchronous mode. This means that the microcontroller serial port will supply a clock to synchronize the data I/O shifting. One clock pulse is generated per bit. The external device that is communicating with the micro must also use this clock. This mode is typically used with serial peripherals. Synchronous mode is generally capable of a higher speed communication speeds than the asynchronous modes. It generates its speed as a fixed function of 12 microcontroller oscillator clocks per bit.

Mode 1 is a 10–bit asynchronous mode using 8–bit words, one start bit, and one stop bit. The time base is generated from the Timer 1 overflow and is therefore fully programmable. A user simply loads the timer with a

value that generates the required time interval at its overflow. This is the most common mode of communicating with a PC COM port or similar device. When talking to a PC in Mode 1, the PC would be set to 8–N–1 (8 bits, no parity, 1 stop). Common baud rates are 2400, 9600, and 19200 bps, but it can communicate as fast as 57,600 bps in Mode 1.

Mode 2 is an 11–bit asynchronous mode using 8 or 9–bit words and one stop bit. The time base offers a choice of two fixed relationships of either 32 or 64 oscillator clocks per bit. It is not otherwise programmable in speed. The 9th bit is selected manually. It can be set to a 1, 0, or parity. Thus Mode 2 could appear to have two stop bits by selecting the 9th bit to be a logic 1.

Mode 3 is similar to Modes 1 and 2. Like Mode 2, it uses 9–bit words instead of 8. Also like Mode 2, the 9th bit can be either 0, 1, or parity. Like Mode 1, it uses the Timer 1 mechanism to generate baud rates. This mode can be used with a PC COM port set for 8–N–2 (8 bits, no parity, two stop bits) by setting the 9th bit to a 1. It can also support 8E1 (8 bits, even parity, one stop). Parity is done by transferring the parity bit (PSW.0) to the 9th bit of the serial port (SCON.3). Since the CPU sets the parity bit to indicate an odd number of bits in the accumulator, a 9–bit serial word containing this parity bit would have even parity.

The serial port is controlled by the SCON register at SFR location 98h. These bits are described below in Figure 14–1. The serial port begins transmission after software writes to the SBUF register. Data is always shifted out with the LSB first. Each mode is discussed in detail below following Figure 14–1.

SERIAL PORT CONTROL REGISTER Figure 14–1

Bit Description:

SCON.7, SCON.6: **SM0, SM1**

"Mode Select": Used to select the operational mode of the serial I/O port as follows:

SM0	SM1	MODE	FUNCTION	WORD LENGTH	BAUD CLOCK PERIOD
0	0	Mode 0	Sync	8 bits	12 t_{CLK}
0	1	Mode 1	Async	10 bits	Timer 1 Overflow
1	0	Mode 2	Async	11 bits	64 t_{CLK} or 32 t_{CLK}
1	1	Mode 3	Async	11 bits	Timer 1 Overflow

Initialization: Cleared to a 0 on any type of reset.

SCON.5: **SM2**

"Multiple MCU Comm.": Used to enable the multiple microcontroller communications feature for Modes 2 and 3. When SM2=1, RI will be activated only when serial words are received which cause RB8 to be set to 1.

Initialization: Cleared to a 0 on any type of reset.

SCON.4: **REN**

"Receive Enable": When set to 1, the receive shift register will be enabled. Disabled when cleared to 0.

Initialization: Cleared to a 0 on any type of reset.

SCON.3: **TB8**

"Xmit Bit 8": Can be set or cleared to define the state of the 9th data bit in Modes 2 and 3 of a serial data word.

Initialization: Cleared to a 0 on any type of reset.

SCON.2: **RB8**

"Rcv. Bit 8": Indicates the state of the 9th data bit received while in Mode 2 or 3 operation. If Mode 1 is selected with SM2=0, RB8 is the state of the stop bit which was received. RB8 is not used in Mode 0.

Initialization: Cleared to a 0 on any type of reset.

SCON.1: **TI**

"Xmit Interrupt": Status bit used to signal that a word has been completely shifted out in Mode 0; it is set at the end of the 8th data bit. Set when the stop bit is transmitted.

Initialization: Cleared to a 0 on any type of reset.

SCON.0: **RI**

"Receive Interrupt": Status bit used to signal that a serial data word has been received and loaded into the receive buffer register. In Mode 0, it is set at the end of the 8th bit time. It is set at the mid–bit time of the incoming stop bit in all other modes of a valid received word according to the state of SM2.

Initialization: Cleared to a 0 on any type of reset.

BAUD RATE GENERATION

As shown in Table 14–1, the baud rate clock source for the serial I/O is determined by the selection of the operating mode.

In Modes 0 and 2, the baud rate is divided down from the clock oscillator frequency by a fixed value. In Mode 0, the baud rate is 1/12 of the clock oscillator frequency, or:

$$\text{Mode 0 Baud Rate} = \frac{1}{12t_{CLK}}$$

In Mode 2, the baud rate is either 1/32 or 1/64 of the clock oscillator frequency. This selection is dependent on the state of the SMOD bit (PCON.7). If SMOD=0, the baud rate will be 1/64 the clock oscillator frequency. If SMOD=1, the baud rate is 1/32 the clock oscillator frequency. This can also be given as:

$$\text{Mode 2 Baud Rate} = \frac{2^{SMOD}}{64} \text{ X } \frac{1}{t_{CLK}}$$

Note that 2^{SMOD} means two to the power of SMOD. $2^0=1$, $2^1=2$

The baud rates in Modes 1 and 3 are variable because they are a function of the Timer 1 overflow signal and the value of the SMOD bit. A general equation which describes the baud rate frequency can be given as:

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{SMOD}}{32} \text{ X } \frac{1}{t_{T1}}$$

where, t_{T1} is the overflow period of Timer 1. In this application Timer 1 can be configured in either the timer or the counter configuration. If the counter configuration is selected, then the baud rate frequency will be divided down from an external clock source applied to the P3.3 ($\overline{INT1}$) pin. As a general guideline, the GATE bit for Timer 1 should be 0 if the counter function is selected in this situation so that a continuous clock source will be available for the baud generator.

In most applications, Timer 1 will be configured as a timer which uses the internal clock oscillator frequency as its clock source. The baud rate will then be divided down from the time base applied to the XTAL1 and XTAL2 pins. In order to provide the most flexibility, Timer 1 should be programmed to operate in Mode 2 which configures TL1 as an 8–bit timer which is automatically reloaded with the value held in TH1 when its timeout condition is reached. This operational mode is selected by assigning the TMOD register control bits in the following configuration:

D7	D6	D5	D4	D3	D2	D1	D0
GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
0	0	1	0	X	X	X	X

In the configuration selected above, the baud rate for the serial port can be expressed as:

$$\text{Serial I/O Mode 1, 3 Baud Rate} = \frac{2^{SMOD}}{32} \times \frac{1}{12t_{CLK}\ (256 - (TH1))}$$

Table 14–2 lists some commonly used baud rates which can be derived by using Timer 1 in the timer configuration described above with a 11.059 MHz crystal as the time base.

TIMER 1 BAUD RATE GENERATION Table 14–2

BAUD RATE (BPS)	$1/t_{CLK}$ (MHz)	SMOD (PCON.7)	TIMER 1 C/$\overline{T}$	TIMER MODE	TH1
19200	11.059	1	0	2	0FDH
9600	11.059	0	0	2	0FDH
4800	11.059	0	0	2	0FAH
2400	11.059	0	0	2	0F4H
1200	11.059	0	0	2	0E8H
300	11.059	0	0	2	0A0H

When Timer 1 is used in this manner its interrupt should be disabled since the timeout period is much faster than is reasonable for interrupt response and service by the application software. See the application note at the end of this section.

SYNCHRONOUS OPERATION (MODE 0)

Mode 0 is the synchronous operating Mode 0 of the Serial I/O Port. It is intended primarily for transferring data to external shift registers or for communication with serial peripheral devices. The word length is eight bits on both transmit and receive. Serial data is both input and output on the RXD pin. Both transmit and receive data are synchronized to a clock signal which is output on the TXD pin at the serial data rate fixed at 1/12 of the frequency of the clock oscillator. A block diagram of the serial I/O port and timing waveforms for Mode 0 is shown in Figure 14–2 as a reference for the following discussion.

Serial data output is initiated following any instruction which causes data to be written to the Transmit Shift register located at the SBUF register address. At the time that data is written to the Transmit Shift register, a 1 is simultaneously written to the 9th bit position of the register (D8). The internal WRSBUF signal is pulsed during S6P2 and data is shifted out LSB first beginning at S6P2 of the next machine cycle. The contents of the Transmit Shift register are shifted to the right one position during S6P2 of every machine cycle until D7 has been output. As each shift right operation is performed, a 0 is shifted into the MSB position from the left. At the end of the D7 bit time, another shift is performed at S6P2 which loads the output latch of RXD with the 1 which

was originally written into bit position D8. During the final shift register operation, another 0 is shifted in from the left so that the Transmit Shift register contains all 0's. Also at this time, the Transmit Interrupt flag (TI) is set and a serial interrupt will be generated if enabled.

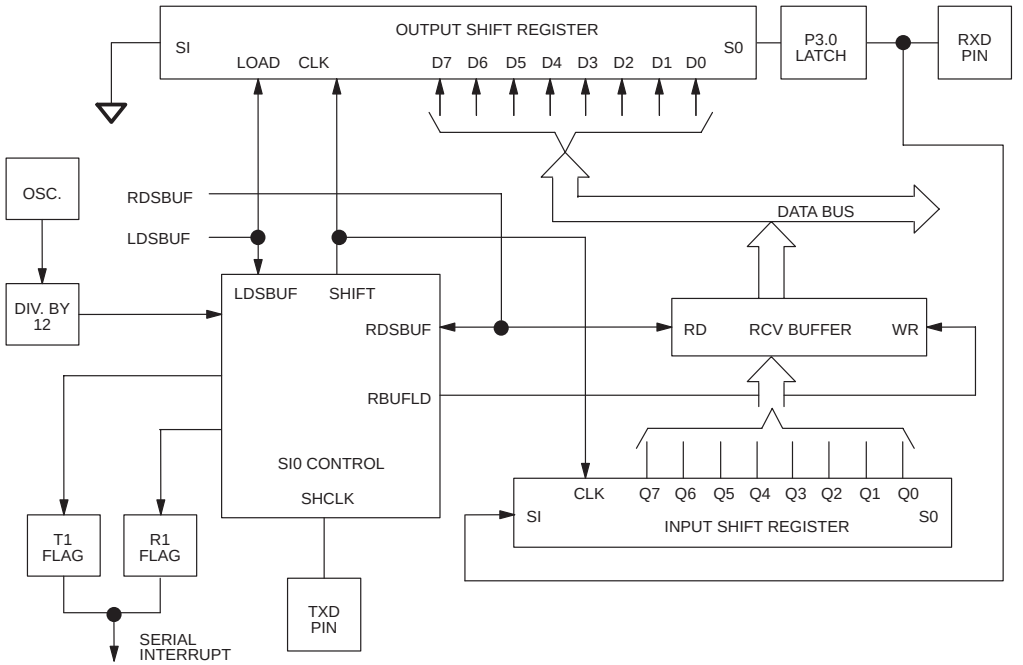
During serial data transmission in Mode 0, SHCLK is initially driven low onto the TXD pin at S3 of the machine cycle when D0 is output. During the time that the data word is shifted out, SHCLK will be low during S3, S4, and S5 and high during S6, S1, and S2 of every machine cycle.

A serial data word will be shifted into the Receive Shift register as soon as the condition REN=1 and RI=0 is satisfied. This condition can only be initiated by a write to the SCON register from the application software. At S6P2 of the second machine cycle following the write to SCON, the RXD pin will be sampled and the value (D0) will be shifted into the MSB position of the Receive Shift register. Seven more shifts will occur at S6P2 of subsequent machine cycles until the entire 8–bit word has been shifted into the Receive Shift register.

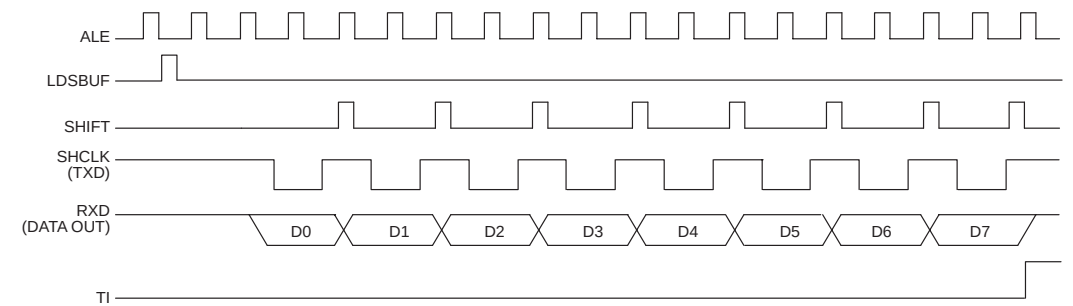
The SHCLK signal will be initially output low on the TXD pin starting at S3P1 of the same machine cycle in which D0 was sampled. As in the case described above for transmit, SHCLK will be low during S3, S4, and S5 and high during S6, S1, and S2 of every machine cycle.

After the last data bit (D7) has been shifted in, the control logic will immediately load the Receive Data Buffer at the SBUF register address with the contents of the Receive Shift register. At S1P1 of the 10th machine cycle following the write to SCON which initiated reception, the Receive Interrupt flag will be set and a serial interrupt will be generated if it has been enabled.

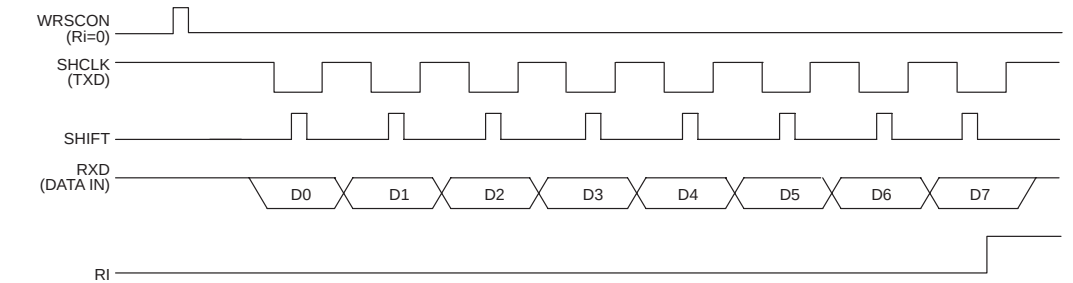
MODE 0 BLOCK DIAGRAM AND TIMING Figure 14–2



TRANSMIT TIMING:



RECEIVE TIMING:



ASYNCHRONOUS OPERATION

Mode 1, 2, and 3 provide asynchronous, full-duplex communication via the Serial I/O Port. The serial data word is either 10 or 11 bits long, depending on the mode selected. All three modes include one start bit, eight data bits, and one stop bit. Modes 2 and 3 include an additional, programmable 9th data bit. TXD is used for serial data output, while RXD is used for serial data input. In all three modes, the serial data word is both transmitted and received LSB first. The baud rate generator clock pulse (BRG clock) is derived either from the Timer 1 overflow output or divided directly from the clock oscillator frequency (of period t_{CLK}). The following description applies to all three of the operational modes. Figure 14–3 is a functional block diagram of the operation of the serial I/O port in Mode 1 including the timing waveforms which should be referred to in the discussion below.

Asynchronous serial data output begins whenever software writes to the SBUF register. When the write operation occurs at the time indicated by the WRSBUF signal in the timing diagram, the contents of the 8–bit data bus will be loaded into bits D8–D1 of the Transmit Shift register. Simultaneously, a 0 will be loaded into the D0 bit position of the shift register and a 1 will be loaded into the Stop bit position. (D9 for Mode 1, D10 for Modes 2 or 3).

During data transmission, the clocking frequency provided by the output of Timer 1 is divided down by a factor of 16 by the hardware to establish the serial output bit rate. Following the write operation to the Transmit Shift register, the LSB will be shifted out to the output latch of the TXD pin at the next time the divide–by–16 counter rolls over to zero. This counter is not synchronized to the machine cycles associated with instruction execution. As a result, data transmission will commence anywhere from 0 to 16 of the Baud Rate Generator clocks from the time that the Transmit Shift register is written. Successive bits from the Transmit Shift register will be shifted into the output latch of the TXD pin each time the divide–by–16 counter rolls over to zero. As each shift right operation is performed, a 0 is shifted into the MSB position from the left. When the Stop bit is shifted into the latch, the shifting operation is complete and the TI flag will be set. A serial interrupt will be generated if it has been enabled.

The Baud Rate Generator clock output is fed directly into the Bit Detector to perform serial data reception. Reception begins when a valid start bit of 0 is detected on the RXD pin. The Bit Detector will determine when this has occurred as follows: On each BRG clock pulse, the RXD pin will be sampled for a 1–to–0 transition. When such a transition is recognized, the Bit Detector will then reset its own internal divide–by–16 counter and sample the RXD pin on the 7th, 8th, and 9th BRG clock times following the transition. If a logic 0 level is detected on two out of these three sample times, a valid start bit is assumed. Otherwise, the Bit Detector will reject the incoming signal as a start bit and will repeat the process by searching for another 1–to–0 transition on RXD.

If a valid start bit is detected, the RXD pin will be sampled in the middle of each successive bit time until the entire 10–bit or 11–bit serial word has been received. Following the detection of a valid start bit, successive bit times begin each time that the Bit Detector's divide–by–16 counter rolls over to 0. During each bit time, the RXD pin is sampled on the 7th, 8th, and 9th BRG clock times. For the data bits, the logic level which is read at least two out of the three sample times by the Bit Detector will be the one which is shifted into the Receive Shift register. Just after the logic level is detected during the 10th bit time, the control logic will test to see if the following conditions are true:

a) The previous state of RI was 0.

b) SM2=0; or if SM2=1, then if the 10th received bit=1.

If these conditions are met during the 10th bit time, then the control logic will not perform another shift, but will instead load the contents of the Receive Shift register into the Receive Data Buffer, load the logic state determined at the Stop bit time into the RB8 status flag (if SM2=0), and set the RI bit. A serial interrupt will then be generated if it the appropriate enable bits have been set. If the above conditions are not satisfied during the stop bit time, then the received word is lost.

The first condition is interpreted by the control logic to mean an "overrun" condition has been detected. This means that a serial data word has been received before software read the previous word from the Receive Data Buffer. Only a hardware reset or writing logic 0 to the RI bit will clear RI. It is therefore recommended that software clear the RI bit after reading from the SBUF register. This signals the hardware that a properly received data word has been processed by the application soft-

ware. In an overrun condition with RI=1, the originally received word will remain in the Receive Data Buffer and all successively received data words will be lost.

When SM2=1, received data words will be selectively discarded in a manner depending on the asynchronous mode selected.

The operational details which are unique to each of the asynchronous modes are summarized below.

Mode 1

In Mode 1, the asynchronous serial data word is ten bits long, including one start bit, eight data bits, and one stop bit. The baud rate generation is derived from the Timer 1 overflow output and is therefore programmable. Figure 14–3 is a functional block diagram of the operation of the serial I/O port in Mode 1 operation including the timing waveform.

In Mode 1 operation, the SM2 bit may be used to discard a received serial data word in which a "framing error" is detected, i.e., when a valid stop bit has not been detected. When SM2=1. the incoming serial data word will be ignored unless the received Stop bit=1. If SM2=0, then the value of the received Stop bit will be loaded into the RB8 status flag so that it may be processed by the application software.

Mode 2 and 3

In Mode 2 and 3, the asynchronous serial data word is 11 bits long, including one start bit, eight data bits, a programmable 9th data bit, and one stop bit. For Mode 2, the Baud Rate Generator clock is programmable to either 1/32 or 1/64 of the clock oscillator frequency (f_{CLK}), depending on the state of the SMOD bit (PCON.7) as follows:

SMOD	BRG CLOCK
0	$\frac{f_{CLK}}{64}$
1	$\frac{f_{CLK}}{32}$

For Mode operation, the baud rate generator clock is the Timer 1 Overflow output as described for Mode 1.

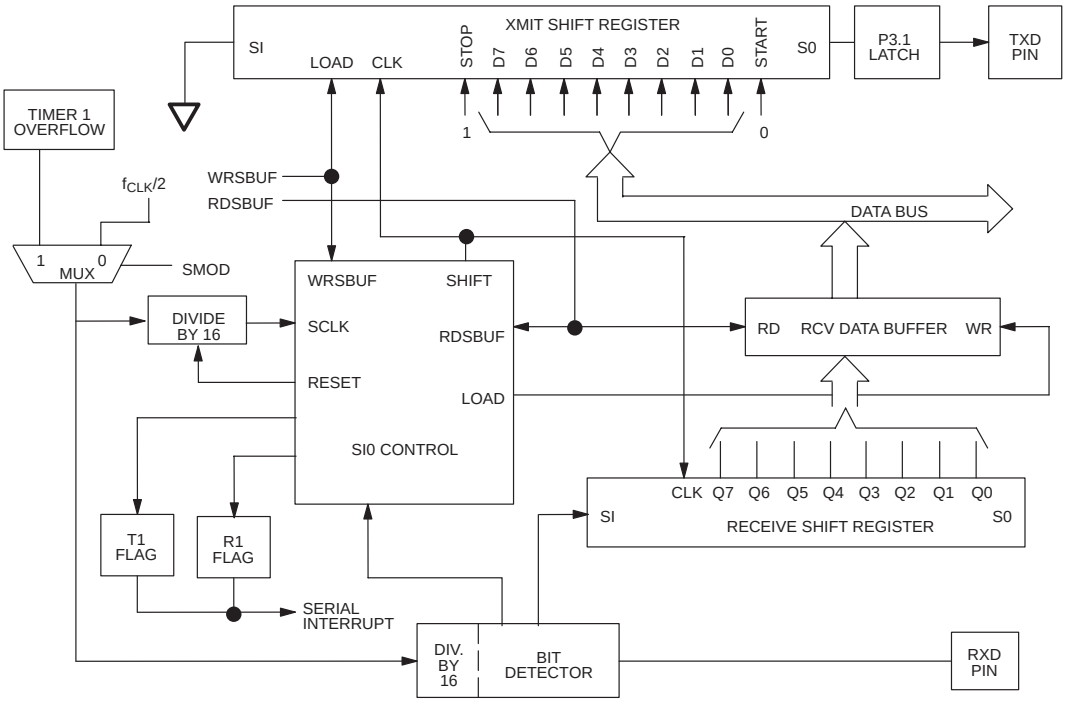
Transmission and reception takes place for Modes 2 and 3 as described except as noted below.

When the Transmit Shift register is written in Mode 2, the register is simultaneously written with a 0 in bit position D0 for a Start bit and a 1 is written into D10 for a Stop bit. D9 is the programmable bit which is written with the state of TB8 (SCON.3). TB8 can be written with the value of 1 or 0 by the application software.

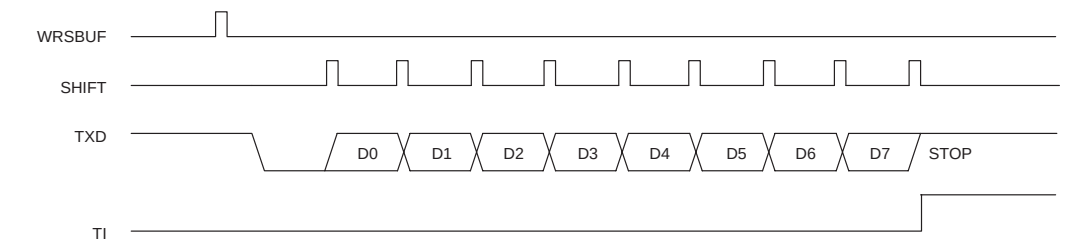
On receive, the eight data bits are shifted into the Receive Shift register following the detection of a valid Start bit. After the Stop bit has been detected, the Receive Data Buffer will be loaded with the contents of the Receive Shift register if RI=0 and SM2=0. Also at this time, the programmable 9th data bit will be loaded into RB8 in the SCON register. If RI=1 after the time the Stop bit is sample, then the incoming word will be lost.

The SM2 flag may be used in the implementation of a multiprocessor communication scheme by selectively discarding incoming serial data words according to the state of the programmable 9th data bit. When SM2=1, only those words in which this 9th bit is a 1 will be loaded into the Receive Data Buffer and cause a serial interrupt to be generated. Thus, the programmable 9th bit can be used to flag an incoming data character as an address field as opposed to a data field, for example.

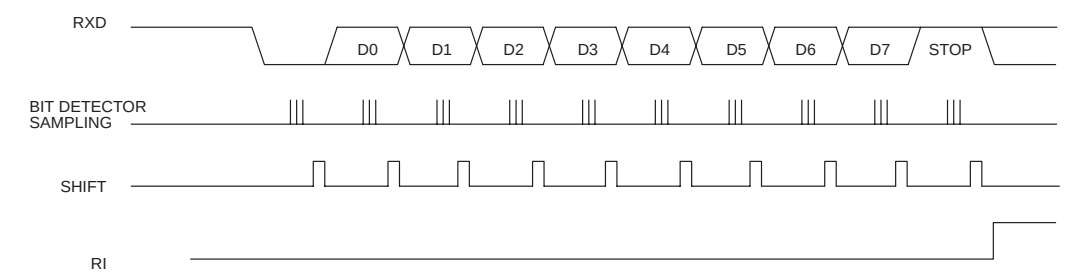
SERIAL PORT MODE 1 BLOCK DIAGRAM Figure 14–3



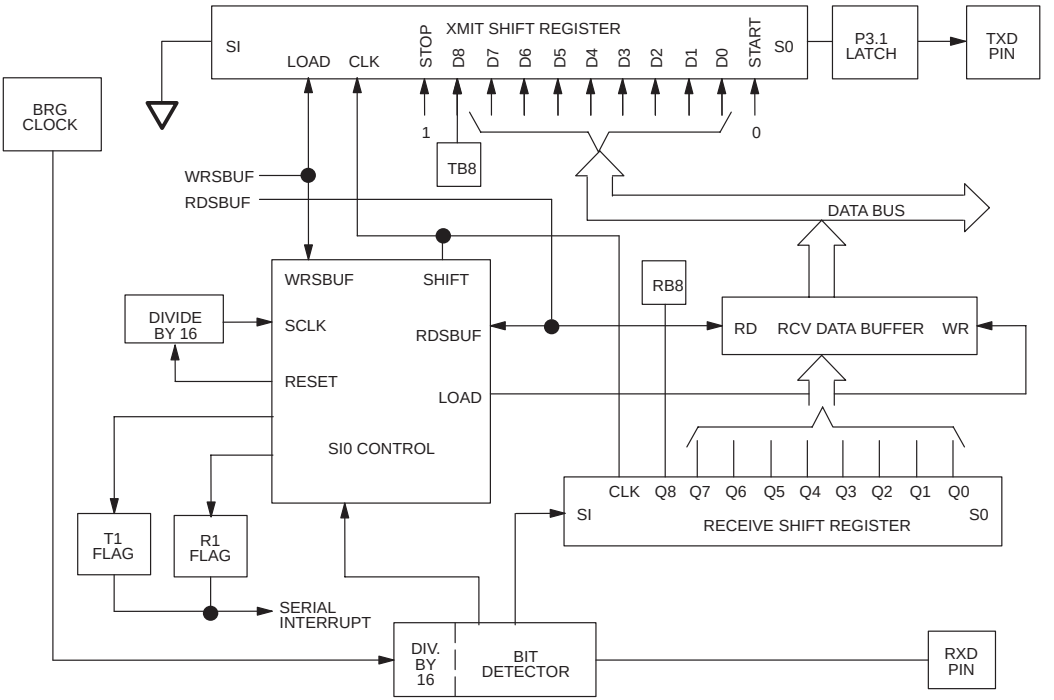
TRANSMIT TIMING:



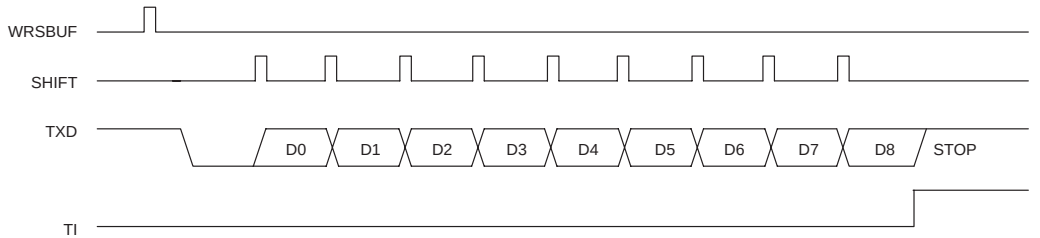
RECEIVE TIMING:



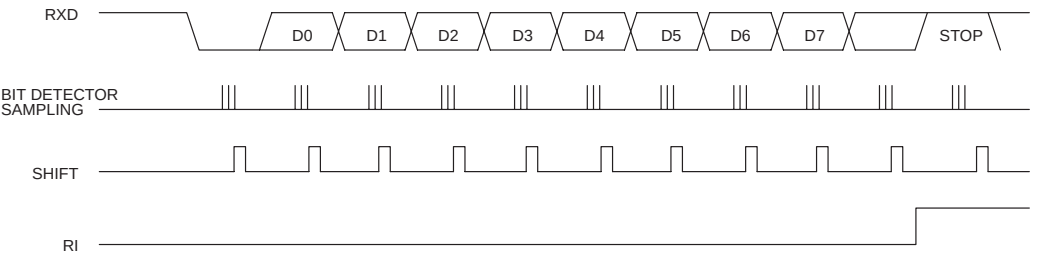
MODE2 AND 3 BLOCK DIAGRAM Figure 14–4



TRANSMIT TIMING:



RECEIVE TIMING:



APPLICATION: SERIAL PORT INITIALIZATION

The serial port can provide either synchronous or asynchronous serial communication. This note demonstrates how to initialize the serial port and includes an example showing how to perform asynchronous communication with a PC COM port.

A typical goal of microcontroller to PC communication is to transfer stored data from the nonvolatile RAM. This example will show how to move 256 bytes from NV RAM to the PC via the serial port. Once the 256 bytes have been received by the PC, it will send confirmation. For this example, the confirmation code will be A5h. The Microcontroller will run at 11.0592 MHz, a common crystal choice. This example will demonstrate both 9600 and 19,200 bps. A typical application has some form of error checking built into the data, so no parity is required. This code will therefore run at 8N1 or 8 bits, no parity, 1 stop bit. This is a common selection for PC terminal emulator software. Thus the setup summary is as follows:

Communication type :	Asynchronous
Baud Rate :	9600, 19200
Bits per word :	8
Stop bits :	1

As shown in the following table, this most closely corresponds to Serial Mode 1.

SERIAL I/O OPERATING MODES

MODE	SYNC/ASYNC	BAUD CLOCK	DATA BITS	START/STOP	9TH DATA BIT FUNCTION
MODE 0	SYNC	12 t_{CLK}	8	None	None
MODE 1	ASYNC	Timer 1 Overflow	8	1 Start 1 Stop	None
MODE 2	ASYNC	32 t_{CLK} or 64 t_{CLK}	9	1 Start 1 Stop	0, 1, or parity
MODE 3	ASYNC	Timer 1 Overflow	9	1 Start 1 Stop	0, 1, or parity

The Serial Port is controlled by the SCON register. Serial Interrupts will also be used. These are controlled by IE and IP. The setup for each SFR is shown below. In addition, Mode 1 is associated with Timer 1, which is controlled by TCON and TMOD.

Mode 1 is selected using the SCON register. The table from the SCON register shown below indicates that Mode 1 is selected by choosing the value SM0 = 0 and SM1 = 1.

SMO	SM1	MODE	FUNCTION	WORD LENGTH	BAUD CLOCK
0	0	Mode 0	Synchronous	8 bits	12 t_{CLK}
0	*1*	*Mode 1*	*Asynchronous*	*10 bits*	*Timer 1 Overflow*
1	0	Mode 2	Asynchronous	11 bits	32 or 64 t_{CLK}
1	1	Mode 3	Asynchronous	11 bits	Timer 1 Overflow

SM0 = 0 and SM1 = 1 corresponds to the value SCON.7 = 0 and SCON.6 = 1. In addition the since the application requires receiving data, the serial receiver must be enabled. This is done by setting the REN bit at SCON.4 to a logic 1. The remaining bits in SCON can be written to 0. Thus the value for SCON is 01010000b or 50h.

SCON – 98h

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
0	1	0	1	0	0	0	0

This application uses the serial interrupt. It serves two purposes. First, the software knows when a byte has been sent, so it knows when another can be written. Second, after the 256 bytes have been transmitted, the PC will respond. It is not know when this will occur and the software may have other tasks to attend. Therefore the serial interrupt will inform the microcontroller when the confirmation code has been received by the serial port. This example will enable only the serial Interrupt. It will be set for high priority since in a real system, other interrupts might be enabled.

IE – 0A8h

EA	–	–	ES	ET1	EX1	ET0	EX0
1	0	0	1	0	0	0	0

To enable interrupts, the EA bit must be set. In addition, the setting the ES bit turns on the Serial Interrupt. Thus the value 10010000b or 90h will enable serial interrupts. Note that although a timer will ultimately be used to generate serial baud rates, the timer interrupt is not used. As mentioned above, this example will use a high priority for the serial interrupt. This is done as follows.

IP – 0B8h

RWT	–	–	PS	PT1	PX1	PT0	PX0
0	0	0	1	0	0	0	0

The serial port interrupt has been set to a high priority by setting the PS bit to a logic 1. Thus the serial port is configured for high priority by writing a 00010000b or 10h to the IP register at location 0B8h.

The serial port is now configured. The only remaining task is to set the correct baud rate. The example stated above that the communication rate would be either 9600 or 19,200 baud. To generate baud rates in Serial Mode 1, the Timer 1 is used. The serial port uses the Timer 1 overflow, then divides this frequency by either 16 or 32 to generate the internal baud rate clock. Each time the Timer 1 value increments past 0FFh is considered an overflow. Due to the formula used for generating baud rates shown below, the 11.0592 MHz crystal assumed for this example generates good baud rate values. This is the most convenient and commonly used choice for generating baud rates. Other convenient values are 7.3728 MHz and 1.8432 MHz.

To get 9600 bits per second, the baud rate generator must create an interval af 1/9600 seconds = 1.0416 ms. To get 19,200 bits per second, the interval is 1/19200 = 520.83 µs. Note that the timers count up, so the value that the timer starts from must be selected to generate the desired interval. The following values are useful:

1/11.0592 MHz = t_{CLK} = 90.4 *10 $^{-9}$

Timer runs at 12 t_{CLK} per count = 1.085 * 10 $^{-6}$

Time out = (256 – Timer start value) * 12 t_{CLK} = (256 – Timer start value) * 1.085 * 10 $^{-6}$

serial port Baud rate clock = 1/baud rate = (16 or 32) * Time out

Whether 16 or 32 is used in the baud rate generator is determined by the SMOD bit at PCON.7. 16 is used for SMOD=1, and 32 is used for SMOD=0. This is commonly referred to by the expression (2^{SMOD})/32. Since SMOD is either 0 or 1, this value is either 1/32 or 2/32 respectively.

The user selects the time–out value and the setting for SMOD to set the baud rate. This is done as follows.

$$\text{Baud Rate} = \frac{2^{SMOD}}{32} \times \frac{1}{12t_{CLK} * (256 - TH1)}$$

This formula solves as :

$$TH1 = 256 - \frac{2^{SMOD}}{32 * 12\ t_{CLK} * BaudRate}$$

For 9600 = Baud rate, TH1 = FDh with SMOD = 0.

To create 19,200 baud, the SMOD bit should be set to a logic 1 with the same value for TH1. SMOD has the effect of doubling the baud rate for any time out value.

TH1 – 8Dh

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	0	1

PCON – 87h

SMOD	$\overline{POR}$	PFW	WTR	EPFW	EWT	STOP	IDL
0	X	X	X	X	X	X	X

TMOD – 89h

GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
0	0	1	0	X	X	X	X

As shown in the TCON description, setting M1 = 1 and M0 = 0 selects Timer 1 mode 2 which is the 8–bit auto–reload mode. In this example, Timer 0 is not used, so the lower four bits of TMOD are unused. Therefore the TMOD register can be written with 00100000b or 20h.

TCON – 88h

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
0	1	0	0	0	0	0	0

In summary the following SFRs are configured to enable the serial port for 9600 baud asynchronous operation:

TH1	FDh
SCON	50h
IE	90h
IP	10h
TMOD	20h
TCON	40h
PCON	00h (SMOD=0)

To set up the Serial Port for 19,200 baud, the only difference is that the SMOD bit at PCON.7 is set. Therefore, writing 80h to PCON will accomplish this.

The value for TH1 and SMOD have been determined. The only remaining task is to configure the Timer 1 for 8–bit auto reload operation. This will cause the timer to start counting from the TH1 value after each time out. The TMOD register is set as follows:

The remaining step is to enable the timer. Once this is done, the baud rate generator will be in operation and serial I/O can be performed. The TCON register is used to enable the timers as shown below. The TR1 bit is set to a logic 1 to enable the Timer 1 function. Writing a 01000000b or 40h to TCON will do this.

The software that configures the serial port can be simply seven move instruction the configure the SFRs mentioned above to the value as shown. This example will show this code in the context of performing the application described at the beginning of this note.

The application example described moving 256 bytes of data from memory to the serial port, then receiving a confirmation code of 0A5h. The memory will be assumed to be located in the MOVX RAM beginning at the Partition address. For this example, the Partition will be 4000h and the microcontroller will be a DS5000. Using a DS5001/2 would change the value written to the MCON to configure the Partition.

```
;This code example shows how to initialize the serial port and transmit /
; receive code as described above.
TA              Equ             0C7h
MCON            Equ             0C6h

Org             00h
Reset :
SJMP            Start

Org             23h
Serial_ISR :
CLR             RI              ;Clear receive flag
CLR             TI              ;Clear transmit flag
RETI                            ;No special processing, return to application

Org             30h
Start :
MOV             TA, #0AAh       ;Start Timed Access
MOV             TA, #55h        ; finish Timed Access
MOV             MCON, #88h      ;Select Partition at 4000h (DS5000)
CLR             RI              ;Initialize receive flag
CLR             TI              :Initialize transmit flag
MOV             SCON, #50h      ;Configure Serial Port for Mode 1 and enable receiver
MOV             TMOD, #20h      ;Configure the Timer 1 for 8–bit auto–reload
MOV             TCON, #40h      ;Enable the Timer 1
MOV             TH1, #0FDh      ;Set Baud Rate
ANL             PCON, #7Fh      ;Set SMOD=0
MOV             IP, #10h        ;Set Serial Interrupt to high priority
MOV             IE, #90h        ;Globally enable interrupts and Serial Interrupt

Send :
MOV             R0, #0FFh       ;Set loop counter to 255
MOV             DPTR, #4000h    ;Put the data pointer at the beginning of data memory

Send_Loop :
MOVX            A, @DPTR        ;Get data byte
MOV             SBUF, A         ;Transmit data byte
JNB             TI, $           ;Wait for serial word to be sent (interrupt)
INC             DPTR            ;Next byte to be sent
DJNZ            R0, Send_Loop   ;Decrement loop counter

Receive :
JNB             RI, $           ;Wait for incoming word (interrupt)
MOV             R1, SBUF        ;Get received byte
CJNE            R1, #0A5h, Send ;Check for confirmation code
                                ; and resend all data if wrong

End
```