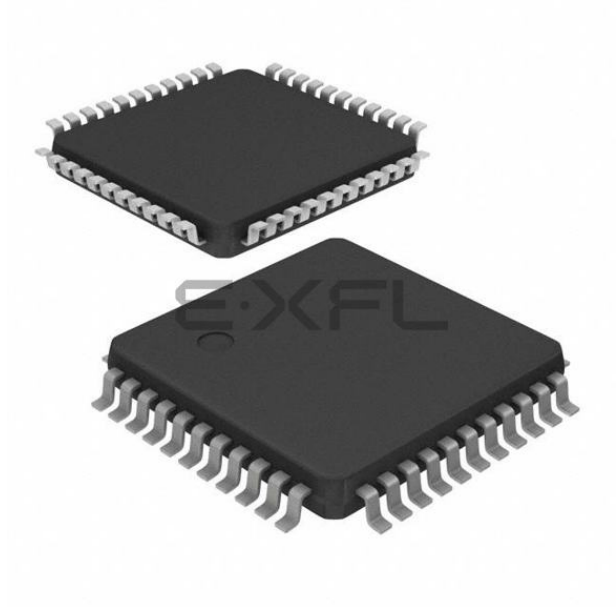


**Welcome to E-XFL.COM**

## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

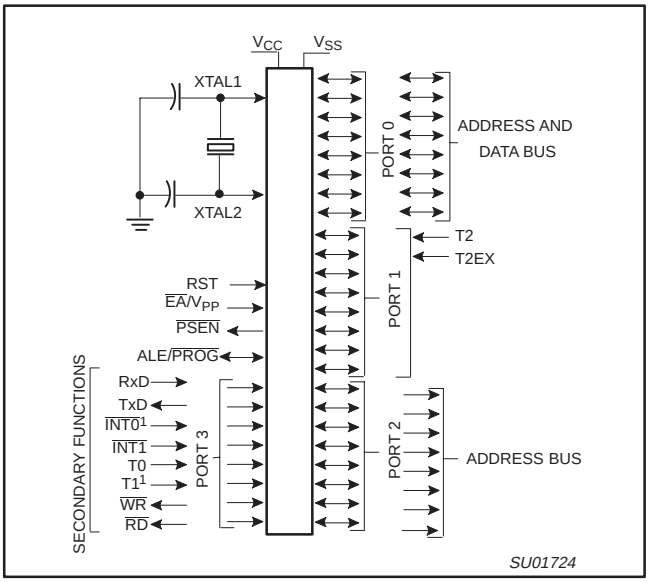
## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Obsolete |
| Core Processor | 8051 |
| Core Size | 8-Bit |
| Speed | 33MHz |
| Connectivity | EBI/EMI, UART/USART |
| Peripherals | POR |
| Number of I/O | 32 |
| Program Memory Size | 16KB (16K x 8) |
| Program Memory Type | OTP |
| EEPROM Size | - |
| RAM Size | 256 x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.7V ~ 5.5V |
| Data Converters | - |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-LQFP |
| Supplier Device Package | 44-LQFP (10x10) |
| Purchase URL | https://www.e-xfl.com/product-detail/nxp-semiconductors/p87c54x2fbd-157 |

## FEATURES

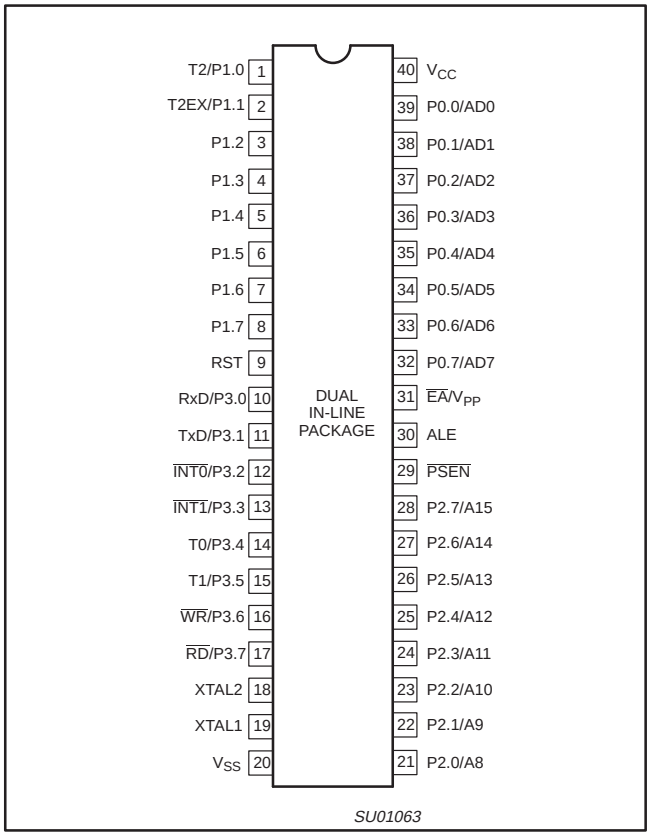
- 80C51 Central Processing Unit
  - 4 kbytes ROM/EPROM (P80/P87C51X2)
  - 8 kbytes ROM/EPROM (P80/P87C52X2)
  - 16 kbytes ROM/EPROM (P80/P87C54X2)
  - 32 kbytes ROM/EPROM (P80/P87C58X2)
  - 128 byte RAM (P80/P87C51X2 and P80C31X2)
  - 256 byte RAM (P80/P87C52/54X2/58X2 and P80C32X2)
  - Boolean processor
  - Fully static operation
  - Low voltage (2.7 V to 5.5 V at 16 MHz) operation
- 12-clock operation with selectable 6-clock operation (via software or via parallel programmer)
- Memory addressing capability
  - Up to 64 kbytes ROM and 64 kbytes RAM
- Power control modes:
  - Clock can be stopped and resumed
  - Idle mode
  - Power-down mode
- CMOS and TTL compatible
- Two speed ranges at $V_{CC}$ = 5 V
  - 0 to 30 MHz with 6-clock operation
  - 0 to 33 MHz with 12-clock operation
- PLCC, DIP, TSSOP or LQFP packages
- Extended temperature ranges
- Dual Data Pointers
- Security bits:
  - ROM (2 bits)
  - OTP (3 bits)
- Encryption array - 64 bytes
- Four interrupt priority levels
- Six interrupt sources
- Four 8-bit I/O ports
- Full-duplex enhanced UART
  - Framing error detection
  - Automatic address recognition
- Three 16-bit timers/counters T0, T1 (standard 80C51) and additional T2 (capture and compare)
- Programmable clock-out pin
- Asynchronous port reset
- Low EMI (inhibit ALE, slew rate controlled outputs, and 6-clock mode)
- Wake-up from Power Down by an external interrupt.

## LOGIC SYMBOL




**NOTE:**
1. $\overline{INT0}$/P3.2 and T1/P3.5 are absent in the TSSOP38 package.

## PLASTIC DUAL IN-LINE PACKAGE PIN CONFIGURATIONS



## OSCILLATOR CHARACTERISTICS

### Using the oscillator

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured for use as an on-chip oscillator, as shown in the logic symbol.

To drive the device from an external clock source, XTAL1 should be driven while XTAL2 is left unconnected. However, minimum and maximum high and low times specified in the data sheet must be observed.

### Clock Control Register (CKCON)

This device provides control of the 6-clock/12-clock mode by both an SFR bit (bit X2 in register CKCON and an OTP bit (bit OX2). When X2 is 0, 12-clock mode is activated. By setting this bit to 1, the system is switching to 6-clock mode. Having this option implemented as SFR bit, it can be accessed anytime and changed to either value. Changing X2 from 0 to 1 will result in executing user code at twice the speed, since all system time intervals will be divided by 2. Changing back from 6-clock to 12-clock mode will slow down running code by a factor of 2.

The OTP clock control bit (OX2) activates the 6-clock mode when programmed using a parallel programmer, superceding the X2 bit (CKCON.0). Please also see Table 2 below.

**Table 2.**

| OX2 clock mode bit (can only be set by parallel programmer) | X2 bit (CKCON.0) | CPU clock mode |
|---|---|---|
| erased | 0 | 12-clock mode (default) |
| erased | 1 | 6-clock mode |
| programmed | X | 6-clock mode |

### Programmable Clock-Out

A 50% duty cycle clock can be programmed to be output on P1.0. This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed:

1. to input the external clock for Timer/Counter 2, or
2. to output a 50% duty cycle clock ranging from 61 Hz to 4 MHz at a 16 MHz operating frequency in 12-clock mode (122 Hz to 8 MHz in 6-clock mode).

To configure the Timer/Counter 2 as a clock generator, bit C/$\overline{T}$2 (in T2CON) must be cleared and bit T20E in T2MOD must be set. Bit TR2 (T2CON.2) also must be set to start the timer.

The Clock-Out frequency depends on the oscillator frequency and the reload value of Timer 2 capture registers (RCAP2H, RCAP2L) as shown in this equation:

$$\frac{\text{Oscillator Frequency}}{n \times (65536 - \text{RCAP2H, RCAP2L})}$$

Where:

n = 2 in 6-clock mode, 4 in 12-clock mode.

(RCAP2H,RCAP2L) = the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

In the Clock-Out mode Timer 2 roll-overs will not generate an interrupt. This is similar to when it is used as a baud-rate generator. It is possible to use Timer 2 as a baud-rate generator and a clock generator simultaneously. Note, however, that the baud-rate and the Clock-Out frequency will be the same.

## RESET

A reset is accomplished by holding the RST pin HIGH for at least two machine cycles (24 oscillator periods in 12-clock and 12 oscillator periods in 6-clock mode), while the oscillator is running. To insure a reliable power-up reset, the RST pin must be high long enough to allow the oscillator time to start up (normally a few milliseconds) plus two machine cycles. After the reset, the part runs in 12-clock mode, unless it has been set to 6-clock operation using a parallel programmer.

## LOW POWER MODES

### Stop Clock Mode

The static design enables the clock speed to be reduced down to 0 MHz (stopped). When the oscillator is stopped, the RAM and Special Function Registers retain their values. This mode allows step-by-step utilization and permits reduced system power consumption by lowering the clock frequency down to any value. For lowest power consumption the Power Down mode is suggested.

### Idle Mode

In idle mode (see Table 3), the CPU puts itself to sleep while all of the on-chip peripherals stay active. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

### Power-Down Mode

To save even more power, a Power Down mode (see Table 3) can be invoked by software. In this mode, the oscillator is stopped and the instruction that invoked Power Down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values down to 2.0 V and care must be taken to return $V_{CC}$ to the minimum specified operating voltages before the Power Down Mode is terminated.

Either a hardware reset or external interrupt can be used to exit from Power Down. Reset redefines all the SFRs but does not change the on-chip RAM. An external interrupt allows both the SFRs and the on-chip RAM to retain their values. WUPD (AUXR1.3–Wakeup from Power Down) enables or disables the wakeup from power down with external interrupt. Where:

WUPD = 0: Disable
WUPD = 1: Enable

To properly terminate Power Down, the reset or external interrupt should not be executed before $V_{CC}$ is restored to its normal operating level and must be held active long enough for the oscillator to restart and stabilize (normally less than 10 ms).

To terminate Power Down with an external interrupt, $\overline{INT0}$ or $\overline{INT1}$ must be enabled and configured as level-sensitive. Holding the pin low restarts the oscillator but bringing the pin back high completes the exit. Once the interrupt is serviced, the next instruction to be executed after RETI will be the one following the instruction that put the device into Power Down.

**TMOD** Address = 89H Reset Value = 00H

Not Bit Addressable

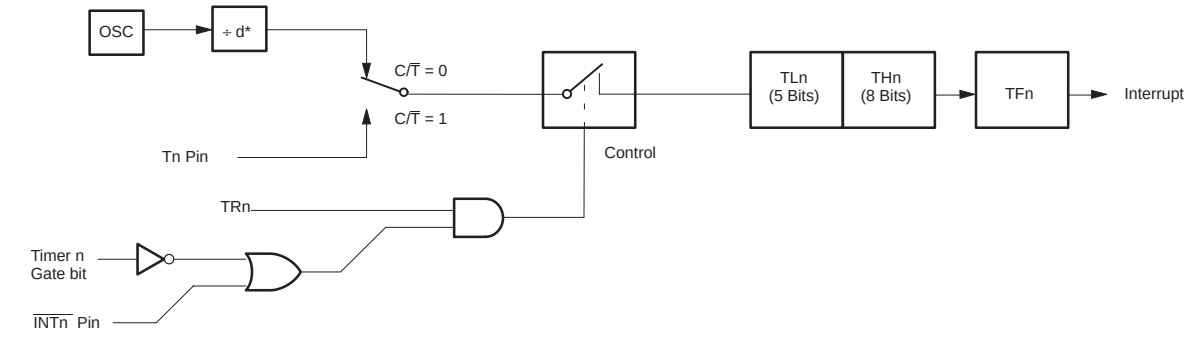
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| GATE | C/$\overline{T}$ | M1 | M0 | GATE | C/$\overline{T}$ | M1 | M0 |
| **TIMER 1** | | | | **TIMER 0** | | | |

| BIT | SYMBOL | FUNCTION |
|---|---|---|
| TMOD.3/ TMOD.7 | GATE | Gating control when set. Timer/Counter "n" is enabled only while "$\overline{INTn}$" pin is high and "TRn" control pin is set. when cleared Timer "n" is enabled whenever "TRn" control bit is set. |
| TMOD.2/ TMOD.6 | C/$\overline{T}$ | Timer or Counter Selector cleared for Timer operation (input from internal system clock.) Set for Counter operation (input from "Tn" input pin). |

| M1 | M0 | OPERATING |
|---|---|---|
| 0 | 0 | 8048 Timer: "TLn" serves as 5-bit prescaler. |
| 0 | 1 | 16-bit Timer/Counter: "THn" and "TLn" are cascaded; there is no prescaler. |
| 1 | 0 | 8-bit auto-reload Timer/Counter: "THn" holds a value which is to be reloaded into "TLn" each time it overflows. |
| 1 | 1 | (Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. TH0 is an 8-bit timer only controlled by Timer 1 control bits. |
| 1 | 1 | (Timer 1) Timer/Counter 1 stopped. |

*SU01580*

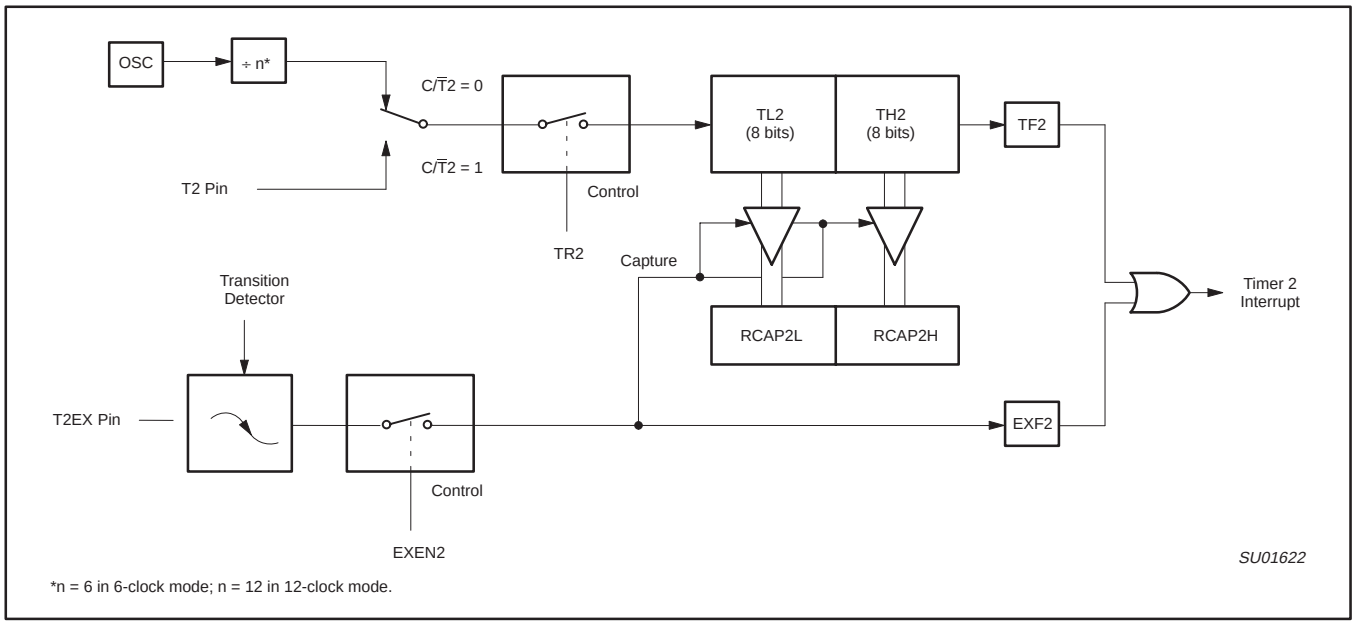
**Figure 1. Timer/Counter 0/1 Mode Control (TMOD) Register**




*d = 6 in 6-clock mode; d = 12 in 12-clock mode.

*SU01618*

**Figure 2. Timer/Counter 0/1 Mode 0: 13-Bit Timer/Counter**




*n = 6 in 6-clock mode; n = 12 in 12-clock mode.

**Figure 7. Timer 2 in Capture Mode**

**T2MOD** Address = 0C9H Reset Value = XXXX XX00B

Not Bit Addressable

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | T2OE | DCEN |

| Symbol | Position | Function |
|---|---|---|
| — | | Not implemented, reserved for future use.* |
| T2OE | T2MOD.1 | Timer 2 Output Enable bit. |
| DCEN | T2MOD.0 | Down Count Enable bit. When set, this allows Timer 2 to be configured as an up/down counter. |

* User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.

SU01519

**Figure 8. Timer 2 Mode (T2MOD) Control Register**

**Table 5. Timer 2 Generated Commonly Used Baud Rates**

| Baud Rate | | Osc Freq | Timer 2 | |
|---|---|---|---|---|
| 12-clk mode | 6-clk mode | | RCAP2H | RCAP2L |
| 375 K | 750 K | 12 MHz | FF | FF |
| 9.6 K | 19.2 K | 12 MHz | FF | D9 |
| 4.8 K | 9.6 K | 12 MHz | FF | B2 |
| 2.4 K | 4.8 K | 12 MHz | FF | 64 |
| 1.2 K | 2.4 K | 12 MHz | FE | C8 |
| 300 | 600 | 12 MHz | FB | 1E |
| 110 | 220 | 12 MHz | F2 | AF |
| 300 | 600 | 6 MHz | FD | 8F |
| 110 | 220 | 6 MHz | F9 | 57 |

## Summary Of Baud Rate Equations

Timer 2 is in baud rate generating mode. If Timer 2 is being clocked through pin T2(P1.0) the baud rate is:

$$\text{Baud Rate} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

If Timer 2 is being clocked internally, the baud rate is:

$$\text{Baud Rate} = \frac{f_{OSC}}{[n \times [65536 - (RCAP2H, RCAP2L)]]}$$

Where:

n = 16 in 6-clock mode, 32 in 12-clock mode.

$f_{OSC}$= Oscillator Frequency

To obtain the reload value for RCAP2H and RCAP2L, the above equation can be rewritten as:

$$RCAP2H, RCAP2L = 65536 - \left(\frac{f_{OSC}}{n \times \text{Baud Rate}}\right)$$

## Timer/Counter 2 Set-up

Except for the baud rate generator mode, the values given for T2CON do not include the setting of the TR2 bit. Therefore, bit TR2 must be set, separately, to turn the timer on. See Table 6 for set-up of Timer 2 as a timer. Also see Table 7 for set-up of Timer 2 as a counter.

**Table 6. Timer 2 as a Timer**

| MODE | T2CON | |
|---|---|---|
| | INTERNAL CONTROL (Note 1) | EXTERNAL CONTROL (Note 2) |
| 16-bit Auto-Reload | 00H | 08H |
| 16-bit Capture | 01H | 09H |
| Baud rate generator receive and transmit same baud rate | 34H | 36H |
| Receive only | 24H | 26H |
| Transmit only | 14H | 16H |

**Table 7. Timer 2 as a Counter**

| MODE | TMOD | |
|---|---|---|
| | INTERNAL CONTROL (Note 1) | EXTERNAL CONTROL (Note 2) |
| 16-bit | 02H | 0AH |
| Auto-Reload | 03H | 0BH |

**NOTES:**
1. Capture/reload occurs only on timer/counter overflow.
2. Capture/reload occurs on timer/counter overflow and a 1-to-0 transition on T2EX (P1.1) pin except when Timer 2 is used in the baud rate generator mode.

## FULL-DUPLEX ENHANCED UART

### Standard UART operation

The serial port is full duplex, meaning it can transmit and receive simultaneously. It is also receive-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the register. (However, if the first byte still hasn't been read by the time reception of the second byte is complete, one of the bytes will be lost.) The serial port receive and transmit registers are both accessed at Special Function Register SBUF. Writing to SBUF loads the transmit register, and reading SBUF accesses a physically separate receive register.

The serial port can operate in 4 modes:

**Mode 0:** Serial data enters and exits through RxD. TxD outputs the shift clock. 8 bits are transmitted/received (LSB first). The baud rate is fixed at 1/12 the oscillator frequency in 12-clock mode or 1/6 the oscillator frequency in 6-clock mode.

**Mode 1:** 10 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB8 in Special Function Register SCON. The baud rate is variable.

**Mode 2:** 11 bits are transmitted (through TxD) or received (through RxD): start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On Transmit, the 9th data bit (TB8 in SCON) can be assigned the value of 0 or 1. Or, for example, the parity bit (P, in the PSW) could be moved into TB8. On receive, the 9th data bit goes into RB8 in Special Function Register SCON, while the stop bit is ignored. The baud rate is programmable to either 1/32 or 1/64 the oscillator frequency in 12-clock mode or 1/16 or 1/32 the oscillator frequency in 6-clock mode.

**Mode 3:** 11 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). In fact, Mode 3 is the same as Mode 2 in all respects except baud rate. The baud rate in Mode 3 is variable.

In all four modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in Mode 0 by the condition RI = 0 and REN = 1. Reception is initiated in the other modes by the incoming start bit if REN = 1.

**Multiprocessor Communications**

Modes 2 and 3 have a special provision for multiprocessor communications. In these modes, 9 data bits are received. The 9th one goes into RB8. Then comes a stop bit. The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1. This feature is enabled by setting bit SM2 in SCON. A way to use this feature in multiprocessor systems is as follows:

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte which identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SM2 = 1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SM2 bit and prepare to receive the data bytes that will be coming. The slaves that weren't being addressed leave their SM2s set and go on about their business, ignoring the coming data bytes.

SM2 has no effect in Mode 0, and in Mode 1 can be used to check the validity of the stop bit. In a Mode 1 reception, if SM2 = 1, the receive interrupt will not be activated unless a valid stop bit is received.

**Serial Port Control Register**

The serial port control and status register is the Special Function Register SCON, shown in Figure 12. This register contains not only the mode selection bits, but also the 9th data bit for transmit and receive (TB8 and RB8), and the serial port interrupt bits (TI and RI).

**Baud Rates**

The baud rate in Mode 0 is fixed: Mode 0 Baud Rate = Oscillator Frequency / 12 (12-clock mode) or / 6 (6-clock mode). The baud rate in Mode 2 depends on the value of bit SMOD in Special Function Register PCON. If SMOD = 0 (which is the value on reset), and the port pins in 12-clock mode, the baud rate is 1/64 the oscillator frequency. If SMOD = 1, the baud rate is 1/32 the oscillator frequency. In 6-clock mode, the baud rate is 1/32 or 1/16 the oscillator frequency, respectively.

Mode 2 Baud Rate =

$$\frac{2^{SMOD}}{n} \times (\text{Oscillator Frequency})$$

Where:

n = 64 in 12-clock mode, 32 in 6-clock mode

The baud rates in Modes 1 and 3 are determined by the Timer 1 or Timer 2 overflow rate.

**Using Timer 1 to Generate Baud Rates**

When Timer 1 is used as the baud rate generator (T2CON.RCLK = 0, T2CON.TCLK = 0), the baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate and the value of SMOD as follows:

Mode 1, 3 Baud Rate =

$$\frac{2^{SMOD}}{n} \times (\text{Timer 1 Overflow Rate})$$

Where:

n = 32 in 12-clock mode, 16 in 6-clock mode

The Timer 1 interrupt should be disabled in this application. The Timer itself can be configured for either "timer" or "counter" operation, and in any of its 3 running modes. In the most typical applications, it is configured for "timer" operation, in the auto-reload mode (high nibble of TMOD = 0010B). In that case the baud rate is given by the formula:

Mode 1, 3 Baud Rate =

$$\frac{2^{SMOD}}{n} \times \frac{\text{Oscillator Frequency}}{12 \times [256-(TH1)]}$$

Where:

n = 32 in 12-clock mode, 16 in 6-clock mode

One can achieve very low baud rates with Timer 1 by leaving the Timer 1 interrupt enabled, and configuring the Timer to run as a 16-bit timer (high nibble of TMOD = 0001B), and using the Timer 1 interrupt to do a 16-bit software reload. Figure 13 lists various commonly used baud rates and how they can be obtained from Timer 1.

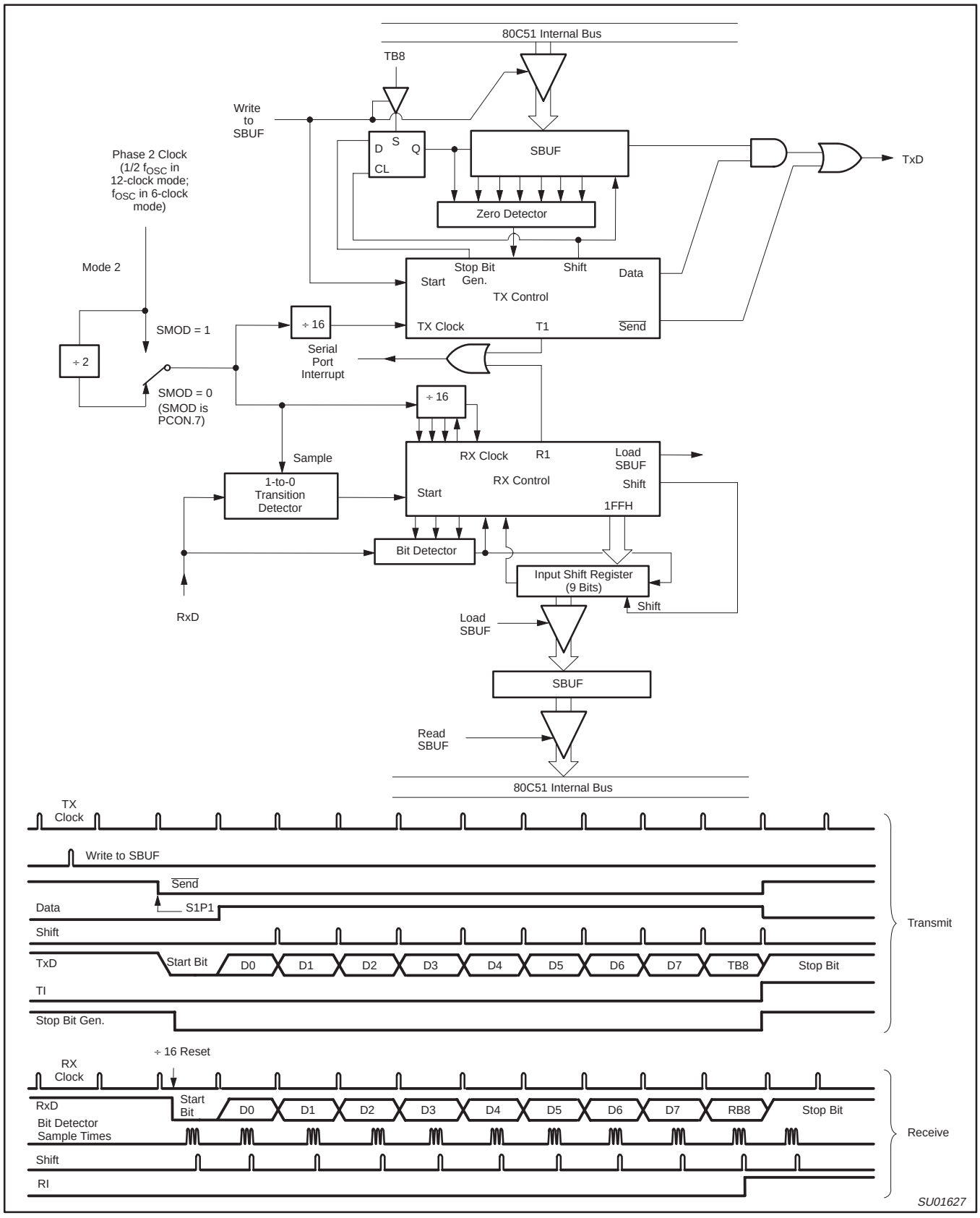



**Figure 16. Serial Port Mode 2**

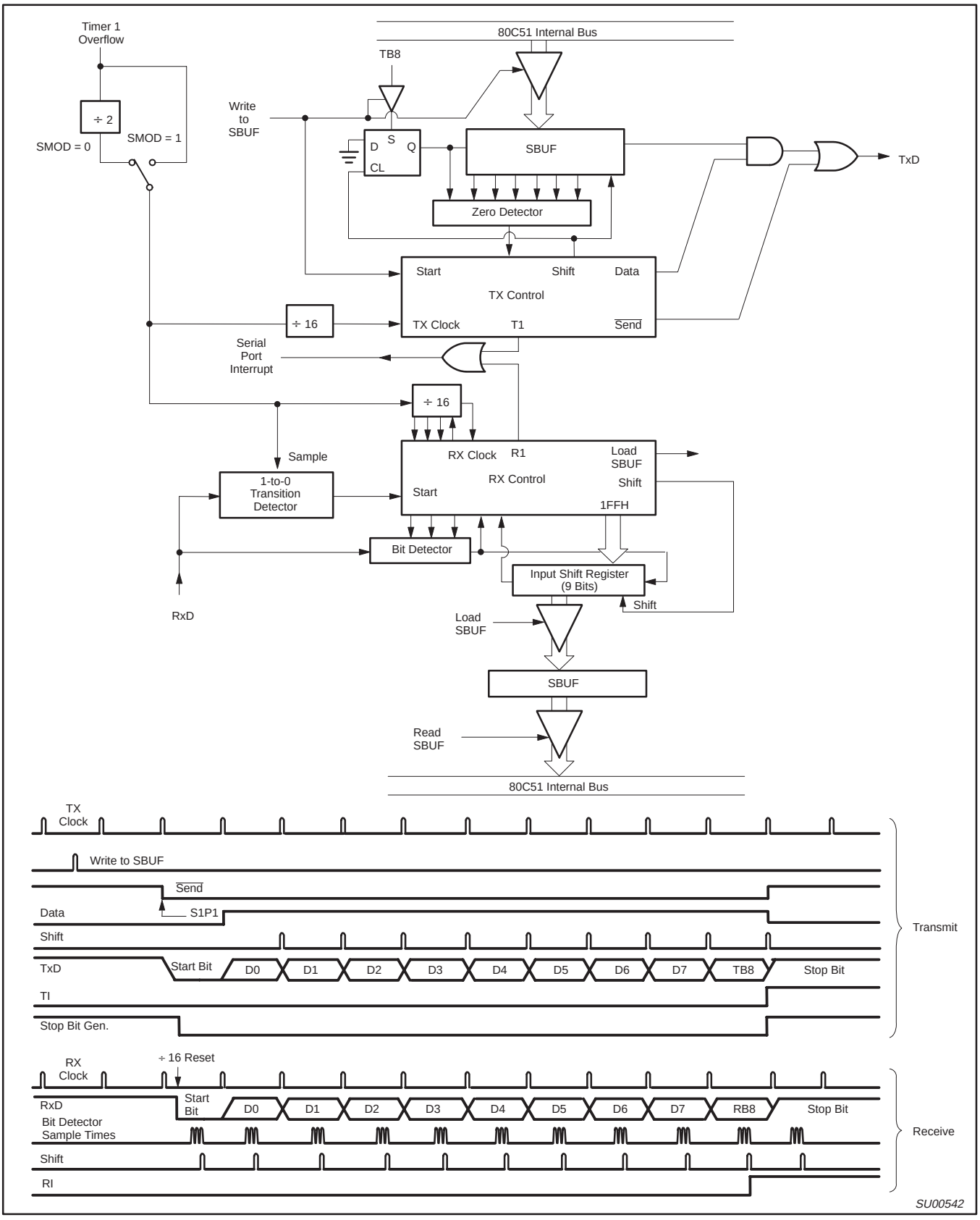



**Figure 17. Serial Port Mode 3**

**SCON** Address = 98H

Bit Addressable

Reset Value = 0000 0000B

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SM0/FE | SM1 | SM2 | REN | TB8 | RB8 | TI | RI |

(SMOD0 = 0/1)*

| Symbol | Position | Function |
|---|---|---|
| FE | SCON.7 | Framing Error bit. This bit is set by the receiver when an invalid stop bit is detected. The FE bit is not cleared by valid frames but should be cleared by software. The SMOD0 bit must be set to enable access to the FE bit.* |
| SM0 | SCON.7 | Serial Port Mode Bit 0, (SMOD0 must = 0 to access bit SM0) |
| SM1 | SCON.6 | Serial Port Mode Bit 1 |

| SM0 | SM1 | Mode | Description | Baud Rate** |
|---|---|---|---|---|
| 0 | 0 | 0 | shift register | $f_{OSC}$/12 (12-clk mode) or $f_{OSC}$/6 (6-clk mode) |
| 0 | 1 | 1 | 8-bit UART | variable |
| 1 | 0 | 2 | 9-bit UART | $f_{OSC}$/64 or $f_{OSC}$/32 or $f_{OSC}$/16 (6-clock mode) or $f_{OSC}$/32 (12-clock mode) |
| 1 | 1 | 3 | 9-bit UART | variable |

| | | |
|---|---|---|
| SM2 | SCON.5 | Enables the Automatic Address Recognition feature in Modes 2 or 3. If SM2 = 1 then RI will not be set unless the received 9th data bit (RB8) is 1, indicating an address, and the received byte is a Given or Broadcast Address. In Mode 1, if SM2 = 1 then RI will not be activated unless a valid stop bit was received, and the received byte is a Given or Broadcast Address. In Mode 0, SM2 should be 0. |
| REN | SCON.4 | Enables serial reception. Set by software to enable reception. Clear by software to disable reception. |
| TB8 | SCON.3 | The 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software as desired. |
| RB8 | SCON.2 | In modes 2 and 3, the 9th data bit that was received. In Mode 1, if SM2 = 0, RB8 is the stop bit that was received.<br>In Mode 0, RB8 is not used. |
| TI | SCON.1 | Transmit interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. Must be cleared by software. |
| RI | SCON.0 | Receive interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM2). Must be cleared by software. |

**NOTES:**
*SMOD0 is located at PCON.6.
**$f_{OSC}$ = oscillator frequency

*SU01628*

**Figure 18. SCON: Serial Port Control Register**

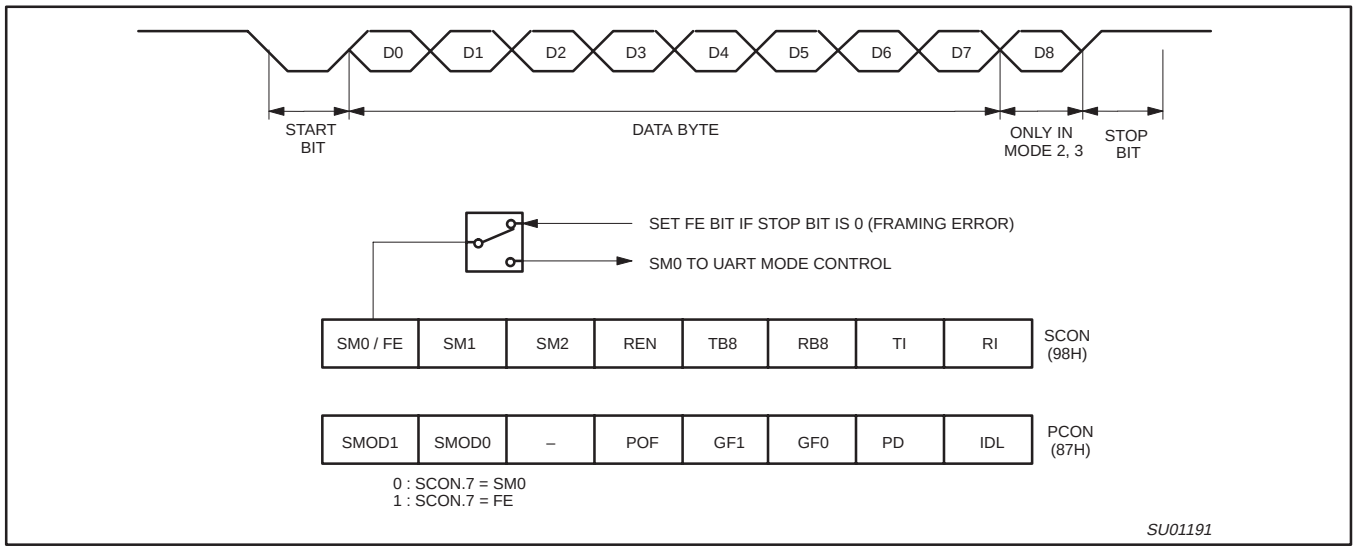



**Figure 19. UART Framing Error Detection**

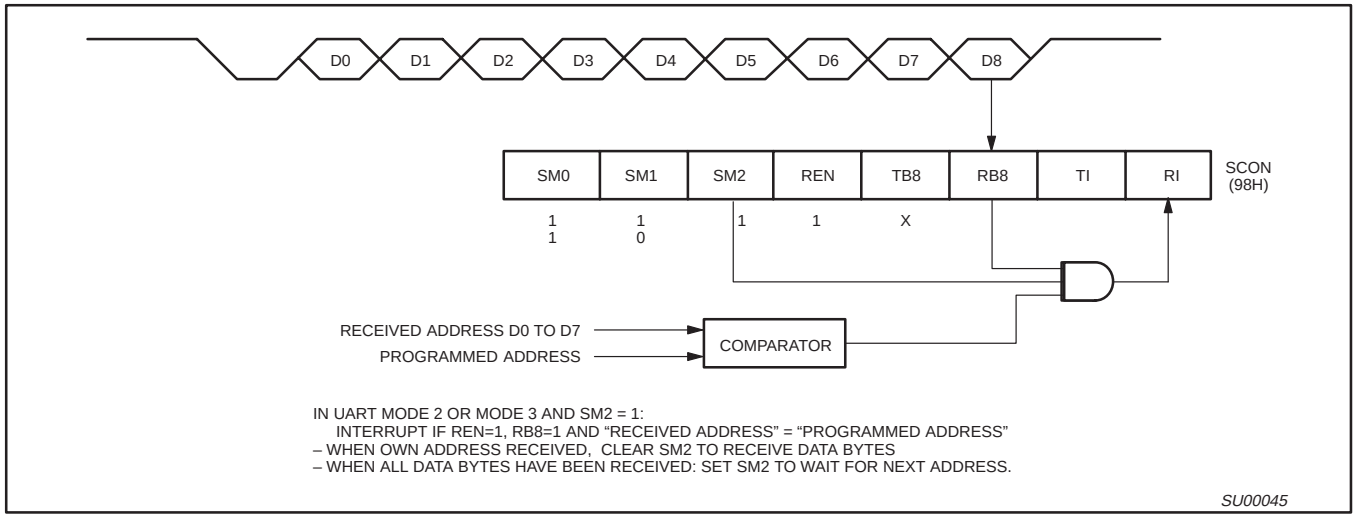



**Figure 20. UART Multiprocessor Communication, Automatic Address Recognition**

**IE** Address = 0A8H Reset Value = 0X000000B

Bit Addressable

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EA | — | ET2 | ES | ET1 | EX1 | ET0 | EX0 |

Enable Bit = 1 enables the interrupt.
Enable Bit = 0 disables it.

| BIT | SYMBOL | FUNCTION |
|---|---|---|
| IE.7 | EA | Global disable bit. If EA = 0, all interrupts are disabled. If EA = 1, each interrupt can be individually enabled or disabled by setting or clearing its enable bit. |
| IE.6 | — | Not implemented. Reserved for future use. |
| IE.5 | ET2 | Timer 2 interrupt enable bit. |
| IE.4 | ES | Serial Port interrupt enable bit. |
| IE.3 | ET1 | Timer 1 interrupt enable bit. |
| IE.2 | EX1 | External interrupt 1 enable bit. |
| IE.1 | ET0 | Timer 0 interrupt enable bit. |
| IE.0 | EX0 | External interrupt 0 enable bit. |

SU01522

**Figure 22. Interrupt Enable (IE) Register**

**IP** Address = 0B8H Reset Value = xx000000B

Bit Addressable

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| — | — | PT2 | PS | PT1 | PX1 | PT0 | PX0 |

Priority Bit = 1 assigns higher priority
Priority Bit = 0 assigns lower priority

| BIT | SYMBOL | FUNCTION |
|---|---|---|
| IP.7 | — | Not implemented, reserved for future use. |
| IP.6 | — | Not implemented, reserved for future use. |
| IP.5 | PT2 | Timer 2 interrupt priority bit. |
| IP.4 | PS | Serial Port interrupt priority bit. |
| IP.3 | PT1 | Timer 1 interrupt priority bit. |
| IP.2 | PX1 | External interrupt 1 priority bit. |
| IP.1 | PT0 | Timer 0 interrupt priority bit. |
| IP.0 | PX0 | External interrupt 0 priority bit. |

SU01523

**Figure 23. Interrupt Priority (IP) Register**

**IPH** Address = B7H Reset Value = xx000000B

Bit Addressable

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| — | — | PT2H | PSH | PT1H | PX1H | PT0H | PX0H |

Priority Bit = 1 assigns higher priority
Priority Bit = 0 assigns lower priority

| BIT | SYMBOL | FUNCTION |
|---|---|---|
| IPH.7 | — | Not implemented, reserved for future use. |
| IPH.6 | — | Not implemented, reserved for future use. |
| IPH.5 | PT2H | Timer 2 interrupt priority bit high. |
| IPH.4 | PSH | Serial Port interrupt priority bit high. |
| IPH.3 | PT1H | Timer 1 interrupt priority bit high. |
| IPH.2 | PX1H | External interrupt 1 priority bit high. |
| IPH.1 | PT0H | Timer 0 interrupt priority bit high. |
| IPH.0 | PX0H | External interrupt 0 priority bit high. |

SU01524

**Figure 24. Interrupt Priority HIGH (IPH) Register**

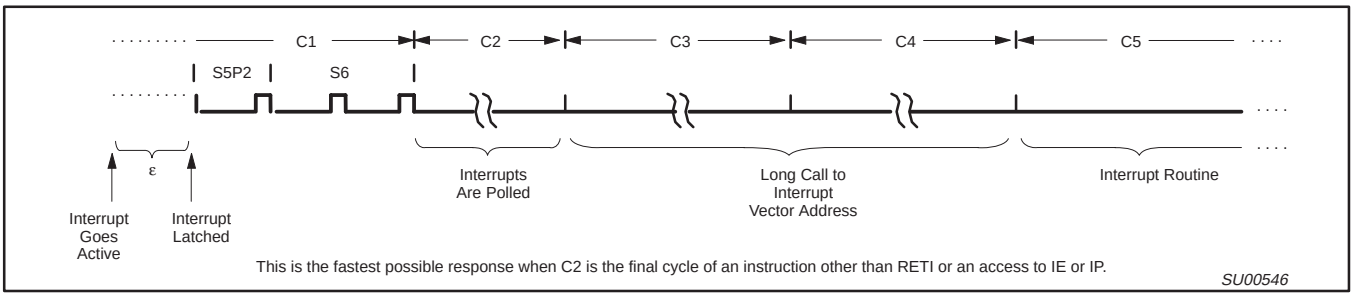



**Figure 25. Interrupt Response Timing Diagram**

The polling cycle/LCALL sequence is illustrated in Figure 25.

Note that if an interrupt of higher priority level goes active prior to S5P2 of the machine cycle labeled C3 in Figure 25, then in accordance with the above rules it will be vectored to during C5 and C6, without any instruction of the lower priority routine having been executed.

Thus the processor acknowledges an interrupt request by executing a hardware-generated LCALL to the appropriate servicing routine. In some cases it also clears the flag that generated the interrupt, and in other cases it doesn't. It never clears the Serial Port flag. This has to be done in the user's software. It clears an external interrupt flag (IE0 or IE1) only if it was transition-activated. The hardware-generated LCALL pushes the contents of the Program Counter on to the stack (but it does not save the PSW) and reloads the PC with an address that depends on the source of the interrupt being vectored to, as shown in Table 8.

Execution proceeds from that location until the RETI instruction is encountered. The RETI instruction informs the processor that this interrupt routine is no longer in progress, then pops the top two bytes from the stack and reloads the Program Counter. Execution of the interrupted program continues from where it left off.

Note that a simple RET instruction would also have returned execution to the interrupted program, but it would have left the interrupt control system thinking an interrupt was still in progress, making future interrupts impossible.

**External Interrupts**

The external sources can be programmed to be level-activated or transition-activated by setting or clearing bit IT1 or IT0 in Register TCON. If ITx = 0, external interrupt x is triggered by a detected low at the $\overline{\text{INT}}$x pin. If ITx = 1, external interrupt x is edge triggered. In this mode if successive samples of the $\overline{\text{INT}}$x pin show a high in one cycle and a low in the next cycle, interrupt request flag IEx in TCON is set. Flag bit IEx then requests the interrupt.

Since the external interrupt pins are sampled once each machine cycle, an input high or low should hold for at least 12 oscillator periods to ensure sampling. If the external interrupt is transition-activated, the external source has to hold the request pin high for at least one cycle, and then hold it low for at least one cycle. This is done to ensure that the transition is seen so that interrupt request flag IEx will be set. IEx will be automatically cleared by the CPU when the service routine is called.

If the external interrupt is level-activated, the external source has to hold the request active until the requested interrupt is actually generated. Then it has to deactivate the request before the interrupt service routine is completed, or else another interrupt will be generated.

**Response Time**

The $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ levels are inverted and latched into IE0 and IE1 at S5P2 of every machine cycle. The values are not actually polled by the circuitry until the next machine cycle. If a request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The call itself takes two cycles. Thus, a minimum of three complete machine cycles elapse between activation of an external interrupt request and the beginning of execution of the first instruction of the service routine. Figure 25 shows interrupt response timings.

A longer response time would result if the request is blocked by one of the 3 previously listed conditions. If an interrupt of equal or higher priority level is already in progress, the additional wait time obviously depends on the nature of the other interrupt's service routine. If the instruction in progress is not in its final cycle, the additional wait time cannot be more the 3 cycles, since the longest instructions (MUL and DIV) are only 4 cycles long, and if the instruction in progress is RETI or an access to IE or IP, the additional wait time cannot be more than 5 cycles (a maximum of one more cycle to complete the instruction in progress, plus 4 cycles to complete the next instruction if the instruction is MUL or DIV).

Thus, in a single-interrupt system, the response time is always more than 3 cycles and less than 9 cycles.

As previously mentioned, the derivatives described in this data sheet have a four-level interrupt structure. The corresponding registers are IE, IP and IPH. (See Figures 22, 23, and 24.) The IPH (Interrupt Priority High) register makes the four-level interrupt structure possible.

The function of the IPH SFR is simple and when combined with the IP SFR determines the priority of each interrupt. The priority of each interrupt is determined as shown in the following table:

| PRIORITY BITS | | INTERRUPT PRIORITY LEVEL |
|---|---|---|
| IPH.x | IP.x | |
| 0 | 0 | Level 0 (lowest priority) |
| 0 | 1 | Level 1 |
| 1 | 0 | Level 2 |
| 1 | 1 | Level 3 (highest priority) |

## AC ELECTRICAL CHARACTERISTICS (6-CLOCK MODE, 2.7 V TO 5.5 V OPERATION)

$T_{amb}$ = 0 °C to +70 °C or –40 °C to +85 °C ; $V_{CC}$=2.7 V to 5.5 V, $V_{SS}$ = 0 V[1,2,3,4,5]

| Symbol | Figure | Parameter | Limits | | 16 MHz Clock | | Unit |
|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | |
| $1/t_{CLCL}$ | 31 | Oscillator frequency | 0 | 16 | – | – | MHz |
| $t_{LHLL}$ | 27 | ALE pulse width | $t_{CLCL}$–10 | – | 52.5 | – | ns |
| $t_{AVLL}$ | 27 | Address valid to ALE low | 0.5 $t_{CLCL}$ –15 | – | 16.25 | – | ns |
| $t_{LLAX}$ | 27 | Address hold after ALE low | 0.5 $t_{CLCL}$ –25 | – | 6.25 | – | ns |
| $t_{LLIV}$ | 27 | ALE low to valid instruction in | – | 2 $t_{CLCL}$ –55 | – | 70 | ns |
| $t_{LLPL}$ | 27 | ALE low to $\overline{PSEN}$ low | 0.5 $t_{CLCL}$ –15 | – | 16.25 | – | ns |
| $t_{PLPH}$ | 27 | $\overline{PSEN}$ pulse width | 1.5 $t_{CLCL}$ –15 | – | 78.75 | – | ns |
| $t_{PLIV}$ | 27 | $\overline{PSEN}$ low to valid instruction in | – | 1.5 $t_{CLCL}$ –55 | – | 38.75 | ns |
| $t_{PXIX}$ | 27 | Input instruction hold after $\overline{PSEN}$ | 0 | – | 0 | – | ns |
| $t_{PXIZ}$ | 27 | Input instruction float after $\overline{PSEN}$ | – | 0.5 $t_{CLCL}$ –10 | – | 21.25 | ns |
| $t_{AVIV}$ | 27 | Address to valid instruction in | – | 2.5 $t_{CLCL}$ –50 | – | 101.25 | ns |
| $t_{PLAZ}$ | 27 | $\overline{PSEN}$ low to address float | – | 10 | – | 10 | ns |
| **Data Memory** | | | | | | | |
| $t_{RLRH}$ | 28 | $\overline{RD}$ pulse width | 3 $t_{CLCL}$ –25 | – | 162.5 | – | ns |
| $t_{WLWH}$ | 29 | $\overline{WR}$ pulse width | 3 $t_{CLCL}$ –25 | – | 162.5 | – | ns |
| $t_{RLDV}$ | 28 | $\overline{RD}$ low to valid data in | – | 2.5 $t_{CLCL}$ –50 | – | 106.25 | ns |
| $t_{RHDX}$ | 28 | Data hold after $\overline{RD}$ | 0 | – | 0 | – | ns |
| $t_{RHDZ}$ | 28 | Data float after $\overline{RD}$ | – | $t_{CLCL}$ –20 | – | 42.5 | ns |
| $t_{LLDV}$ | 28 | ALE low to valid data in | – | 4 $t_{CLCL}$ –55 | – | 195 | ns |
| $t_{AVDV}$ | 28 | Address to valid data in | – | 4.5 $t_{CLCL}$ –50 | – | 231.25 | ns |
| $t_{LLWL}$ | 28, 29 | ALE low to $\overline{RD}$ or $\overline{WR}$ low | 1.5 $t_{CLCL}$ –20 | 1.5 $t_{CLCL}$ +20 | 73.75 | 113.75 | ns |
| $t_{AVWL}$ | 28, 29 | Address valid to $\overline{WR}$ low or $\overline{RD}$ low | 2 $t_{CLCL}$ –20 | – | 105 | – | ns |
| $t_{QVWX}$ | 29 | Data valid to $\overline{WR}$ transition | 0.5 $t_{CLCL}$ –30 | – | 1.25 | – | ns |
| $t_{WHQX}$ | 29 | Data hold after $\overline{WR}$ | 0.5 $t_{CLCL}$ –20 | – | 11.25 | – | ns |
| $t_{QVWH}$ | 29 | Data valid to $\overline{WR}$ high | 3.5 $t_{CLCL}$ –10 | – | 208.75 | – | ns |
| $t_{RLAZ}$ | 28 | $\overline{RD}$ low to address float | – | 0 | – | 0 | ns |
| $t_{WHLH}$ | 28, 29 | $\overline{RD}$ or $\overline{WR}$ high to ALE high | 0.5 $t_{CLCL}$ –15 | 0.5 $t_{CLCL}$ +15 | 16.25 | 46.25 | ns |
| **External Clock** | | | | | | | |
| $t_{CHCX}$ | 31 | High time | 0.4 $t_{CLCL}$ | $t_{CLCL} - t_{CLCX}$ | – | – | ns |
| $t_{CLCX}$ | 31 | Low time | 0.4 $t_{CLCL}$ | $t_{CLCL} - t_{CHCX}$ | – | – | ns |
| $t_{CLCH}$ | 31 | Rise time | – | 5 | – | – | ns |
| $t_{CHCL}$ | 31 | Fall time | – | 5 | – | – | ns |
| **Shift register** | | | | | | | |
| $t_{XLXL}$ | 30 | Serial port clock cycle time | 6 $t_{CLCL}$ | – | 375 | – | ns |
| $t_{QVXH}$ | 30 | Output data setup to clock rising edge | 5 $t_{CLCL}$ –25 | – | 287.5 | – | ns |
| $t_{XHQX}$ | 30 | Output data hold after clock rising edge | $t_{CLCL}$ –15 | – | 47.5 | – | ns |
| $t_{XHDX}$ | 30 | Input data hold after clock rising edge | 0 | – | 0 | – | ns |
| $t_{XHDV}$ | 30 | Clock rising edge to input data valid | – | 5 $t_{CLCL}$ –133 | – | 179.5 | ns |

**NOTES:**
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$=100 pF, load capacitance for all outputs = 80 pF
3. Interfacing the microcontroller to devices with float time up to 45ns is permitted. This limited bus contention will not cause damage to port 0 drivers.
4. Parts are guaranteed by design to operate down to 0 Hz.
5. Data shown in the table are the best mathematical models for the set of measured values obtained in tests. If a particular parameter calculated at a customer specified frequency has a negative value, it should be considered equal to zero.

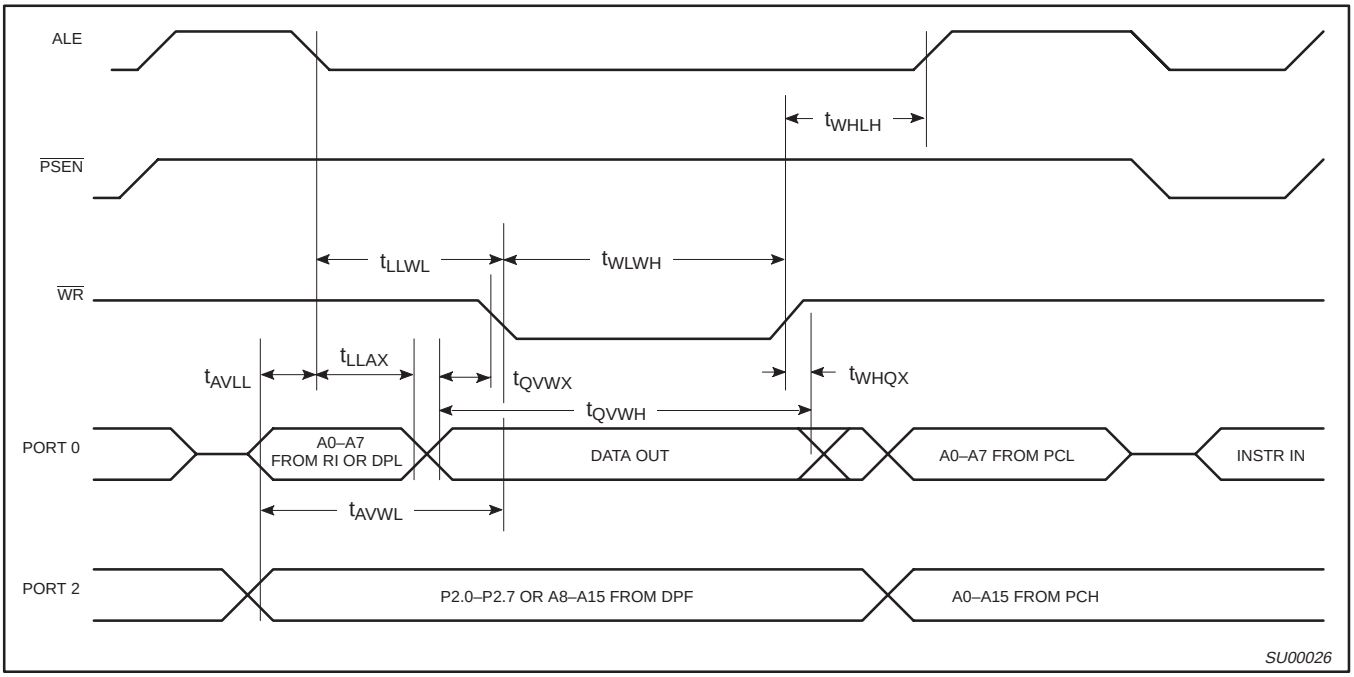



**Figure 29. External Data Memory Write Cycle**

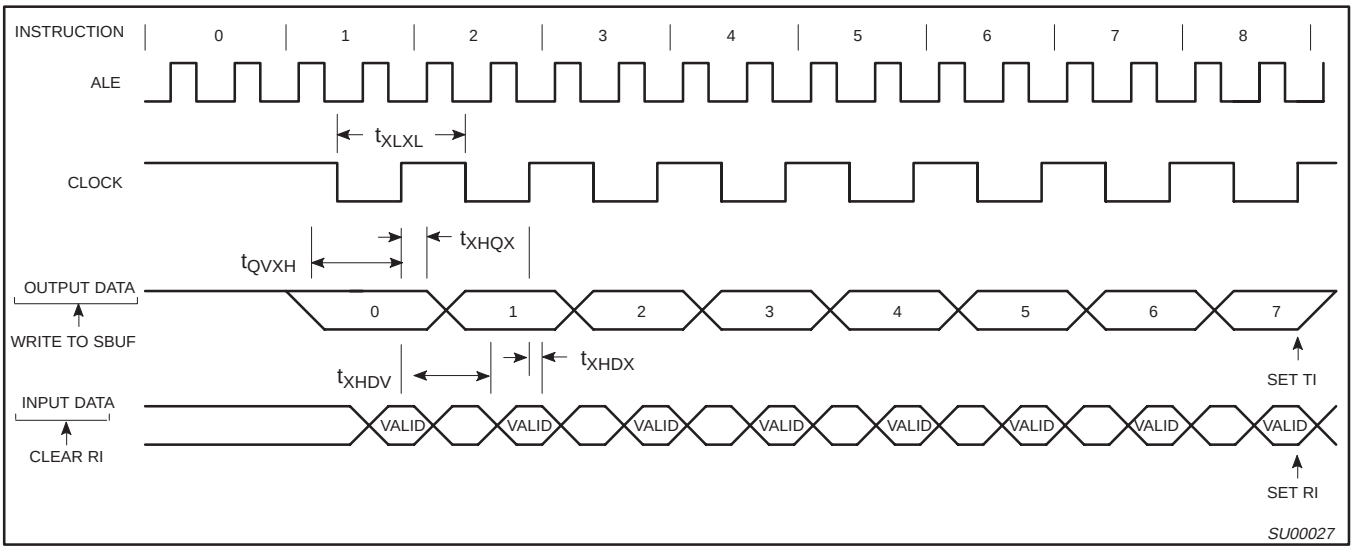



**Figure 30. Shift Register Mode Timing**

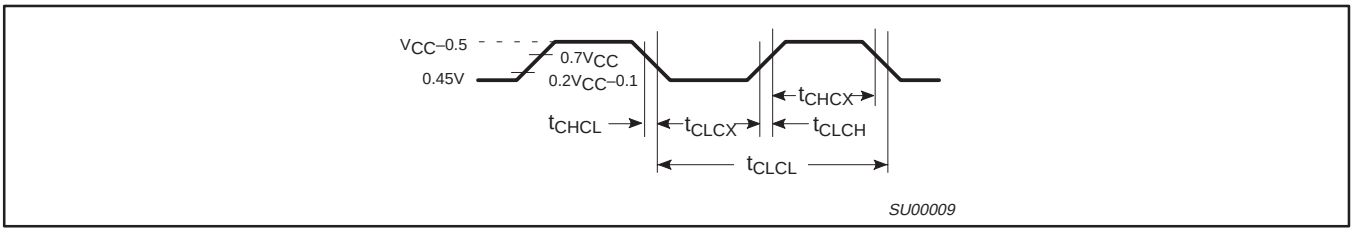



**Figure 31. External Clock Drive**

```
/*
##      as31 version V2.10           / *js* /
##
##
##      source file:  idd_ljmp1.asm
##        list file:  idd_ljmp1.lst   created Fri Apr 20 15:51:40 2001
##
#############################################################
#0000                # AUXR  equ 08Eh
#0000                # CKCON equ 08Fh
                     #
                     #
#0000                # org 0
                     #
                     # LJMP_LABEL:
0000 /75;/8E;/01;    #          MOV     AUXR,#001h   ; turn off ALE
0003 /02;/FF;/FD;    #          LJMP    LJMP_LABEL   ; jump to end of address space
0005 /00;            #          NOP
                     #
#FFFD                # org 0fffdh
                     #
                     # LJMP_LABEL:
                     #
FFFD /02;/FD;FF;     #          LJMP LJMP_LABEL
                     # ;        NOP
                     #
                     #
*/"
```

SU01499

**Figure 35. Source code used in measuring $I_{DD}$ operational**

**Table 9. EPROM Programming Modes**

| MODE | RST | $\overline{PSEN}$ | ALE/$\overline{PROG}$ | $\overline{EA}$/$V_{PP}$ | P2.7 | P2.6 | P3.7 | P3.6 | P3.3 |
|---|---|---|---|---|---|---|---|---|---|
| Read signature | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | X |
| Program code data | 1 | 0 | 0* | $V_{PP}$ | 1 | 0 | 1 | 1 | X |
| Verify code data | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | X |
| Pgm encryption table | 1 | 0 | 0* | $V_{PP}$ | 1 | 0 | 1 | 0 | X |
| Pgm security bit 1 | 1 | 0 | 0* | $V_{PP}$ | 1 | 1 | 1 | 1 | X |
| Pgm security bit 2 | 1 | 0 | 0* | $V_{PP}$ | 1 | 1 | 0 | 0 | X |
| Pgm security bit 3 | 1 | 0 | 0* | $V_{PP}$ | 0 | 1 | 0 | 1 | X |
| Program to 6-clock mode | 1 | 0 | 0* | $V_{PP}$ | 0 | 0 | 1 | 0 | 0 |
| Verify 6-clock[4] | 1 | 0 | 1 | 1 | e | 0 | 0 | 1 | 1 |
| Verify security bits[5] | 1 | 0 | 1 | 1 | e | 0 | 1 | 0 | X |

**NOTES:**
1. '0' = Valid low for that pin, '1' = valid high for that pin.
2. $V_{PP}$ = 12.75 V ±0.25 V.
3. $V_{CC}$ = 5 V±10% during programming and verification.
4. Bit is output on P0.4 (1 = 12x, 0 = 6x).
5. Security bit one is output on P0.7.
   Security bit two is output on P0.6.
   Security bit three is output on P0.3.

* ALE/$\overline{PROG}$ receives 5 programming pulses for code data (also for user array; 5 pulses for encryption or security bits) while $V_{PP}$ is held at 12.75 V. Each programming pulse is low for 100 µs (±10 µs) and high for a minimum of 10 µs.

**Table 10. Program Security Bits for EPROM Devices**

| PROGRAM LOCK BITS[1, 2] | | | | PROTECTION DESCRIPTION |
|---|---|---|---|---|
| | SB1 | SB2 | SB3 | |
| 1 | U | U | U | No Program Security features enabled. (Code verify will still be encrypted by the Encryption Array if programmed.) |
| 2 | P | U | U | MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory, $\overline{EA}$ is sampled and latched on Reset, and further programming of the EPROM is disabled. |
| 3 | P | P | U | Same as 2, also verify is disabled. |
| 4 | P | P | P | Same as 3, external execution is disabled. Internal data RAM is not accessible. |

**NOTES:**
1. P – programmed. U – unprogrammed.
2. Any other combination of the security bits is not defined.

## MASK ROM DEVICES

### Security Bits

With none of the security bits programmed the code in the program memory can be verified. If the encryption table is programmed, the code will be encrypted when verified. When only security bit 1 (see Table 11) is programmed, MOVC instructions executed from external program memory are disabled from fetching code bytes from the internal memory, $\overline{EA}$ is latched on Reset and all further programming of the EPROM is disabled. When security bits 1 and 2 are programmed, in addition to the above, verify mode is disabled.

### Encryption Array

64 bytes (87C51), or 32 bytes (87C52/4) of encryption array are initially unprogrammed (all 1s).

### Table 11. Program Security Bits

| PROGRAM LOCK BITS[1, 2] | | | |
|---|---|---|---|
| | SB1 | SB2 | PROTECTION DESCRIPTION |
| 1 | U | U | No Program Security features enabled. (Code verify will still be encrypted by the Encryption Array if programmed.) |
| 2 | P | U | MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory, $\overline{EA}$ is sampled and latched on Reset, and further programming of the EPROM is disabled. |

**NOTES:**
1. P – programmed. U – unprogrammed.
2. Any other combination of the security bits is not defined.

## 80C51X2 ROM CODE SUBMISSION

When submitting a ROM code for the 80C51X2, the following must be specified:
1. 4 kbyte user ROM data
2. 64 byte ROM encryption key
3. ROM security bits.

| ADDRESS | CONTENT | BIT(S) | COMMENT |
|---|---|---|---|
| 0000H to 0FFFH | DATA | 7:0 | User ROM Data |
| 1000H to 103FH | KEY | 7:0 | ROM Encryption Key |
| 1040H | SEC | 0 | ROM Security Bit 1 |
| 1040H | SEC | 1 | ROM Security Bit 2 |

**Security Bit 1:** When programmed, this bit has two effects on masked ROM parts:
1. External MOVC is disabled, and
2. $\overline{EA}$ is latched on Reset.

**Security Bit 2:** When programmed, this bit inhibits Verify User ROM.

**NOTE:** Security Bit 2 cannot be enabled unless Security Bit 1 is enabled.

If the ROM Code file does not include the options, the following information must be included with the ROM code.

For each of the following, check the appropriate box, and send to Philips along with the code:

Security Bit #1: ☐ Enabled ☐ Disabled

Security Bit #2: ☐ Enabled ☐ Disabled

Encryption: ☐ No ☐ Yes If Yes, must send key file.

## 80C52X2 ROM CODE SUBMISSION

When submitting a ROM code for the 80C52X2, the following must be specified:
1. 8 kbyte user ROM data
2. 64 byte ROM encryption key
3. ROM security bits.

| ADDRESS | CONTENT | BIT(S) | COMMENT |
|---|---|---|---|
| 0000H to 1FFFH | DATA | 7:0 | User ROM Data |
| 2000H to 203FH | KEY | 7:0 | ROM Encryption Key |
| 2040H | SEC | 0 | ROM Security Bit 1 |
| 2040H | SEC | 1 | ROM Security Bit 2 |

**Security Bit 1:** When programmed, this bit has two effects on masked ROM parts:
1. External MOVC is disabled, and
2. $\overline{EA}$ is latched on Reset.

**Security Bit 2:** When programmed, this bit inhibits Verify User ROM.

**NOTE:** Security Bit 2 cannot be enabled unless Security Bit 1 is enabled.

If the ROM Code file does not include the options, the following information must be included with the ROM code.

For each of the following, check the appropriate box, and send to Philips along with the code:

Security Bit #1: ☐ Enabled ☐ Disabled

Security Bit #2: ☐ Enabled ☐ Disabled

Encryption: ☐ No ☐ Yes If Yes, must send key file.

## REVISION HISTORY

| Rev | Date | Description |
|---|---|---|
| _6 | 20030124 | **Product data (9397 750 10995); ECN 853-2337 29260 of 06 December 2002**<br>Modifications:<br>• Added TSSOP38 package details |
| _5 | 20020912 | **Product data (9397 750 10361); ECN 853-2337 28906 of 12 September 2002** |
| _4 | 20020612 | **Product data (9397 750 09969); ECN 853-2337 28427 of 12 June 2002** |
| _3 | 20020422 | **Product data (9397 750 09779); ECN 853-2337 28059 of 22 April 2002** |
| _2 | 20020219 | **Preliminary data (9397 750 09467)** |
| _1 | 20010924 | **Preliminary data (9397 750 08895); initial release** |