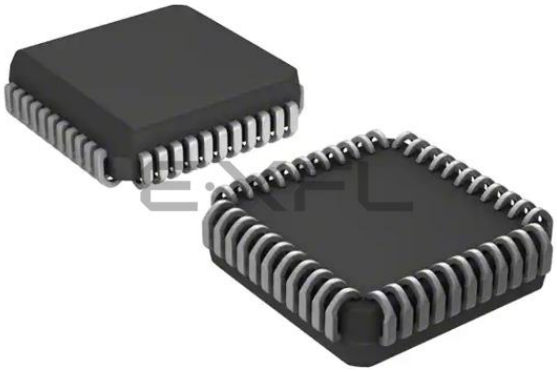


**Welcome to E-XFL.COM**

## 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 | I²C, UART/USART |
| Peripherals | POR, PWM, WDT |
| Number of I/O | 32 |
| Program Memory Size | 64KB (64K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 2K x 8 |
| Voltage - Supply (Vcc/Vdd) | 4.75V ~ 5.25V |
| Data Converters | - |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-LCC (J-Lead) |
| Supplier Device Package | 44-PLCC (16.59x16.59) |
| Purchase URL | https://www.e-xfl.com/product-detail/nxp-semiconductors/p89c664hfa-00-512 |

# 80C51 8-bit Flash microcontroller family
## 16KB/32KB/64KB ISP/IAP Flash with 512B/1KB/2KB/8KB RAM

# P89C660/P89C662/P89C664/ P89C668

## DESCRIPTION

The P89C660/662/664/668 device contains a non-volatile 16KB/32KB/64KB Flash program memory that is both parallel programmable and serial In-System and In-Application Programmable. In-System Programming (ISP) allows the user to download new code while the microcontroller sits in the application. In-Application Programming (IAP) means that the microcontroller fetches new program code and reprograms itself while in the system. This allows for remote programming over a modem link. A default serial loader (boot loader) program in ROM allows serial In-System Programming of the Flash memory via the UART without the need for a loader in the Flash code. For In-Application Programming, the user program erases and reprograms the Flash memory by use of standard routines contained in ROM.

This device executes one instruction in 6 clock cycles, hence providing twice the speed of a conventional 80C51. An OTP configuration bit gives the user the option to select conventional 12-clock timing.

This device is a Single-Chip 8-Bit Microcontroller manufactured in advanced CMOS process and is a derivative of the 80C51 microcontroller family. The instruction set is 100% executing and timing compatible with the 80C51 instruction set.

The device also has four 8-bit I/O ports, three 16-bit timer/event counters, a multi-source, four-priority-level, nested interrupt structure, an enhanced UART and on-chip oscillator and timing circuits.

The added features of the P89C660/662/664/668 makes it a powerful microcontroller for applications that require pulse width modulation, high-speed I/O and up/down counting capabilities such as motor control.

## FEATURES

- 80C51 Central Processing Unit
- On-chip Flash program memory with In-System Programming (ISP) and In-Application Programming (IAP) capability
- Boot ROM contains low level Flash programming routines for downloading via the UART
- Can be programmed by the end-user application (IAP)
- Parallel programming with 87C51 compatible hardware interface to programmer
- Six clocks per machine cycle operation (standard)
- 12 clocks per machine cycle operation (optional)
- Speed up to 20 MHz with 6 clock cycles per machine cycle (40 MHz equivalent performance); up to 33 MHz with 12 clocks per machine cycle
- Fully static operation
- RAM externally expandable to 64 kbytes
- Four interrupt priority levels
- Eight interrupt sources
- Four 8-bit I/O ports
- Full-duplex enhanced UART
  - Framing error detection
  - Automatic address recognition
- Power control modes
  - Clock can be stopped and resumed
  - Idle mode
  - Power-Down mode
- Programmable clock out
- Second DPTR register
- Asynchronous port reset
- Low EMI (inhibit ALE)
- $I^2C$ serial interface
- Programmable Counter Array (PCA)
  - PWM
  - Capture/compare
- Well-suited for IPMI applications

## PIN DESCRIPTIONS

| MNEMONIC | PIN NUMBER | | TYPE | NAME AND FUNCTION |
|---|---|---|---|---|
| | PLCC | LQFP | | |
| $V_{SS}$ | 22 | 16 | I | **Ground:** 0 V reference. |
| $V_{CC}$ | 44 | 38 | I | **Power Supply:** This is the power supply voltage for normal, idle, and power-down operation. |
| P0.0–0.7 | 43–36 | 37–30 | I/O | **Port 0:** Port 0 is an open-drain, bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high-impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory. In this application, it uses strong internal pull-ups when emitting 1s. |
| P1.0–P1.7 | 2–9 | 40–44, 1–3 | I/O | **Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pull-ups on all pins except P1.6 and P1.7 which are open drain. Port 1 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 1 pins that are externally pulled low will source current because of the internal pull-ups. (See DC Electrical Characteristics: $I_{IL}$). |
| | | | | Alternate functions for P89C660/662/664/668 Port 1 include: |
| | 2 | 40 | I/O | **T2 (P1.0):** Timer/Counter 2 external count input/Clockout (see Programmable Clock-Out) |
| | 3 | 41 | I | **T2EX (P1.1):** Timer/Counter 2 Reload/Capture/Direction Control |
| | 4 | 42 | I | **ECI (P1.2):** External Clock Input to the PCA |
| | 5 | 43 | I/O | **CEX0 (P1.3):** Capture/Compare External I/O for PCA module 0 |
| | 6 | 44 | I/O | **CEX1 (P1.4):** Capture/Compare External I/O for PCA module 1 |
| | 7 | 1 | I/O | **CEX2 (P1.5):** Capture/Compare External I/O for PCA module 2 |
| | 8 | 2 | I/O | **SCL (P1.6):** $I^2C$ bus clock line (open drain) |
| | 9 | 3 | I/O | **SDA (P1.7):** $I^2C$ bus data line (open drain) |
| P2.0–P2.7 | 24–31 | 18–25 | I/O | **Port 2:** Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. Port 2 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 2 pins that are externally being pulled low will source current because of the internal pull-ups. (See DC Electrical Characteristics: $I_{IL}$). Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @DPTR). In this application, it uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOV @Ri), port 2 emits the contents of the P2 special function register. |
| P3.0–P3.7 | 11, 13–19 | 5, 7–13 | I/O | **Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. Port 3 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pull-ups. (See DC Electrical Characteristics: $I_{IL}$). Port 3 also serves the special features of the P89C660/662/664/668, as listed below: |
| | 11 | 5 | I | **RxD (P3.0):** Serial input port |
| | 13 | 7 | O | **TxD (P3.1):** Serial output port |
| | 14 | 8 | I | **$\overline{INT0}$ (P3.2):** External interrupt |
| | 15 | 9 | I | **$\overline{INT1}$ (P3.3):** External interrupt |
| | 16 | 10 | I | **CEX3/T0 (P3.4):** Timer 0 external input; Capture/Compare External I/O for PCA module 3 |
| | 17 | 11 | I | **CEX4/T1 (P3.5):** Timer 1 external input; Capture/Compare External I/O for PCA module 4 |
| | 18 | 12 | O | **$\overline{WR}$ (P3.6):** External data memory write strobe |
| | 19 | 13 | O | **$\overline{RD}$ (P3.7):** External data memory read strobe |
| RST | 10 | 4 | I | **Reset:** A high on this pin for two machine cycles while the oscillator is running, resets the device. An internal resistor to $V_{SS}$ permits a power-on reset using only an external capacitor to $V_{CC}$. |
| ALE | 33 | 27 | O | **Address Latch Enable:** Output pulse for latching the low byte of the address during an access to external memory. In normal operation, ALE is emitted twice every machine cycle, and can be used for external timing or clocking. Note that one ALE pulse is skipped during each access to external data memory. ALE can be disabled by setting SFR auxiliary.0. With this bit set, ALE will be active only during a MOVX instruction. |
| $\overline{PSEN}$ | 32 | 26 | O | **Program Store Enable:** The read strobe to external program memory. When executing code from the external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory. $\overline{PSEN}$ is not activated during fetches from internal program memory. |

| MNEMONIC | PIN NUMBER | | TYPE | NAME AND FUNCTION |
|---|---|---|---|---|
| | PLCC | LQFP | | |
| $\overline{EA}/V_{PP}$ | 35 | 29 | I | **External Access Enable/Programming Supply Voltage:** $\overline{EA}$ must be externally held low to enable the device to fetch code from external program memory locations. If $\overline{EA}$ is held high, the device executes from internal program memory. The value on the $\overline{EA}$ pin is latched when RST is released and any subsequent changes have no effect. This pin also receives the programming supply voltage ($V_{PP}$) during Flash programming. |
| XTAL1 | 21 | 15 | I | **Crystal 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits. |
| XTAL2 | 20 | 14 | O | **Crystal 2:** Output from the inverting oscillator amplifier. |

**NOTE:**
To avoid “latch-up” effect at power-on, the voltage on any pin (other than $V_{PP}$) must not be higher than $V_{CC}$ + 0.5 V or less than $V_{SS}$ – 0.5 V.

**Table 1. Special Function Registers**

| SYMBOL | DESCRIPTION | DIRECT ADDRESS | BIT ADDRESS, SYMBOL, OR ALTERNATIVE PORT FUNCTION MSB | | | | | | | LSB | RESET VALUE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ACC* | Accumulator | E0H | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 | 00H |
| AUXR# | Auxiliary | 8EH | – | – | – | – | – | – | EXTRAM | AO | xxxxxx10B |
| AUXR1# | Auxiliary 1 | A2H | – | – | ENBOOT | – | GF2 | 0 | – | DPS | xxxxx0x0B |
| B* | B register | F0H | F7 | F6 | F5 | F4 | F3 | F2 | F1 | F0 | 00H |
| CCAP0H# | Module 0 Capture High | FAH | | | | | | | | | xxxxxxxxB |
| CCAP1H# | Module 1 Capture High | FBH | | | | | | | | | xxxxxxxxB |
| CCAP2H# | Module 2 Capture High | FCH | | | | | | | | | xxxxxxxxB |
| CCAP3H# | Module 3 Capture High | FDH | | | | | | | | | xxxxxxxxB |
| CCAP4H# | Module 4 Capture High | FEH | | | | | | | | | xxxxxxxxB |
| CCAP0L# | Module 0 Capture Low | EAH | | | | | | | | | xxxxxxxxB |
| CCAP1L# | Module 1 Capture Low | EBH | | | | | | | | | xxxxxxxxB |
| CCAP2L# | Module 2 Capture Low | ECH | | | | | | | | | xxxxxxxxB |
| CCAP3L# | Module 3 Capture Low | EDH | | | | | | | | | xxxxxxxxB |
| CCAP4L# | Module 4 Capture Low | EEH | | | | | | | | | xxxxxxxxB |
| CCAPM0# | Module 0 Mode | C2H | – | ECOM | CAPP | CAPN | MAT | TOG | PWM | ECCF | x0000000B |
| CCAPM1# | Module 1 Mode | C3H | – | ECOM | CAPP | CAPN | MAT | TOG | PWM | ECCF | x0000000B |
| CCAPM2# | Module 2 Mode | C4H | – | ECOM | CAPP | CAPN | MAT | TOG | PWM | ECCF | x0000000B |
| CCAPM3# | Module 3 Mode | C5H | – | ECOM | CAPP | CAPN | MAT | TOG | PWM | ECCF | x0000000B |
| CCAPM4# | Module 4 Mode | C6H | – | ECOM | CAPP | CAPN | MAT | TOG | PWM | ECCF | x0000000B |
| | | | C7 | C6 | C5 | C4 | C3 | C2 | C1 | C0 | |
| CCON*# | PCA Counter Control | C0H | CF | CR | – | CCF4 | CCF3 | CCF2 | CCF1 | CCF0 | 00x00000B |
| CH# | PCA Counter High | F9H | | | | | | | | | 00H |
| CL# | PCA Counter Low | E9H | | | | | | | | | 00H |
| CMOD# | PCA Counter Mode | C1H | CIDL | WDTE | – | – | – | CPS1 | CPS0 | ECF | 00xxx000B |
| DPTR: | Data Pointer (2 bytes) | | | | | | | | | | |
| DPH | Data Pointer High | 83H | | | | | | | | | 00H |
| DPL | Data Pointer Low | 82H | | | | | | | | | 00H |
| | | | AF | AE | AD | AC | AB | AA | A9 | A8 | |
| IEN0* | Interrupt Enable 0 | A8H | EA | EC | ES1 | ES0 | ET1 | EX1 | ET0 | EX0 | 00H |
| IEN1* | Interrupt Enable 1 | E8 | – | – | – | – | – | – | – | ET2 | xxxxxxx0B |
| | | | BF | BE | BD | BC | BB | BA | B9 | B8 | |
| IP* | Interrupt Priority | B8H | PT2 | PPC | PS1 | PS0 | PT1 | PX1 | PT0 | PX0 | x0000000B |
| IPH# | Interrupt Priority High | B7H | PT2H | PPCH | PS1H | PS0H | PT1H | PX1H | PT0H | PX0H | x0000000B |
| | | | 87 | 86 | 85 | 84 | 83 | 82 | 81 | 80 | |
| P0* | Port 0 | 80H | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | FFH |
| | | | 97 | 96 | 95 | 94 | 93 | 92 | 91 | 90 | |
| P1* | Port 1 | 90H | SDA | SCL | CEX2 | CEX1 | CEX0 | ECI | T2EX | T2 | FFH |
| | | | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | |
| P2* | Port 2 | A0H | AD15 | AD14 | AD13 | AD12 | AD11 | AD10 | AD9 | AD8 | FFH |
| | | | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 | |
| P3* | Port 3 | B0H | $\overline{RD}$ | $\overline{WR}$ | T1/ CEX4 | T0/ CEX3 | $\overline{INT1}$ | $\overline{INT0}$ | TxD | RxD | FFH |
| | | | | | | | | | | | |
| PCON#[1] | Power Control | 87H | SMOD1 | SMOD0 | – | POF | GF1 | GF0 | PD | IDL | 00xxx000B |

* SFRs are bit addressable.
\# SFRs are modified from or added to the 80C51 SFRs.
– Reserved bits.
1. Reset value depends on reset source.

## I²C SERIAL COMMUNICATION — SIO1

The I²C serial port is identical to the I²C serial port on the 8XC554, 8XC654, and 8XC652 devices.

Note that the P89C660/662/664/668 I²C pins are alternate functions to port pins P1.6 and P1.7. Because of this, P1.6 and P1.7 on these parts do not have a pull-up structure as found on the 80C51. Therefore P1.6 and P1.7 have open drain outputs on the P89C660/662/664/668.

The I²C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of the bus are:

– Bidirectional data transfer between masters and slaves
– Multimaster bus (no central master)
– Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
– Serial clock synchronization allows devices with different bit rates to communicate via one serial bus
– Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer
– The I²C bus may be used for test and diagnostic purposes

The output latches of P1.6 and P1.7 must be set to logic 1 in order to enable SIO1.

The P89C66x on-chip I²C logic provides a serial interface that meets the I²C bus specification and supports all transfer modes (other than the low-speed mode) from and to the I²C bus. The SIO1 logic handles bytes transfer autonomously. It also keeps track of serial transfers, and a status register (S1STA) reflects the status of SIO1 and the I²C bus.

The CPU interfaces to the I²C logic via the following four special function registers: S1CON (SIO1 control register), S1STA (SIO1 status register), S1DAT (SIO1 data register), and S1ADR (SIO1 slave address register). The SIO1 logic interfaces to the external I²C bus via two port 1 pins: P1.6/SCL (serial clock line) and P1.7/SDA (serial data line).

A typical I²C bus configuration is shown in Figure 1. Figure 2 shows how a data transfer is accomplished on the bus. Depending on the state of the direction bit (R/W), two types of data transfers are possible on the I²C bus:

1. Data transfer from a master transmitter to a slave receiver. The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte.
2. Data transfer from a slave transmitter to a master receiver. The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. Next follows the data bytes transmitted by the slave to the master. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a “not acknowledge” is returned.

The master device generates all of the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a repeated START condition. Since a repeated START condition is also the beginning of the next serial transfer, the I²C bus will not be released.

### Modes of Operation

The on-chip SIO1 logic may operate in the following four modes:

1. Master Transmitter mode:

   Serial data output through P1.7/SDA while P1.6/SCL outputs the serial clock. The first transmitted byte contains the slave address of the receiving device (7 bits) and the data direction bit. In this mode the data direction bit (R/$\overline{W}$) will be logic 0, and we say that a “W” is transmitted. Thus the first byte transmitted is SLA+W. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an acknowledge bit is received. START and STOP conditions are output to indicate the beginning and the end of a serial transfer.

2. Master Receiver Mode:

   The first transmitted byte contains the slave address of the transmitting device (7 bits) and the data direction bit. In this mode the data direction bit (R/$\overline{W}$) will be logic 1, and we say that an “R” is transmitted. Thus the first byte transmitted is SLA+R. Serial data is received via P1.7/SDA while P1.6/SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are output to indicate the beginning and end of a serial transfer.

3. Slave Receiver mode:

   Serial data and the serial clock are received through P1.7/SDA and P1.6/SCL. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and direction bit.

4. Slave Transmitter mode:

   The first byte is received and handled as in the Slave Receiver mode. However, in this mode, the direction bit will indicate that the transfer direction is reversed. Serial data is transmitted via P1.7/SDA while the serial clock is input through P1.6/SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer.

In a given application, SIO1 may operate as a master and as a slave. In the Slave mode, the SIO1 hardware looks for its own slave address and the general call address. If one of these addresses is detected, an interrupt is requested. When the microcontroller wishes to become the bus master, the hardware waits until the bus is free before the Master mode is entered so that a possible slave action is not interrupted. If bus arbitration is lost in the Master mode, SIO1 switches to the Slave mode immediately and can detect its own slave address in the same serial transfer.

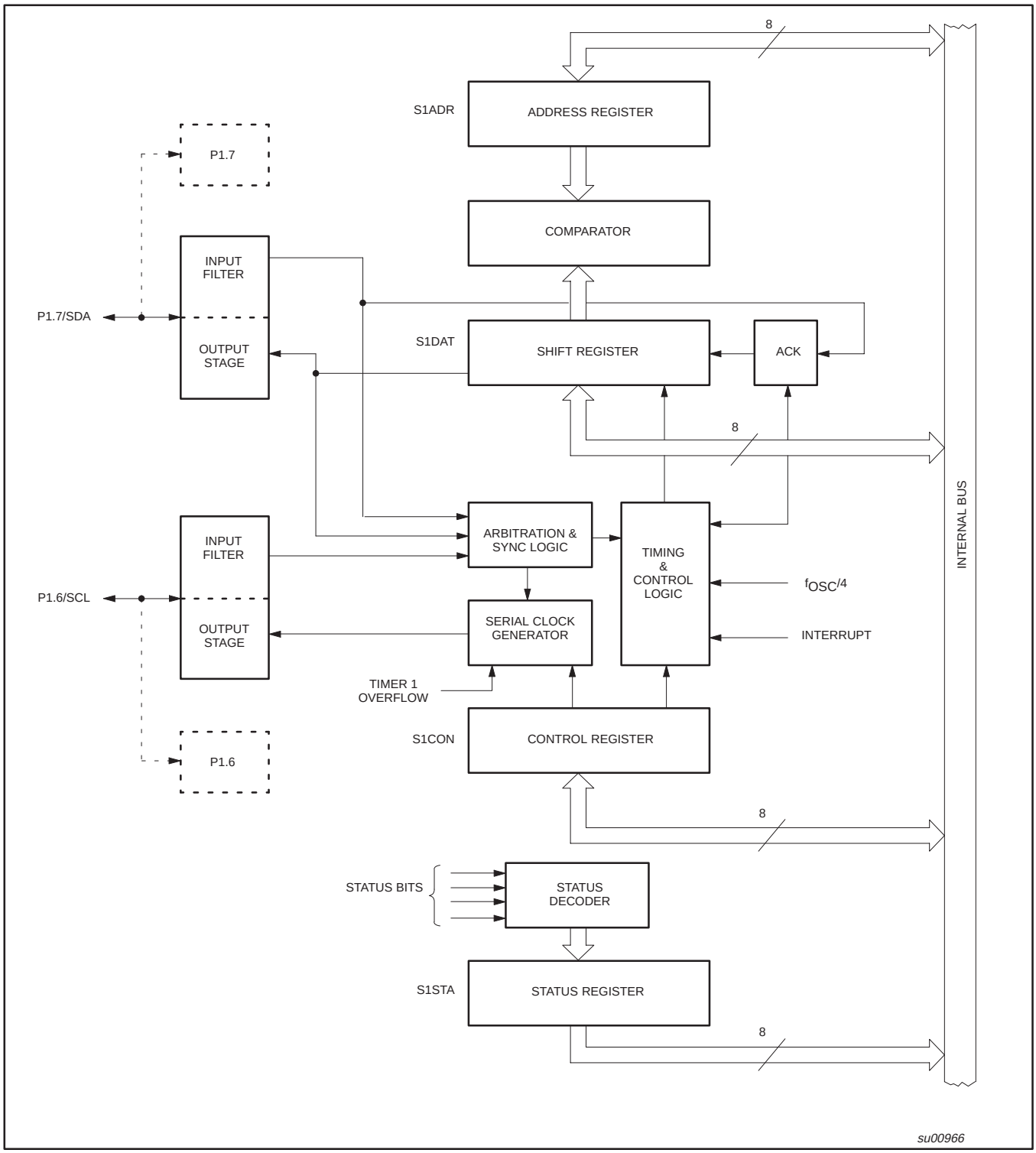



**Figure 3. I²C Bus Serial Interface Block Diagram**

**Serial Clock Generator**
This programmable clock pulse generator provides the SCL clock pulses when SIO1 is in the Master Transmitter or Master Receiver mode. It is switched off when SIO1 is in a Slave mode. The programmable output clock frequencies are: $f_{OSC}/120$, $f_{OSC}/9600$ (12-clock mode) or $f_{OSC}/60$, $f_{OSC}/4800$ (6-clock mode) and the Timer 1 overflow rate divided by eight. The output clock pulses have a 50% duty cycle unless the clock generator is synchronized with other SCL clock sources as described above.

**Timing and Control**
The timing and control logic generates the timing and control signals for serial byte handling. This logic block provides the shift pulses for S1DAT, enables the comparator, generates and detects start and stop conditions, receives and transmits acknowledge bits, controls the master and Slave modes, contains interrupt request logic, and monitors the $I^2C$ bus status.

**Control Register, S1CON**
This 7-bit special function register is used by the microcontroller to control the following SIO1 functions: start and restart of a serial transfer, termination of a serial transfer, bit rate, address recognition, and acknowledgment.

**Status Decoder and Status Register**
The status decoder takes all of the internal status bits and compresses them into a 5-bit code. This code is unique for each $I^2C$ bus status. The 5-bit code may be used to generate vector addresses for fast processing of the various service routines. Each service routine processes a particular bus status. There are 26 possible bus states if all four modes of SIO1 are used. The 5-bit status code is latched into the five most significant bits of the status register when the serial interrupt flag is set (by hardware) and remains stable until the interrupt flag is cleared by software. The three least significant bits of the status register are always zero. If the status code is used as a vector to service routines, then the routines are displaced by eight address locations. Eight bytes of code is sufficient for most of the service routines.

## The Four SIO1 Special Function Registers

The microcontroller interfaces to SIO1 via four special function registers. These four SFRs (S1ADR, S1DAT, S1CON, and S1STA) are described individually in the following sections.

**The Address Register, S1ADR**
The CPU can read from and write to this 8-bit, directly addressable SFR. S1ADR is not affected by the SIO1 hardware. The contents of this register are irrelevant when SIO1 is in a Master mode. In the Slave modes, the seven most significant bits must be loaded with the microcontroller's own slave address, and, if the least significant bit is set, the general call address (00H) is recognized; otherwise it is ignored.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| S1ADR (DBH) | X | X | X | X | X | X | X | GC |
| | own slave address | | | | | | | |

The most significant bit corresponds to the first bit received from the $I^2C$ bus after a start condition. A logic 1 in S1ADR corresponds to a high level on the $I^2C$ bus, and a logic 0 corresponds to a low level on the bus.

**The Data Register, S1DAT**
S1DAT contains a byte of serial data to be transmitted or a byte which has just been received. The CPU can read from and write to this 8-bit, directly addressable SFR while it is not in the process of shifting a byte. This occurs when SIO1 is in a defined state and the serial interrupt flag is set. Data in S1DAT remains stable as long as SI is set. Data in S1DAT is always shifted from right to left: the first bit to be transmitted is the MSB (bit 7), and, after a byte has been received, the first bit of received data is located at the MSB of S1DAT. While data is being shifted out, data on the bus is simultaneously being shifted in; S1DAT always contains the last data byte present on the bus. Thus, in the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data in S1DAT.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| S1DAT (DAH) | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 |
| | shift direction | | | | | | | |

SD7 - SD0:
Eight bits to be transmitted or just received. A logic 1 in S1DAT corresponds to a high level on the $I^2C$ bus, and a logic 0 corresponds to a low level on the bus. Serial data shifts through S1DAT from right to left. Figure 6 shows how data in S1DAT is serially transferred to and from the SDA line.

S1DAT and the ACK flag form a 9-bit shift register which shifts in or shifts out an 8-bit byte, followed by an acknowledge bit. The ACK flag is controlled by the SIO1 hardware and cannot be accessed by the CPU. Serial data is shifted through the ACK flag into S1DAT on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into S1DAT, the serial data is available in S1DAT, and the acknowledge bit is returned by the control logic during the ninth clock pulse. Serial data is shifted out from S1DAT via a buffer (BSD7) on the falling edges of clock pulses on the SCL line.

When the CPU writes to S1DAT, BSD7 is loaded with the content of S1DAT.7, which is the first bit to be transmitted to the SDA line (see Figure 7). After nine serial clock pulses, the eight bits in S1DAT will have been transmitted to the SDA line, and the acknowledge bit will be present in ACK. Note that the eight transmitted bits are shifted back into S1DAT.

**The Control Register, S1CON**
The CPU can read from and write to this 8-bit, directly addressable SFR. Two bits are affected by the SIO1 hardware: the SI bit is set when a serial interrupt is requested, and the STO bit is cleared when a STOP condition is present on the $I^2C$ bus. The STO bit is also cleared when ENS1 = "0".

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| S1CON (D8H) | CR2 | ENS1 | STA | STO | SI | AA | CR1 | CR0 |

**ENS1, the SIO1 Enable Bit:** ENS1 = "0": When ENS1 is "0", the SDA and SCL outputs are in a high impedance state. SDA and SCL input signals are ignored, SIO1 is in the "not addressed" slave state, and the STO bit in S1CON is forced to "0". No other bits are affected. P1.6 and P1.7 may be used as open drain I/O ports.

ENS1 = "1": When ENS1 is "1", SIO1 is enabled. The P1.6 and P1.7 port latches must be set to logic 1.

ENS1 should not be used to temporarily release SIO1 from the I2C bus since, when ENS1 is reset, the I2C bus status is lost. The AA flag should be used instead (see description of the AA flag in the following text).

## More Information on SIO1 Operating Modes

The four operating modes are:
– Master Transmitter
– Master Receiver
– Slave Receiver
– Slave Transmitter

Data transfers in each mode of operation are shown in Figures 8-11. These figures contain the following abbreviations:

| Abbreviation | Explanation |
|---|---|
| S | Start condition |
| SLA | 7-bit slave address |
| R | Read bit (high level at SDA) |
| W | Write bit (low level at SDA) |
| A | Acknowledge bit (low level at SDA) |
| $\overline{A}$ | Not acknowledge bit (high level at SDA) |
| Data | 8-bit data byte |
| P | Stop condition |

In Figures 8-11, circles are used to indicate when the serial interrupt flag is set. The numbers in the circles show the status code held in the S1STA register. At these points, a service routine must be executed to continue or complete the serial transfer. These service routines are not critical since the serial transfer is suspended until the serial interrupt flag is cleared by software.

When a serial interrupt routine is entered, the status code in S1STA is used to branch to the appropriate service routine. For each status code, the required software action and details of the following serial transfer are given in Tables 4-8.

**Master Transmitter mode**
In the Master Transmitter mode, a number of data bytes are transmitted to a slave receiver (see Figure 8). Before the Master Transmitter mode can be entered, S1CON must be initialized as follows:

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| S1CON (D8H) | CR2 | ENS1 | STA | STO | SI | AA | CR1 | CR0 |
| | bit rate | 1 | 0 | 0 | 0 | X | bit rate | |

CR0, CR1, and CR2 define the serial bit rate. ENS1 must be set to logic 1 to enable SIO1. If the AA bit is reset, SIO1 will not acknowledge its own slave address or the general call address in the event of another device becoming master of the bus. In other words, if AA is reset, SIO0 cannot enter a Slave mode. STA, STO, and SI must be reset.

The Master Transmitter mode may now be entered by setting the STA bit using the SETB instruction. The SIO1 logic will now test the $I^2C$ bus and generate a start condition as soon as the bus becomes free. When a START condition is transmitted, the serial interrupt flag (SI) is set, and the status code in the status register (S1STA) will be 08H. This status code must be used to vector to an interrupt service routine that loads S1DAT with the slave address and the data direction bit (SLA+W). The SI bit in S1CON must then be reset before the serial transfer can continue.

When the slave address and the direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again, and a number of status codes in S1STA are possible. There are 18H, 20H, or 38H for the Master mode and also 68H, 78H, or B0H if the Slave mode was enabled (AA = logic 1). The appropriate action to be taken for each of these status codes is detailed in Table 4. After a repeated start condition (state 10H). SIO1 may switch to the Master Receiver mode by loading S1DAT with SLA+R).

**Master Receiver mode**
In the Master Receiver mode, a number of data bytes are received from a slave transmitter (see Figure 9). The transfer is initialized as in the Master Transmitter mode. When the start condition has been transmitted, the interrupt service routine must load S1DAT with the 7-bit slave address and the data direction bit (SLA+R). The SI bit in S1CON must then be cleared before the serial transfer can continue.

When the slave address and the data direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again, and a number of status codes in S1STA are possible. These are 40H, 48H, or 38H for the Master mode and also 68H, 78H, or B0H if the Slave mode was enabled (AA = logic 1). The appropriate action to be taken for each of these status codes is detailed in Table 5. ENS1, CR1, and CR0 are not affected by the serial transfer and are not referred to in Table 5. After a repeated start condition (state 10H), SIO1 may switch to the Master Transmitter mode by loading S1DAT with SLA+W.

**Slave Receiver mode**
In the Slave Receiver mode, a number of data bytes are received from a master transmitter (see Figure 10). To initiate the Slave Receiver mode, S1ADR and S1CON must be loaded as follows:

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| S1ADR (DBH) | X | X | X | X | X | X | X | GC |
| | own slave address | | | | | | | |

The upper 7 bits are the address to which SIO1 will respond when addressed by a master. If the LSB (GC) is set, SIO1 will respond to the general call address (00H); otherwise it ignores the general call address.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| S1CON (D8H) | CR2 | ENS1 | STA | STO | SI | AA | CR1 | CR0 |
| | X | 1 | 0 | 0 | 0 | 1 | X | X |

CR0, CR1, and CR2 do not affect SIO1 in the Slave mode. ENS1 must be set to logic 1 to enable SIO1. The AA bit must be set to enable SIO1 to acknowledge its own slave address or the general call address. STA, STO, and SI must be reset.

When S1ADR and S1CON have been initialized, SIO1 waits until it is addressed by its own slave address followed by the data direction bit which must be “0” (W) for SIO1 to operate in the Slave Receiver mode. After its own slave address and the W bit have been received, the serial interrupt flag (I) is set and a valid status code can be read from S1STA. This status code is used to vector to an interrupt service routine, and the appropriate action to be taken for each of these status codes is detailed in Table 6. The Slave Receiver mode may also be entered if arbitration is lost while SIO1 is in the Master mode (see status 68H and 78H).

If the AA bit is reset during a transfer, SIO1 will return a not acknowledge (logic 1) to SDA after the next received data byte. While AA is reset, SIO1 does not respond to its own slave address or a general call address. However, the $I^2C$ bus is still monitored and address recognition may be resumed at any time by setting AA. This means that the AA bit may be used to temporarily isolate SIO1 from the $I^2C$ bus.

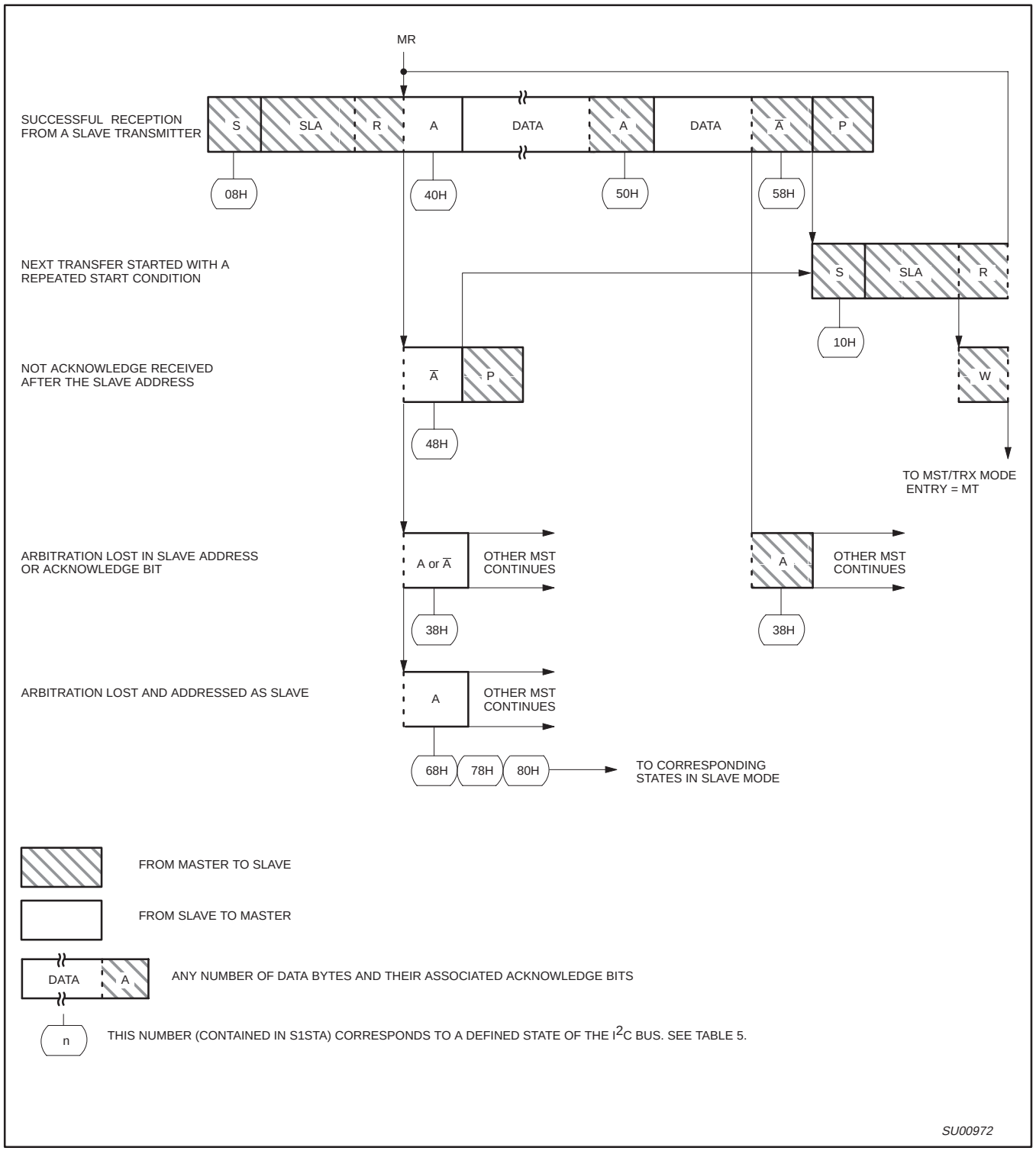



**Figure 9. Format and States in the Master Receiver Mode**

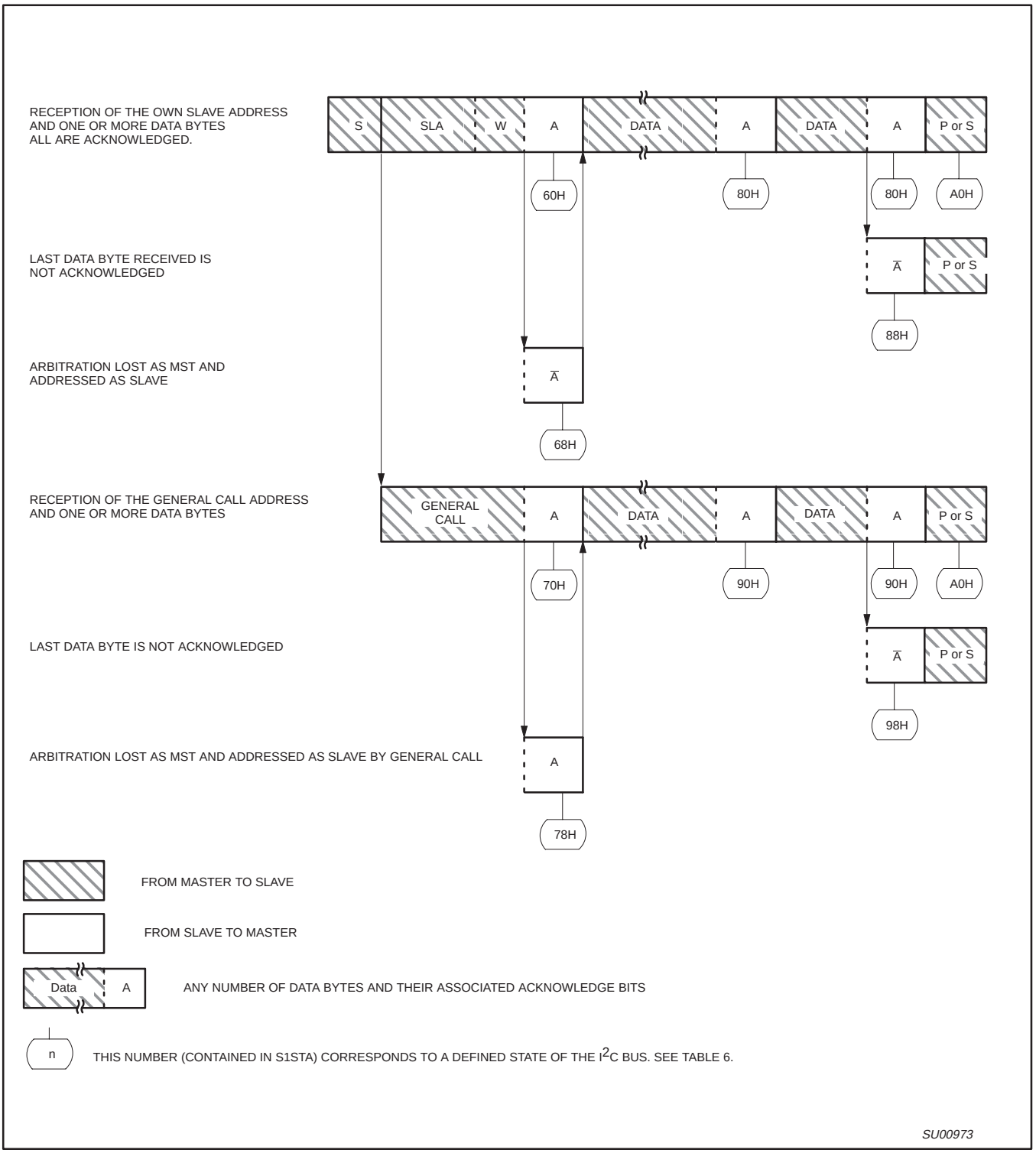



**Figure 10. Format and States in the Slave Receiver mode**

## TIMER 2 OPERATION

### Timer 2

Timer 2 is a 16-bit Timer/Counter which can operate as either an event timer or an event counter, as selected by C/$\overline{T}$2 in the special function register T2CON (see Figure 20). Timer 2 has three operating modes:

– Capture Mode
– Auto-Reload Mode (up or down counting)
– Baud Rate Generator Mode (see Table 10)

**Capture Mode**

In the capture mode there are two options which are selected by bit EXEN2 in T2CON. If EXEN2 = 0, then Timer 2 is a 16-bit timer or counter (as selected by C/$\overline{T}$2 in T2CON) which, upon overflowing sets bit TF2, the Timer 2 overflow bit. This bit can be used to generate an interrupt (by enabling the Timer 2 interrupt bit in the IE register). If EXEN2 = 1, Timer 2 operates as described above, with the added feature that a 1-to-0 transition at external input pin T2EX causes the current value in the Timer 2 registers, TL2 and TH2, to be captured into registers RCAP2L and RCAP2H. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set. If Timer 2 interrupt has been enabled, EXF2 will generate an interrupt (which vectors to the same location as Timer 2 overflow interrupt). The Timer 2 interrupt service routine can interrogate TF2 and EXF2 to determine which event caused the interrupt. The capture mode is illustrated in Figure 21 (There is no reload value for TL2 and TH2 in this mode). Even when a capture event occurs from T2EX, the counter keeps on counting T2EX pin transitions or osc/6 pulses (osc/12 in 12 clock mode).

**Auto-Reload Mode (Up or Down Counter)**

In the 16-bit auto-reload mode, Timer 2 can be configured as either a timer or counter (C/$\overline{T}$2 in T2CON), then programmed to count up or down. The counting direction is determined by bit DCEN (Down Counter Enable) which is located in the T2MOD register (see Figure 22). When reset is applied (DCEN = 0), Timer 2 defaults to counting up. If DCEN bit is set, Timer 2 can count up or down depending on the value of the T2EX pin.

Figure 23 shows Timer 2 which will count up automatically since DCEN = 0. In this mode there are two options selected by bit EXEN2 in T2CON register. If EXEN2 = 0, then Timer 2 counts up to 0FFFFH and sets the TF2 (Overflow Flag) bit upon overflow. This causes the Timer 2 registers to be reloaded with the 16-bit value in RCAP2L and RCAP2H. The values in RCAP2L and RCAP2H are preset by software.

If EXEN2 = 1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input pin T2EX. This transition also sets the EXF2 bit. The Timer 2 interrupt, if enabled, can be generated when either TF2 or EXF2 are 1.

In Figure 24 DCEN = 1 which enables Timer 2 to count up or down. This mode allows pin T2EX to control the direction of count. When a logic 1 is applied at pin T2EX Timer 2 will count up. Timer 2 will overflow at 0FFFFH and set the TF2 flag, which can then generate an interrupt, if the interrupt is enabled. This timer overflow also causes the 16-bit value in RCAP2L and RCAP2H to be reloaded into the timer registers TL2 and TH2.

When a logic 0 is applied at pin T2EX this causes Timer 2 to count down. The timer will underflow when TL2 and TH2 become equal to the value stored in RCAP2L and RCAP2H. Timer 2 underflow sets the TF2 flag and causes 0FFFFH to be reloaded into the timer registers TL2 and TH2.

The external flag EXF2 toggles when Timer 2 underflows or overflows. This EXF2 bit can be used as a 17th bit of resolution if needed. The EXF2 flag does not generate an interrupt in this mode of operation.

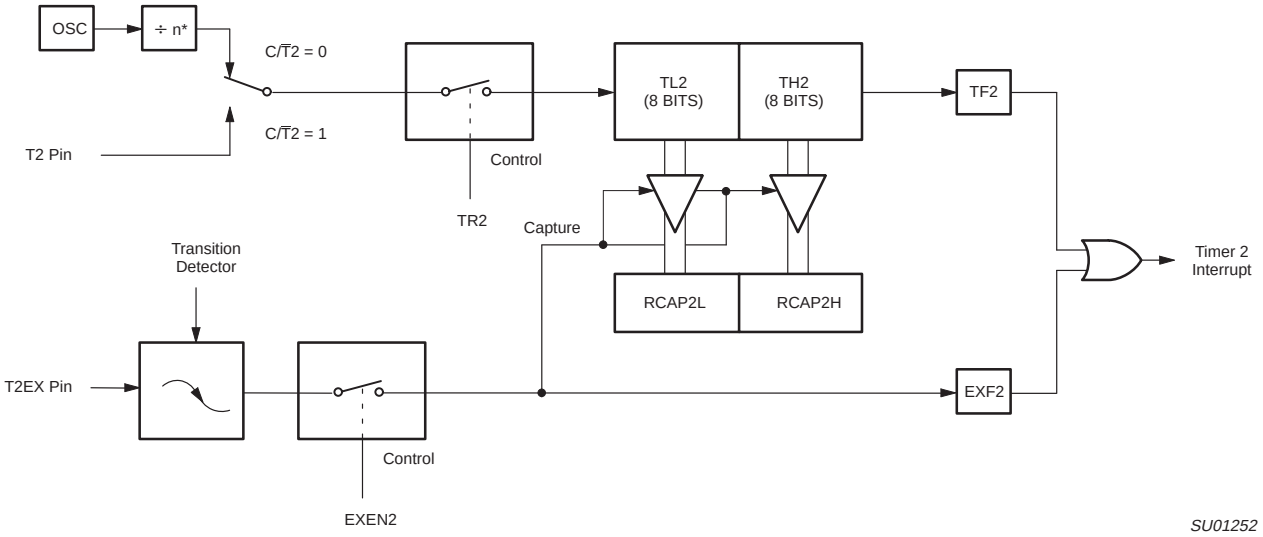

| Symbol | Position | Name and Significance |
|---|---|---|
| TF2 | T2CON.7 | Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK or TCLK = 1. |
| EXF2 | T2CON.6 | Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1). |
| RCLK | T2CON.5 | Receive clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock. |
| TCLK | T2CON.4 | Transmit clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock. |
| EXEN2 | T2CON.3 | Timer 2 external enable flag. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX. |
| TR2 | T2CON.2 | Start/stop control for Timer 2. A logic 1 starts the timer. |
| C/$\overline{T}$2 | T2CON.1 | Timer or counter select. (Timer 2)<br>0 = Internal timer (OSC/6 in 6-clock mode or OSC/12 in 12-clock mode)<br>1 = External event counter (falling edge triggered). |
| CP/$\overline{RL}$2 | T2CON.0 | Capture/Reload flag. When set, captures will occur on negative transitions at T2EX if EXEN2 = 1. When cleared, auto-reloads will occur either with Timer 2 overflows or negative transitions at T2EX when EXEN2 = 1. When either RCLK = 1 or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow. |

*SU01251*

**Figure 20. Timer/Counter 2 (T2CON) Control Register**

**Table 9. Timer 2 Operating Modes**

| RCLK + TCLK | CP/$\overline{RL2}$ | TR2 | MODE |
|---|---|---|---|
| 0 | 0 | 1 | 16-bit Auto-reload |
| 0 | 1 | 1 | 16-bit Capture |
| 1 | X | 1 | Baud rate generator |
| X | X | 0 | (off) |




* n = 6 in 6-clock mode, or 12 in 12-clock mode.

**Figure 21. Timer 2 in Capture Mode**

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

Not Bit Addressable

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

| Symbol | Function |
|---|---|
| — | Not implemented, reserved for future use.* |
| T2OE | Timer 2 Output Enable bit. |
| DCEN | Down Count Enable bit. When set, this allows Timer 2 to be configured as an up/down counter depending on the value of the T2EX pin. |

* 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.

SU01714

**Figure 22. Timer 2 Mode (T2MOD) Control Register**

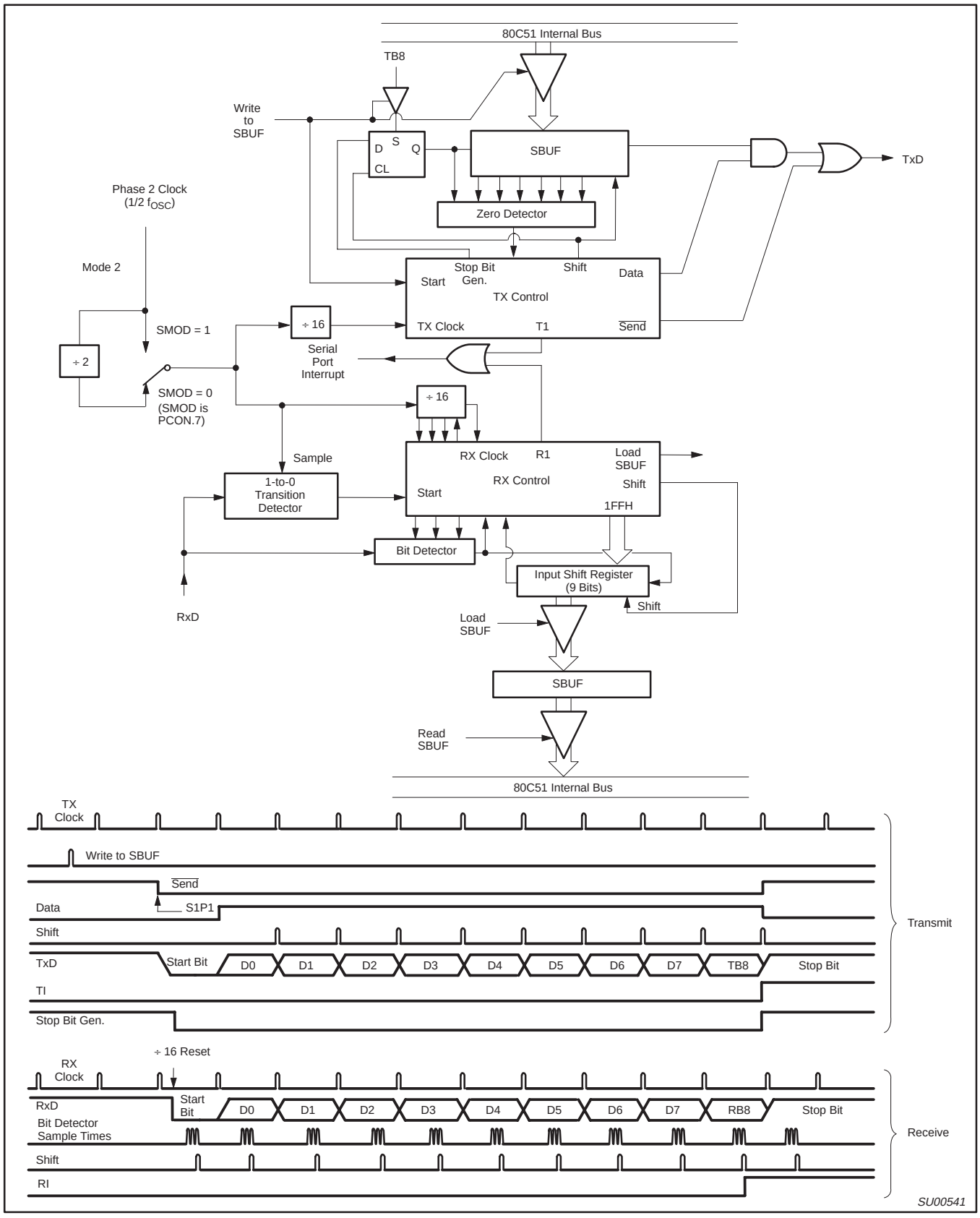



**Figure 30. Serial Port Mode 2**

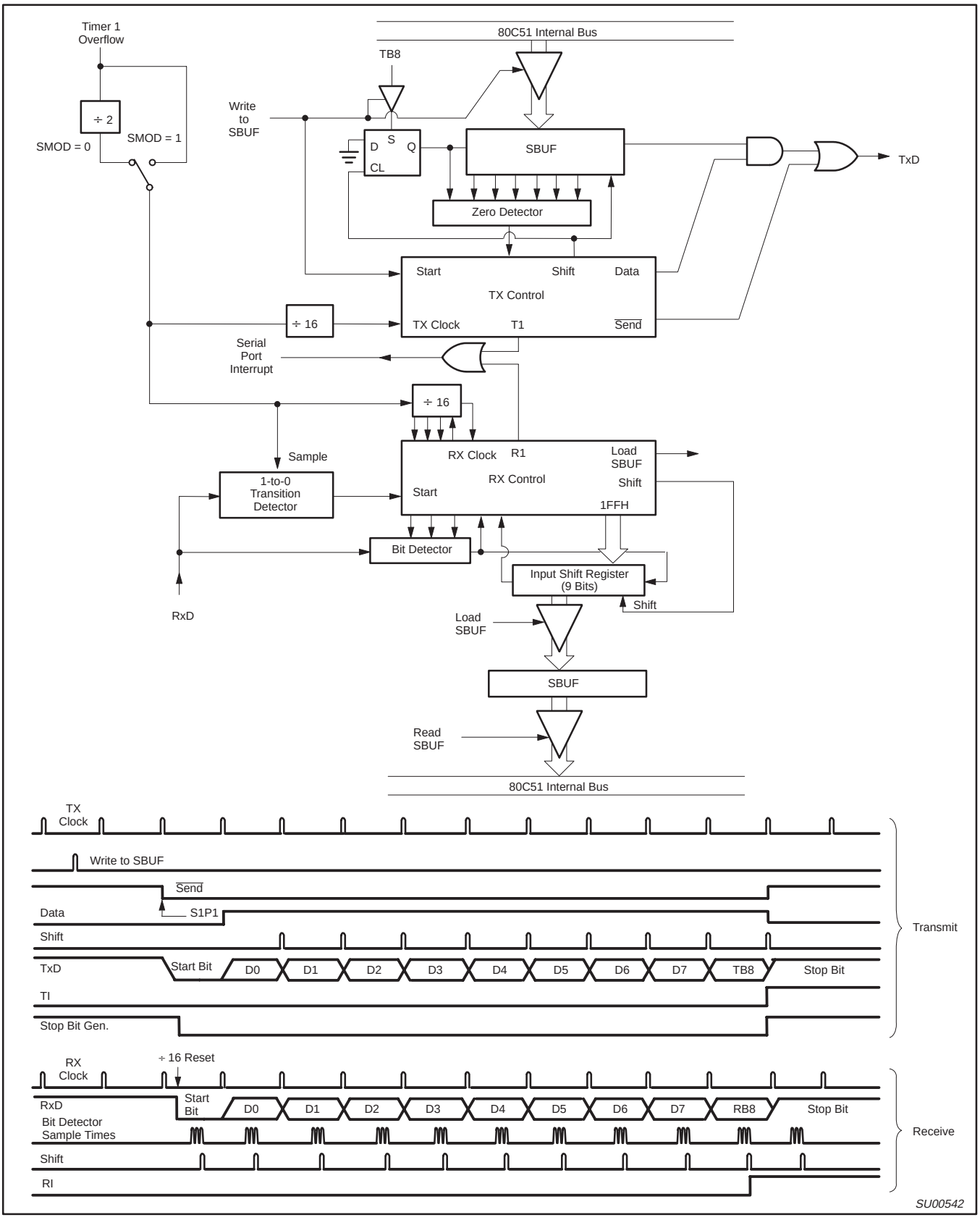



**Figure 31. Serial Port Mode 3**

```
INIT_WATCHDOG:
   MOV CCAPM4, #4CH         ; Module 4 in compare mode
   MOV CCAP4L, #0FFH        ; Write to low byte first
   MOV CCAP4H, #0FFH        ; Before PCA timer counts up to
                            ; FFFF Hex, these compare values
                            ; must be changed
   ORL CMOD, #40H           ; Set the WDTE bit to enable the
                            ; watchdog timer without changing
                            ; the other bits in CMOD
;
;*******************************************************************
;
; Main program goes here, but CALL WATCHDOG periodically.
;
;*******************************************************************
;
WATCHDOG:
   CLR EA                   ; Hold off interrupts
   MOV CCAP4L, #00          ; Next compare value is within
   MOV CCAP4H, CH           ; 255 counts of the current PCA
   SETB EA                  ; timer value
   RET
```

**Figure 52. PCA Watchdog Timer Initialization Code**

| RECORD TYPE | COMMAND/DATA FUNCTION |
|---|---|
| 03 | Miscellaneous Write Functions<br>`:nnxxxx03ffssddcc`<br>Where:<br>`nn = number of bytes (hex) in record`<br>`xxxx = required field, but value is a “don’t care”`<br>`03 = Write Function`<br>`ff = subfunction code`<br>`ss = selection code`<br>`dd = data input (as needed)`<br>`cc = checksum`<br><br>Subfunction Code = 01 (Erase Blocks)<br>`ff = 01`<br>`ss = block code as shown below:`<br>`block 0, 0k to 8k, 00H`<br>`block 1, 8k to 16k, 20H`<br>`block 2, 16k to 32k, 40H`<br>`block 3, 32k to 48k, 80H`<br>`block 4, 48k to 64k, C0H`<br>Example:<br>`:0200000301C03C erase block 4`<br><br>Subfunction Code = 04 (Erase Boot Vector and Status Byte)<br>`ff = 04`<br>`ss = don’t care`<br>Example:<br>`:020000030400F7 erase boot vector and status byte`<br><br>Subfunction Code = 05 (Program Security Bits)<br>`ff = 05`<br>`ss = 00 program security bit 1 (inhibit writing to Flash)`<br>`01 program security bit 2 (inhibit Flash verify)`<br>`02 program security bit 3 (disable external memory)`<br>Example:<br>`:020000030501F5 program security bit 2`<br><br>Subfunction Code = 06 (Program Status Byte or Boot Vector)<br>`ff = 06`<br>`ss = 00 program status byte`<br>`01 program boot vector`<br>Example:<br>`:030000030601FCF7 program boot vector with 0FCH`<br><br>Subfunction Code = 07 (Full Chip Erase)<br>Erases all blocks, security bits, and sets status and boot vector to default values<br>`ff = 07`<br>`ss = don’t care`<br>`dd = don’t care`<br>Example:<br>`:0100000307F5 full chip erase` |
| 04 | Display Device Data or Blank Check – Record type 04 causes the contents of the entire Flash array to be sent out the serial port in a formatted display. This display consists of an address and the contents of 16 bytes starting with that address. No display of the device contents will occur if security bit 2 has been programmed. Data to the serial port is initiated by the reception of any character and terminated by the reception of any character.<br><br>General Format of Function 04<br>`:05xxxx04sssseeeeffcc`<br>Where:<br>`05 = number of bytes (hex) in record`<br>`xxxx = required field, but value is a “don’t care”`<br>`04 = “Display Device Data or Blank Check” function code`<br>`ssss = starting address`<br>`eeee = ending address`<br>`ff = subfunction`<br>`00 = display data`<br>`01 = blank check`<br>`cc = checksum`<br>Example:<br>`:0500000440004FFF0069 display 4000–4FFF` |

| IAP CALL | PARAMETER |
|---|---|
| READ MANUFACTURER ID | Input Parameters:<br>R0 = osc freq (integer)<br>R1 = 00h<br>R1 = 80h (WDT feed, Rx2 & 66x only)<br>DPH = 00h<br>DPL = 00h (manufacturer ID)<br>Return Parameter<br>ACC = value of byte read<br>Sample routine:<br>;*****reads the Manufacturer ID (MID) *****<br>;***** MID returned in ACC (should be 15h for Philips)<br>RDMID:<br>MOV AUXR1,#20H ;set the ENBOOT bit<br>MOV R0,#11 ;FOSC<br>MOV R1,#00H ;read misc function<br>MOV DPTR,#0000H ;specify MID<br>CALL PGM_MTP ;execute the function<br>RET |
| READ DEVICE ID # 1 | Input Parameters:<br>R0 = osc freq (integer)<br>R1 = 00h<br>R1 = 80h (WDT feed, Rx2 & 66x only)<br>DPH = 00h<br>DPL = 01h (device ID # 1)<br>Return Parameter<br>ACC = value of byte read<br>Sample routine:<br>;*****reads the Device ID 1 (DID1) *****<br>;***** DID1 returned in ACC<br>RDDID1:<br>MOV AUXR1,#20H ;set the ENBOOT bit<br>MOV R0,#11 ;FOSC<br>MOV R1,#00H ;read misc function<br>MOV DPTR,#0001H ;specify device id 1<br>CALL PGM_MTP ;execute the function<br>RET |
| READ DEVICE ID # 2 | Input Parameters:<br>R0 = osc freq (integer)<br>R1 = 00h<br>R1 = 80h (WDT feed, Rx2 & 66x only)<br>DPH = 00h<br>DPL = 02h (device ID # 2)<br>Return Parameter<br>ACC = value of byte read<br>Sample routine:<br>;*****reads the Device ID 2 (DID2) *****<br>;***** DID2 returned in ACC<br>RDDID2:<br>MOV AUXR1,#20H ;set the ENBOOT bit<br>MOV R0,#11 ;FOSC<br>MOV R1,#00H ;read misc function<br>MOV DPTR,#0002H ;specify device id 2<br>CALL PGM_MTP ;execute the function<br>RET |

## Security

The security feature protects against software piracy and prevents the contents of the Flash from being read. The Security Lock bits are located in Flash. The P89C660/662/664/668 has 3 programmable security lock bits that will provide different levels of protection for the on-chip code and data (see Table 16).

**Table 16.**

| SECURITY LOCK BITS[1] | | | | PROTECTION DESCRIPTION |
|---|---|---|---|---|
| Level | LB1 | LB2 | LB3 | |
| 1 | 0 | 0 | 0 | MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory. |
| 2 | 1 | 0 | 0 | Same as level 1, plus block erase is disabled. Erase or programming of the status byte or boot vector is disabled. |
| 3 | 1 | 1 | 0 | Same as level 2, plus verify of code memory is disabled. |
| 4 | 1 | 1 | 1 | Same as level 3, plus external execution is disabled. |

**NOTE:**
1. Security bits are independent of each other. Full-chip erase may be performed regardless of the state of the security bits.
2. Any other combination of lockbits is undefined.
3. Setting LBx doesn't prevent programming of unprogrammed bits.

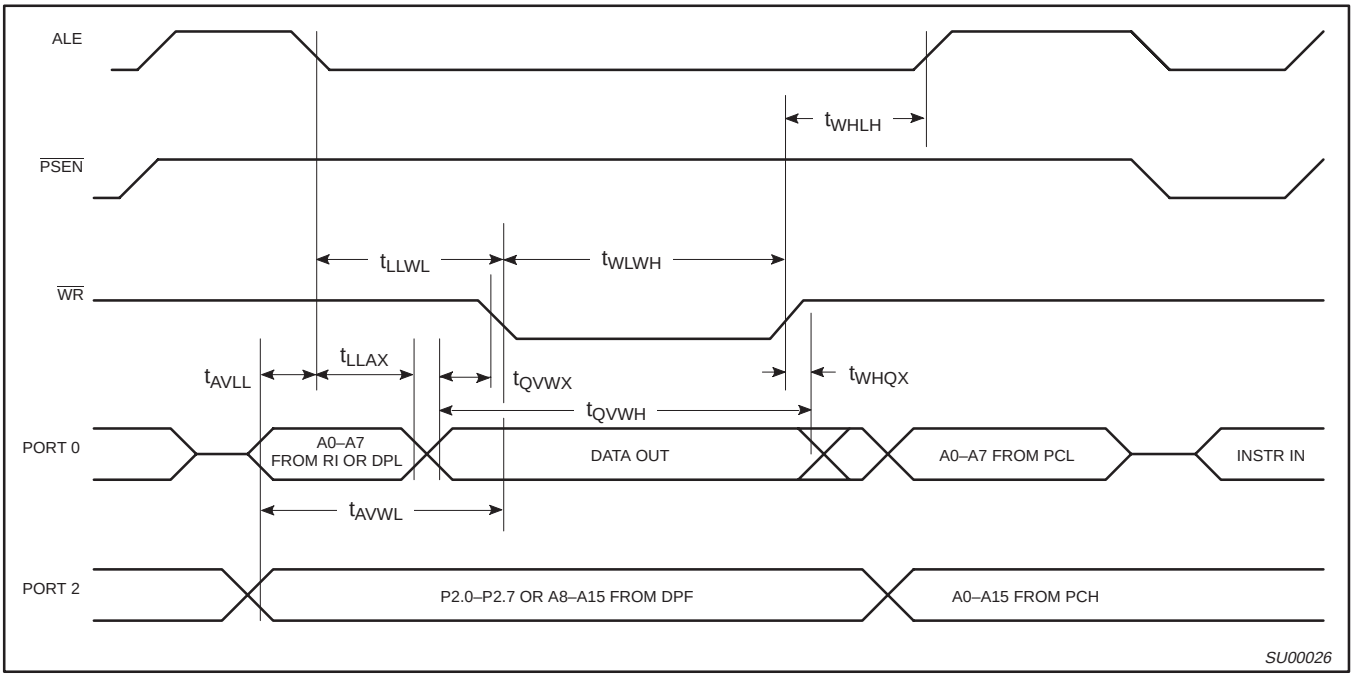



**Figure 59. External Data Memory Write Cycle**

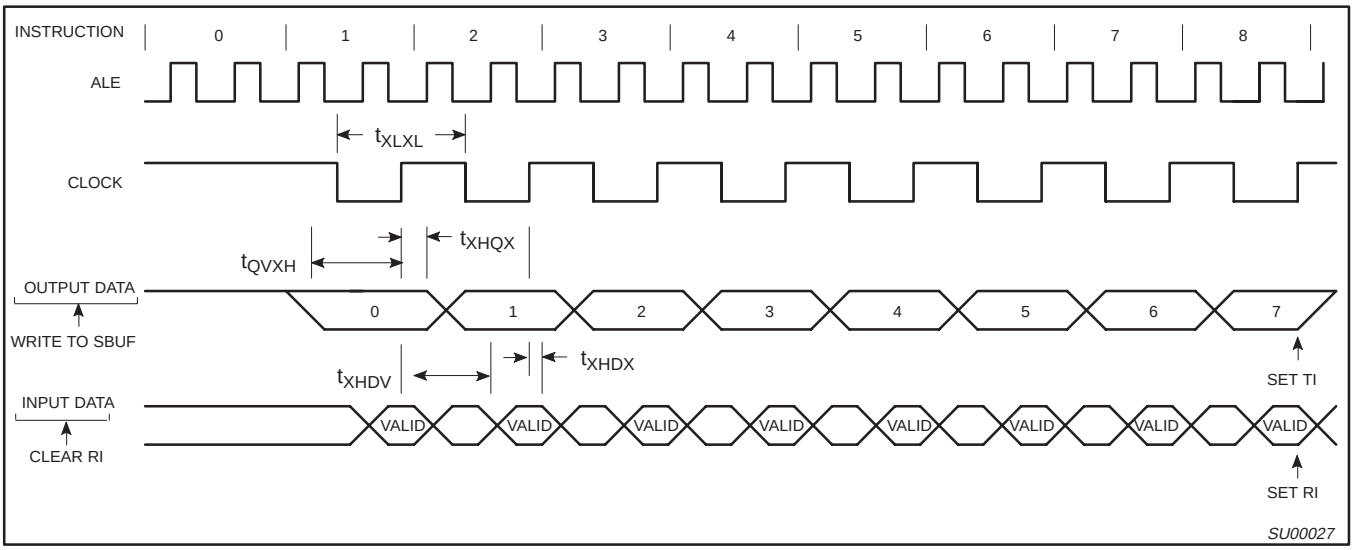



**Figure 60. Shift Register Mode Timing**

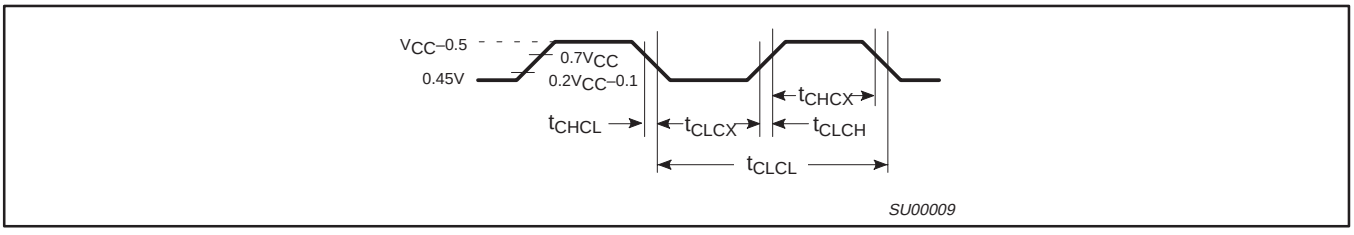
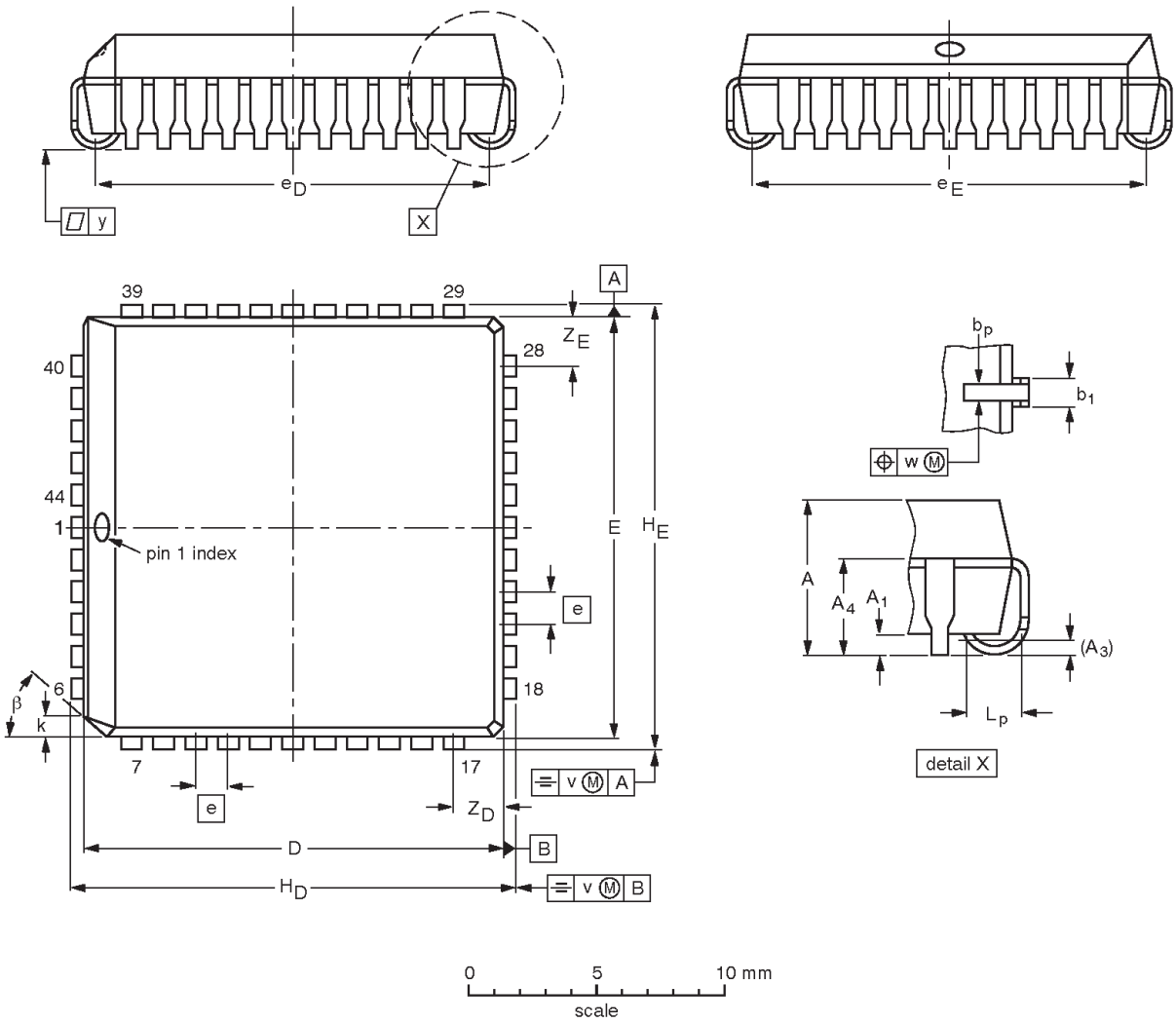



**Figure 61. External Clock Drive**

**PLCC44: plastic leaded chip carrier; 44 leads** **SOT187-2**




**DIMENSIONS (mm dimensions are derived from the original inch dimensions)**

| UNIT | A | $A_1$ min. | $A_3$ | $A_4$ max. | $b_p$ | $b_1$ | $D^{(1)}$ | $E^{(1)}$ | e | $e_D$ | $e_E$ | $H_D$ | $H_E$ | k | $L_p$ | v | w | y | $Z_D^{(1)}$ max. | $Z_E^{(1)}$ max. | β |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| mm | 4.57<br>4.19 | 0.51 | 0.25 | 3.05 | 0.53<br>0.33 | 0.81<br>0.66 | 16.66<br>16.51 | 16.66<br>16.51 | 1.27 | 16.00<br>14.99 | 16.00<br>14.99 | 17.65<br>17.40 | 17.65<br>17.40 | 1.22<br>1.07 | 1.44<br>1.02 | 0.18 | 0.18 | 0.1 | 2.16 | 2.16 | 45° |
| inches | 0.180<br>0.165 | 0.02 | 0.01 | 0.12 | 0.021<br>0.013 | 0.032<br>0.026 | 0.656<br>0.650 | 0.656<br>0.650 | 0.05 | 0.63<br>0.59 | 0.63<br>0.59 | 0.695<br>0.685 | 0.695<br>0.685 | 0.048<br>0.042 | 0.057<br>0.040 | 0.007 | 0.007 | 0.004 | 0.085 | 0.085 | |

**Note**

1. Plastic or metal protrusions of 0.25 mm (0.01 inch) maximum per side are not included.

| OUTLINE VERSION | REFERENCES | | | | EUROPEAN PROJECTION | ISSUE DATE |
|---|---|---|---|---|---|---|
| | IEC | JEDEC | JEITA | | | |
| SOT187-2 | 112E10 | MS-018 | EDR-7319 | | | ~~99-12-27~~<br>01-11-14 |