# Features

- **80C52 Compatible**
  - **8051 Pin and Instruction Compatible**
  - **Four 8-bit I/O ports + 2 I/O 2-wire Interface (TWI) Pins**
  - **Three 16-bit Timer/Counters**
  - **256 bytes Scratch Pad RAM**
  - **10 Interrupt Sources with 4 Priority Levels**
  - **Dual Data Pointer**
- **Variable Length MOVX for Slow RAM/Peripherals**
- **ISP (In-System-Programming) Using Standard $V_{cc}$ Power Supply**
- **Boot ROM Contains Low Level Flash Programming Routines and a Default Serial Loader**
- **High-speed Architecture**
  - **In Standard Mode:**
    **40 MHz (Vcc 2.7V to 5.5V, both Internal and external code execution)**
    **60 MHz (Vcc 4.5V to 5.5V and Internal Code execution only)**
  - **In X2 mode (6 Clocks/machine cycle)**
    **20 MHz (Vcc 2.7V to 5.5V, both Internal and external code execution)**
    **30 MHz (Vcc 4.5V to 5.5V and Internal Code execution only)**
  - **32K Bytes On-chip Flash Program/Data Memory**
  - **Byte and Page (128 Bytes) Erase and Write**
  - **100K Write Cycles**
- **On-chip 1024 Bytes Expanded RAM (XRAM)**
  - **Software Selectable Size (0, 256, 512, 768, 1024 Bytes)**
  - **256 Bytes Selected at Reset for TS87C51RB2/RC2 Compatibility**
- **Keyboard Interrupt Interface on Port P1**
- **400-Kbits/s Multimaster 2-wire Interface**
- **SPI Interface (Master/Slave Mode)**
- **Sub-clock 32 kHz Crystal Oscillator**
- **8-bit clock Prescaler**
- **Improved X2 Mode With Independant Selection for CPU and Each Peripheral**
- **Programmable Counter Array 5 Channels with:**
  - **High Speed Output**
  - **Compare/Capture**
  - **Pulse Width Modulator**
  - **Watchdog Timer Capabilities**
- **Asynchronous Port Reset**
- **Full-duplex Enhanced UART**
- **Dedicated Baud Rate Generator for UART**
- **Low EMI (Inhibit ALE)**
- **Hardware Watchdog Timer (One-time enabled with Reset-Out)**
- **Power Control Modes:**
  - **Idle Mode**
  - **Power-down Mode**
  - **Power-Off Flag**
- **Power Supply:**
  - **2.7 to 3.6 (3V Version)**
  - **2.7 to 5.5V (5V Version)**
- **Temperature Ranges: Commercial (0 to +70°C) and Industrial (-40°C to +85°C)**
- **Packages: PLC44, VQFP44**

# 8-bit Flash Microcontroller with 2-wire Interface

# AT89C51IC2

**Description**

AT89C51IC2 is a high performance Flash version of the 80C51 8-bit microcontrollers. It contains a 32K bytes Flash memory block for program and data.

The 32K bytes Flash memory can be programmed either in parallel mode or in serial mode with the ISP capability or with software. The programming voltage is internally generated from the standard $V_{CC}$ pin.

The AT89C51IC2 retains all features of the 80C52 with 256 bytes of internal RAM, a 10-source 4-level interrupt controller and three timer/counters.

In addition, the AT89C51IC2 has a 32 kHz Subsidiary clock Oscillator, a Programmable Counter Array, an XRAM of 1024 byte, a Hardware Watchdog Timer, a Keyboard Interface, a 2-wire interface, an SPI Interface, a more versatile serial channel that facilitates multiprocessor communication (EUART) and a speed improvement mechanism (X2 mode).

The fully static design of the AT89C51IC2 allows to reduce system power consumption by bringing the clock frequency down to any value, even DC, without loss of data.

The AT89C51IC2 has 2 software-selectable modes of reduced activity and 8-bit clock prescaler for further reduction in power consumption. In the Idle mode the CPU is frozen while the peripherals and the interrupt system are still operating. In the power-down mode the RAM is saved and all other functions are inoperative.

The added features of the AT89C51IC2 make it more powerful for applications that need pulse width modulation, high speed I/O and counting capabilities such as alarms, motor control, corded phones, smart card readers.

**Table 1.** Memory Size

| PLCC44 VQFP44 1.4 | Flash (bytes) | XRAM (bytes) | TOTAL RAM (bytes) | I/O |
|---|---|---|---|---|
| T89C51IC2 | 32k | 1024 | 1280 | 34 |

# Block Diagram

**Figure 1.** Block Diagram



(1): Alternate function of Port 1

(2): Alternate function of Port 3

## SFR Mapping

The Special Function Registers (SFRs) of the AT89C51IC2 fall into the following categories:

- C51 core registers: ACC, B, DPH, DPL, PSW, SP
- I/O port registers: P0, P1, P2, P3, PI2
- Timer registers: T2CON, T2MOD, TCON, TH0, TH1, TH2, TMOD, TL0, TL1, TL2, RCAP2L, RCAP2H
- Serial I/O port registers: SADDR, SADEN, SBUF, SCON
- PCA (Programmable Counter Array) registers: CCON, CCAPMx, CL, CH, CCAPxH, CCAPxL (x: 0 to 4)
- Power and clock control registers: PCON
- Hardware Watchdog Timer registers: WDTRST, WDTPRG
- Interrupt system registers: IEN0, IPL0, IPH0, IEN1, IPL1, IPH1
- Keyboard Interface registers: KBE, KBF, KBLS
- SPI registers: SPCON, SPSTR, SPDAT
- 2-wire Interface registers: SSCON, SSCS, SSDAT, SSADR
- BRG (Baud Rate Generator) registers: BRL, BDRCON
- Flash register: FCON
- Clock Prescaler register: CKRL
- 32 kHz Sub Clock Oscillator registers: CKSEL, OSSCON

**Table 2.** C51 Core SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|------|---|---|---|---|---|---|---|---|
| ACC | E0h | Accumulator | | | | | | | | |
| B | F0h | B Register | | | | | | | | |
| PSW | D0h | Program Status Word | CY | AC | F0 | RS1 | RS0 | OV | F1 | P |
| SP | 81h | Stack Pointer | | | | | | | | |
| DPL | 82h | Data Pointer Low byte | | | | | | | | |
| DPH | 83h | Data Pointer High byte | | | | | | | | |

**Table 3.** System Management SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|------|---|---|---|---|---|---|---|---|
| PCON | 87h | Power Control | SMOD1 | SMOD0 | - | - | GF1 | GF0 | PD | IDL |
| AUXR | 8Eh | Auxiliary Register 0 | - | - | M0 | | XRS1 | XRS0 | EXTRAM | AO |
| AUXR1 | A2h | Auxiliary Register 1 | - | - | ENBOOT | - | GF3 | 0 | - | DPS |
| CKRL | 97h | Clock Reload Register | - | - | - | - | - | - | - | - |
| CKSEL | 85h | Clock Selection Register | - | - | - | - | - | - | - | CKS |
| OSCON | 86h | Oscillator Control Register | - | - | - | - | - | SCLKT0 | OscBEn | OscAEn |
| CKCKON0 | 8Fh | Clock Control Register 0 | TWIX2 | WDTX2 | PCAX2 | SIX2 | T2X2 | T1X2 | T0X2 | X2 |
| CKCKON1 | AFh | Clock Control Register 1 | - | - | - | - | - | - | - | SPIX2 |

**Table 4.** Interrupt SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|------|---|---|---|---|---|---|---|---|
| IEN0 | A8h | Interrupt Enable Control 0 | EA | EC | ET2 | ES | ET1 | EX1 | ET0 | EX0 |
| IEN1 | B1h | Interrupt Enable Control 1 | - | - | - | - | - | ESPI | ETWI | KBD |
| IPH0 | B7h | Interrupt Priority Control High 0 | - | PPCH | PT2H | PHS | PT1H | PX1H | PT0H | PX0H |
| IPL0 | B8h | Interrupt Priority Control Low 0 | - | PPCL | PT2L | PLS | PT1L | PX1L | PT0L | PX0L |
| IPH1 | B3h | Interrupt Priority Control High 1 | - | - | - | - | - | SPIH | TWIH | KBDH |
| IPL1 | B2h | Interrupt Priority Control Low 1 | - | - | - | - | - | SPIL | TWIL | KBDL |

**Table 5.** Port SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|------|---|---|---|---|---|---|---|---|
| P0 | 80h | 8-bit Port 0 | | | | | | | | |
| P1 | 90h | 8-bit Port 1 | | | | | | | | |
| P2 | A0h | 8-bit Port 2 | | | | | | | | |
| P3 | B0h | 8-bit Port 3 | | | | | | | | |

**Table 6.** Timer SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|------|---|---|---|---|---|---|---|---|
| TCON | 88h | Timer/Counter 0 and 1 Control | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| TMOD | 89h | Timer/Counter 0 and 1 Modes | GATE1 | C/T1# | M11 | M01 | GATE0 | C/T0# | M10 | M00 |
| TL0 | 8Ah | Timer/Counter 0 Low Byte | | | | | | | | |
| TH0 | 8Ch | Timer/Counter 0 High Byte | | | | | | | | |
| TL1 | 8Bh | Timer/Counter 1 Low Byte | | | | | | | | |
| TH1 | 8Dh | Timer/Counter 1 High Byte | | | | | | | | |
| WDTRST | A6h | WatchDog Timer Reset | | | | | | | | |
| WDTPRG | A7h | WatchDog Timer Program | - | - | - | - | - | WTO2 | WTO1 | WTO0 |
| T2CON | C8h | Timer/Counter 2 control | TF2 | EXF2 | RCLK | TCLK | EXEN2 | TR2 | C/T2# | CP/RL2# |
| T2MOD | C9h | Timer/Counter 2 Mode | - | - | - | - | - | - | T2OE | DCEN |
| RCAP2H | CBh | Timer/Counter 2 Reload/Capture High byte | | | | | | | | |
| RCAP2L | CAh | Timer/Counter 2 Reload/Capture Low byte | | | | | | | | |
| TH2 | CDh | Timer/Counter 2 High Byte | | | | | | | | |
| TL2 | CCh | Timer/Counter 2 Low Byte | | | | | | | | |

**Table 7.** PCA SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|------|---|---|---|---|---|---|---|---|
| CCON | D8h | PCA Timer/Counter Control | CF | CR | - | CCF4 | CCF3 | CCF2 | CCF1 | CCF0 |
| CMOD | D9h | PCA Timer/Counter Mode | CIDL | WDTE | - | - | - | CPS1 | CPS0 | ECF |
| CL | E9h | PCA Timer/Counter Low byte | | | | | | | | |
| CH | F9h | PCA Timer/Counter High byte | | | | | | | | |

**Table 7.** PCA SFRs (Continued)

| Mnemo-nic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| CCAPM0 | DAh | PCA Timer/Counter Mode 0 | | ECOM0 | CAPP0 | CAPN0 | MAT0 | TOG0 | PWM0 | ECCF0 |
| CCAPM1 | DBh | PCA Timer/Counter Mode 1 | | ECOM1 | CAPP1 | CAPN1 | MAT1 | TOG1 | PWM1 | ECCF1 |
| CCAPM2 | DCh | PCA Timer/Counter Mode 2 | - | ECOM2 | CAPP2 | CAPN2 | MAT2 | TOG2 | PWM2 | ECCF2 |
| CCAPM3 | DDh | PCA Timer/Counter Mode 3 | | ECOM3 | CAPP3 | CAPN3 | MAT3 | TOG3 | PWM3 | ECCF3 |
| CCAPM4 | DEh | PCA Timer/Counter Mode 4 | | ECOM4 | CAPP4 | CAPN4 | MAT4 | TOG4 | PWM4 | ECCF4 |
| CCAP0H | FAh | PCA Compare Capture Module 0 H | CCAP0H7 | CCAP0H6 | CCAP0H5 | CCAP0H4 | CCAP0H3 | CCAP0H2 | CCAP0H1 | CCAP0H0 |
| CCAP1H | FBh | PCA Compare Capture Module 1 H | CCAP1H7 | CCAP1H6 | CCAP1H5 | CCAP1H4 | CCAP1H3 | CCAP1H2 | CCAP1H1 | CCAP1H0 |
| CCAP2H | FCh | PCA Compare Capture Module 2 H | CCAP2H7 | CCAP2H6 | CCAP2H5 | CCAP2H4 | CCAP2H3 | CCAP2H2 | CCAP2H1 | CCAP2H0 |
| CCAP3H | FDh | PCA Compare Capture Module 3 H | CCAP3H7 | CCAP3H6 | CCAP3H5 | CCAP3H4 | CCAP3H3 | CCAP3H2 | CCAP3H1 | CCAP3H0 |
| CCAP4H | FEh | PCA Compare Capture Module 4 H | CCAP4H7 | CCAP4H6 | CCAP4H5 | CCAP4H4 | CCAP4H3 | CCAP4H2 | CCAP4H1 | CCAP4H0 |
| CCAP0L | EAh | PCA Compare Capture Module 0 L | CCAP0L7 | CCAP0L6 | CCAP0L5 | CCAP0L4 | CCAP0L3 | CCAP0L2 | CCAP0L1 | CCAP0L0 |
| CCAP1L | EBh | PCA Compare Capture Module 1 L | CCAP1L7 | CCAP1L6 | CCAP1L5 | CCAP1L4 | CCAP1L3 | CCAP1L2 | CCAP1L1 | CCAP1L0 |
| CCAP2L | ECh | PCA Compare Capture Module 2 L | CCAP2L7 | CCAP2L6 | CCAP2L5 | CCAP2L4 | CCAP2L3 | CCAP2L2 | CCAP2L1 | CCAP2L0 |
| CCAP3L | EDh | PCA Compare Capture Module 3 L | CCAP3L7 | CCAP3L6 | CCAP3L5 | CCAP3L4 | CCAP3L3 | CCAP3L2 | CCAP3L1 | CCAP3L0 |
| CCAP4L | EEh | PCA Compare Capture Module 4 L | CCAP4L7 | CCAP4L6 | CCAP4L5 | CCAP4L4 | CCAP4L3 | CCAP4L2 | CCAP4L1 | CCAP4L0 |

**Table 8.** Serial I/O Port SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| SCON | 98h | Serial Control | FE/SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI | RI |
| SBUF | 99h | Serial Data Buffer | | | | | | | | |
| SADEN | B9h | Slave Address Mask | | | | | | | | |
| SADDR | A9h | Slave Address | | | | | | | | |
| BDRCON | 9Bh | Baud Rate Control | | | | BRR | TBCK | RBCK | SPD | SRC |
| BRL | 9Ah | Baud Rate Reload | | | | | | | | |

**Table 9.** SPI Controller SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| SPCON | C3h | SPI Control | SPR2 | SPEN | SSDIS | MSTR | CPOL | CPHA | SPR1 | SPR0 |
| SPSTA | C4h | SPI Status | SPIF | WCOL | SSERR | MODF | - | - | - | - |
| SPDAT | C5h | SPI Data | SPD7 | SPD6 | SPD5 | SPD4 | SPD3 | SPD2 | SPD1 | SPD0 |

**Table 10.** Two-Wire Interface Controller SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| SSCON | 93h | Synchronous Serial control | SSCR2 | SSPE | SSSTA | SSSTO | SSI | SSAA | SSCR1 | SSCR0 |
| SSCS | 94h | Synchronous Serial Status | SSC4 | SSC3 | SSC2 | SSC1 | SSC0 | 0 | 0 | 0 |
| SSDAT | 95h | Synchronous Serial Data | SSD7 | SSD6 | SSD5 | SSD4 | SSD3 | SSD2 | SSD1 | SSD0 |
| SSADR | 96h | Synchronous Serial Address | SSA7 | SSA6 | SSA5 | SSA4 | SSA3 | SSA2 | SSA1 | SSGC |

**Table 11.** Keyboard Interface SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| KBLS | 9Ch | Keyboard Level Selector | KBLS7 | KBLS6 | KBLS5 | KBLS4 | KBLS3 | KBLS2 | KBLS1 | KBLS0 |
| KBE | 9Dh | Keyboard Input Enable | KBE7 | KBE6 | KBE5 | KBE4 | KBE3 | KBE2 | KBE1 | KBE0 |
| KBF | 9Eh | Keyboard Flag Register | KBF7 | KBF6 | KBF5 | KBF4 | KBF3 | KBF2 | KBF1 | KBF0 |

Table below shows all SFRs with their address and their reset value.

**Table 12.** SFR Mapping

| | Bit addressable | Non Bit addressable | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0/8 | 1/9 | 2/A | 3/B | 4/C | 5/D | 6/E | 7/F | |
| F8h | | CH 0000 0000 | CCAP0H XXXX XXXX | CCAP1H XXXX XXXX | CCAPL2H XXXX XXXX | CCAPL3H XXXX XXXX | CCAPL4H XXXX XXXX | | FFh |
| F0h | B 0000 0000 | | | | | | | | F7h |
| E8h | | CL 0000 0000 | CCAP0L XXXX XXXX | CCAP1L XXXX XXXX | CCAPL2L XXXX XXXX | CCAPL3L XXXX XXXX | CCAPL4L XXXX XXXX | | EFh |
| E0h | ACC 0000 0000 | | | | | | | | E7h |
| D8h | CCON 00X0 0000 | CMOD 00XX X000 | CCAPM0 X000 0000 | CCAPM1 X000 0000 | CCAPM2 X000 0000 | CCAPM3 X000 0000 | CCAPM4 X000 0000 | | DFh |
| D0h | PSW 0000 0000 | FCON (1) XXXX 0000 | | | | | | | D7h |
| C8h | T2CON 0000 0000 | T2MOD XXXX XX00 | RCAP2L 0000 0000 | RCAP2H 0000 0000 | TL2 0000 0000 | TH2 0000 0000 | | | CFh |
| C0h | PI2 bit addressable XXXX XX11 | | | SPCON 0001 0100 | SPSTA 0000 0000 | SPDAT XXXX XXXX | | | C7h |
| B8h | IPL0 X000 000 | SADEN 0000 0000 | | | | | | | BFh |
| B0h | P3 1111 1111 | IEN1 XXXX X000 | IPL1 XXXX X000 | IPH1 XXXX X111 | | | | IPH0 X000 0000 | B7h |
| A8h | IEN0 0000 0000 | SADDR 0000 0000 | | | | | | CKCON1 XXXX XXX0 | AFh |
| A0h | P2 1111 1111 | | AUXR1 XXXX X0X0 | | | | WDTRST XXXX XXXX | WDTPRG XXXX X000 | A7h |
| 98h | SCON 0000 0000 | SBUF XXXX XXXX | BRL 0000 0000 | BDRCON XXX0 0000 | KBLS 0000 0000 | KBE 0000 0000 | KBF 0000 0000 | | 9Fh |
| 90h | P1 1111 1111 | | | SSCON 0000 0000 | SSCS 1111 1000 | SSDAT 1111 1111 | SSADR 1111 1110 | CKRL 1111 1111 | 97h |
| 88h | TCON 0000 0000 | TMOD 0000 0000 | TL0 0000 0000 | TL1 0000 0000 | TH0 0000 0000 | TH1 0000 0000 | AUXR XX0X 0000 | CKCON0 0000 0000 | 8Fh |
| 80h | P0 1111 1111 | SP 0000 0111 | DPL 0000 0000 | DPH 0000 0000 | | CKSEL XXXX XXX0 | OSSCON XXXX X001 | PCON 00X1 0000 | 87h |
| | 0/8 | 1/9 | 2/A | 3/B | 4/C | 5/D | 6/E | 7/F | |

reserved

# Pin Configurations

**Figure 2.** Pin Configurations

**PLCC44**

Top pins (6-44, left to right): 6 P1.4/CEX1, 5 P1.3/CEX0, 4 P1.2/ECI, 3 P1.1/T2EX/$\overline{SS}$, 2 P1.0/T2/XTALB1, 1 XTALB2, 44 VCC, 43 P0.0/AD0, 42 P0.1/AD1, 41 P0.2/AD2, 40 P0.3/AD3

Left pins:
- P1.5/CEX2/MISO — 7
- P1.6/CEX3/SCK — 8
- P1.7/CEx4/MOSI — 9
- RST — 10
- P3.0/RxD — 11
- PI2.1/SDA — 12
- P3.1/TxD — 13
- P3.2/$\overline{INT0}$ — 14
- P3.3/$\overline{INT1}$ — 15
- P3.4/T0 — 16
- P3.5/T1 — 17

Right pins:
- 39 — P0.4/AD4
- 38 — P0.5/AD5
- 37 — P0.6/AD6
- 36 — P0.7/AD7
- 35 — $\overline{EA}$
- 34 — PI2.0/SCL
- 33 — ALE/$\overline{PROG}$
- 32 — $\overline{PSEN}$
- 31 — P2.7/A15
- 30 — P2.6/A14
- 29 — P2.5/A13

Bottom pins (18-28): 18 P3.6/$\overline{WR}$, 19 P3.7/$\overline{RD}$, 20 XTAL2, 21 XTAL1, 22 VSS, 23 NIC*, 24 P2.0/A8, 25 P2.1/A9, 26 P2.2/A10, 27 P2.3/A11, 28 P2.4/A12

**VQFP44 1.4**

Top pins (44-34, left to right): 44 P1.4/CEX1, 43 P1.3/CEX0, 42 P1.2/ECI, 41 P1.1/T2EX/$\overline{SS}$, 40 P1.0/T2/XTALB1, 39 XTALB2, 38 VCC, 37 P0.0/AD0, 36 P0.1/AD1, 35 P0.2/AD2, 34 P0.3/AD3

Left pins:
- P1.5/CEX2/MISO — 1
- P1.6/CEX3/SCK — 2
- P1.7/CEX4/MOSI — 3
- RST — 4
- P3.0/RxD — 5
- PI2.1/SDA — 6
- P3.1/TxD — 7
- P3.2/$\overline{INT0}$ — 8
- P3.3/$\overline{INT1}$ — 9
- P3.4/T0 — 10
- P3.5/T1 — 11

Right pins:
- 33 — P0.4/AD4
- 32 — P0.5/AD5
- 31 — P0.6/AD6
- 30 — P0.7/AD7
- 29 — $\overline{EA}$
- 28 — PI2.0/SCL
- 27 — ALE/$\overline{PROG}$
- 26 — $\overline{PSEN}$
- 25 — P2.7/A15
- 24 — P2.6/A14
- 23 — P2.5/A13

Bottom pins (12-22): 12 P3.6/$\overline{WR}$, 13 P3.7/$\overline{RD}$, 14 XTAL2, 15 XTAL1, 16 VSS, 17 NIC*, 18 P2.0/A8, 19 P2.1/A9, 20 P2.2/A10, 21 P2.3/A11, 22 P2.4/A12

**Table 13.** Pin Description for 40/44 Pin Packages

| Mnemonic | Pin Number | | Type | Name and Function |
|---|---|---|---|---|
| | **PLCC44** | **VQFP44 1.4** | | |
| $V_{SS}$ | 22 | 16 | I | **Ground:** 0V reference |
| $V_{CC}$ | 44 | 38 | I | **Power Supply:** This is the power supply voltage for normal, idle and power-down operation |
| P0.0 - P0.7 | 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 must be polarized to $V_{CC}$ or $V_{SS}$ in order to prevent any parasitic current consumption. Port 0 is also the multiplexed low-order address and data bus during access to external program and data memory. In this application, it uses strong internal pull-up when emitting 1s. Port 0 also inputs the code bytes during EPROM programming. External pull-ups are required during program verification during which P0 outputs the code bytes. |
| P1.0 - P1.7 | 2 - 9 | 40 - 44<br>1 - 3 | I/O | **Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. 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. Port 1 also receives the low-order address byte during memory programming and verification.<br>Alternate functions for AT89C51IC2 Port 1 include: |
| | 2 | 40 | I/O | **P1.0**: Input/Output |
| | | | I/O | **T2 (P1.0):** Timer/Counter 2 external count input/Clockout |
| | | | I | **XTALB1 (P1.0):** Sub Clock input to the inverting oscillator amplifier |
| | 3 | 41 | I/O | **P1.1:** Input/Output |
| | | | I | **T2EX:** Timer/Counter 2 Reload/Capture/Direction Control |
| | | | I | $\overline{\text{SS}}$**:** SPI Slave Select |
| | 4 | 42 | I/O | **P1.2:** Input/Output |
| | | | I | **ECI:** External Clock for the PCA |
| | 5 | 43 | I/O | **P1.3:** Input/Output |
| | | | I/O | **CEX0:** Capture/Compare External I/O for PCA module 0 |
| | 6 | 44 | I/O | **P1.4:** Input/Output |
| | | | I/O | **CEX1:** Capture/Compare External I/O for PCA module 1 |
| | 7 | 1 | I/O | **P1.5:** Input/Output |
| | | | I/O | **CEX2:** Capture/Compare External I/O for PCA module 2 |
| | | | I/O | **MISO:** SPI Master Input Slave Output line |
| | | | | When SPI is in master mode, MISO receives data from the slave peripheral. When SPI is in slave mode, MISO outputs data to the master controller. |
| | 8 | 2 | I/O | **P1.6:** Input/Output |
| | | | I/O | **CEX3:** Capture/Compare External I/O for PCA module 3 |
| | | | I/O | **SCK:** SPI Serial Clock |
| | | | | **SCK outputs clock to the slave peripheral** |

**ATMEL**

**Table 13.** Pin Description for 40/44 Pin Packages  (Continued)

| Mnemonic | Pin Number | | Type | Name and Function |
|---|---|---|---|---|
| | PLCC44 | VQFP44 1.4 | | |
| | 9 | 3 | I/O | **P1.7:** Input/Output: |
| | | | I/O | **CEX4:** Capture/Compare External I/O for PCA module 4 |
| | | | I/O | **MOSI:** SPI Master Output Slave Input line |
| | | | | When SPI is in master mode, MOSI outputs data to the slave peripheral. When SPI is in slave mode, MOSI receives data from the master controller. |
| XTALA1 | 21 | 15 | I | **Crystal A 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits. |
| XTALA2 | 20 | 14 | O | **Crystal A 2:** Output from the inverting oscillator amplifier |
| XTALB1 | 2 | 40 | I | **Crystal B 1: (**Sub Clock) Input to the inverting oscillator amplifier and input to the internal clock generator circuits. |
| XTALB2 | 1 | 39 | O | **Crystal B 2:** (Sub Clock) Output from the inverting oscillator amplifier |
| 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 pulled low will source current because of the internal pull-ups. 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 emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @Ri), port 2 emits the contents of the P2 SFR. Some Port 2 pins receive the high order address bits during EPROM programming and verification. |
| P3.0 - P3.7 | 11,<br>13 - 19 | 5,<br>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 pulled low will source current because of the internal pull-ups. Port 3 also serves the special features of the 80C51 family, as listed below. |
| | 11 | 5 | I | **RXD (P3.0):** Serial input port |
| | 13 | 7 | O | **TXD (P3.1):** Serial output port |
| | 14 | 8 | I | $\overline{\text{INT0}}$ **(P3.2):** External interrupt 0 |
| | 15 | 9 | I | $\overline{\text{INT1}}$ **(P3.3):** External interrupt 1 |
| | 16 | 10 | I | **T0 (P3.4):** Timer 0 external input |
| | 17 | 11 | I | **T1 (P3.5):** Timer 1 external input |
| | 18 | 12 | O | $\overline{\text{WR}}$ **(P3.6):** External data memory write strobe |
| | 19 | 13 | O | $\overline{\text{RD}}$ **(P3.7):** External data memory read strobe |
| PI2.0 - PI2.1 | 34, 12 | 28, 6 | | **Port I2:** Port I2 is an open drain. It can be used as inputs (must be polarized to Vcc with external resistor to prevent any parasitic current consumption). |
| | 34 | 28 | I/O | **SCL (PI2.0): 2-wire Serial Clock** |
| | | | | SCL output the serial clock to slave peripherals<br>SCL input the serial clock from master |
| | 12 | 6 | I/O | **SDA (PI2.1): 2-wire Serial Data** |

**Table 13.** Pin Description for 40/44 Pin Packages (Continued)

| Mnemonic | Pin Number | | Type | Name and Function |
| | PLCC44 | VQFP44 1.4 | | |
| --- | --- | --- | --- | --- |
| | | | | SDA is the bidirectional 2-wire data line |
| RST | 10 | 4 | I/O | **Reset:** A high on this pin for two machine cycles while the oscillator is running, resets the device. An internal diffused resistor to $V_{SS}$ permits a power-on reset using only an external capacitor to $V_{CC}$. This pin is an output when the hardware watchdog forces a system reset. |
| ALE/$\overline{\text{PROG}}$ | 33 | 27 | O (I) | **Address Latch Enable/Program Pulse:** Output pulse for latching the low byte of the address during an access to external memory. In normal operation, ALE is emitted at a constant rate of 1/6 (1/3 in X2 mode) the oscillator frequency, and can be used for external timing or clocking. Note that one ALE pulse is skipped during each access to external data memory. This pin is also the program pulse input ($\overline{\text{PROG}}$) during Flash programming. ALE can be disabled by setting SFR's AUXR.0 bit. With this bit set, ALE will be inactive during internal fetches. |
| PSEN | 32 | 26 | O | **Program Strobe ENable:** The read strobe to external program memory. When executing code from the external program memory, $\overline{\text{PSEN}}$ is activated twice each machine cycle, except that two $\overline{\text{PSEN}}$ activations are skipped during each access to external data memory. $\overline{\text{PSEN}}$ is not activated during fetches from internal program memory. |
| EA | 35 | 29 | I | **External Access Enable:** $\overline{\text{EA}}$ must be externally held low to enable the device to fetch code from external program memory locations 0000H to FFFFH (RD). If security level 1 is programmed, $\overline{\text{EA}}$ will be internally latched on Reset. |

# Oscillators

**Overview**

Two oscillators are available for CPU:

- OSCA used for high frequency: Up to 48 MHz @5V +/- 10%
- OSCB used for low frequency: 32.768 kHz

Several operating modes are available and programmable by software:

- to switch OSCA to OSCB and vice-versa
- to stop OSCA or OSCB to reduce consumption

In order to optimize the power consumption and the execution time needed for a specific task, an internal prescaler feature has been implemented between the selected oscillator and the CPU.

**Registers**

**Table 14.** CKSEL Register

CKSEL - Clock Selection Register (85h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | **CKS** |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved** |
| 6 | - | **Reserved** |
| 5 | - | **Reserved** |
| 4 | - | **Reserved** |
| 3 | - | **Reserved** |
| 2 | - | **Reserved** |
| 1 | - | **Reserved** |
| 0 | CKS | **CPU Oscillator Select Bit: (CKS)**<br>Cleared, CPU and peripherals connected to OSCB<br>Set, CPU and peripherals connected to OSCA<br>Programmed by hardware after a Power-up regarding Hardware Security Byte (HSB).HSB.OSC (Default setting, OSCA selected) |

Reset Value = 0000 000'HSB.OSC'b (see Hardware Security Byte (HSB) Table 84)
Not bit addressable

**Table 15.** OSCCON Register

OSCCON- Oscillator Control Register (86h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | SCLKT0 | OscBEn | OscAEn |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved** |
| 6 | - | **Reserved** |
| 5 | - | **Reserved** |
| 4 | - | **Reserved** |
| 3 | - | **Reserved** |
| 2 | SCLKT0 | **Sub Clock Timer0**<br>Cleared by software to select T0 pin<br>Set by software to select T0 Sub Clock<br>Cleared by hardware after a Power Up |
| 1 | OscBEn | **OscB enable bit**<br>Set by software to run OscB<br>Cleared by software to stop OscB<br>Programmed by hardware after a Power-up regarding HSB.OSC (Default cleared, OSCB stopped) |
| 0 | OscAEn | **OscA enable bit**<br>Set by software to run OscA<br>Cleared by software to stop OscA<br>Programmed by hardware after a Power-up regarding HSB.OSC(Default Set, OSCA runs) |

Reset Value = XXXX X0'$\overline{HSB.OSC}$''HSB.OSC'b (see Hardware Security Byte (HSB) Table 84)
Not bit addressable

**Table 16.** CKRL Register

CKRL - Clock Reload Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| Bit Number | Mnemonic | Description |
|---|---|---|
| 7:0 | CKRL | **Clock Reload Register:**<br>Prescaler value |

Reset Value = 1111 1111b
Not bit addressable

**Table 17.** PCON Register

PCON - Power Control Register (87h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SMOD1 | SMOD0 | - | POF | GF1 | GF0 | PD | IDL |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | SMOD1 | **Serial port Mode bit 1**<br>Set to select double baud rate in mode 1, 2 or 3. |
| 6 | SMOD0 | **Serial port Mode bit 0**<br>Cleared to select SM0 bit in SCON register.<br>Set to select FE bit in SCON register. |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | POF | **Power-Off Flag**<br>Cleared to recognize next reset type.<br>Set by hardware when VCC rises from 0 to its nominal voltage. Can also be set by software. |
| 3 | GF1 | **General purpose Flag**<br>Cleared by software for general purpose usage.<br>Set by software for general purpose usage. |
| 2 | GF0 | **General purpose Flag**<br>Cleared by software for general purpose usage.<br>Set by software for general purpose usage. |
| 1 | PD | **Power-Down mode bit**<br>Cleared by hardware when reset occurs.<br>Set to enter power-down mode. |
| 0 | IDL | **Idle mode bit**<br>Cleared by hardware when interrupt or reset occurs.<br>Set to enter idle mode. |

Reset Value = 00X1 0000b
Not bit addressable

## Functional Block Diagram

**Figure 3.** Functional Oscillator Block Diagram
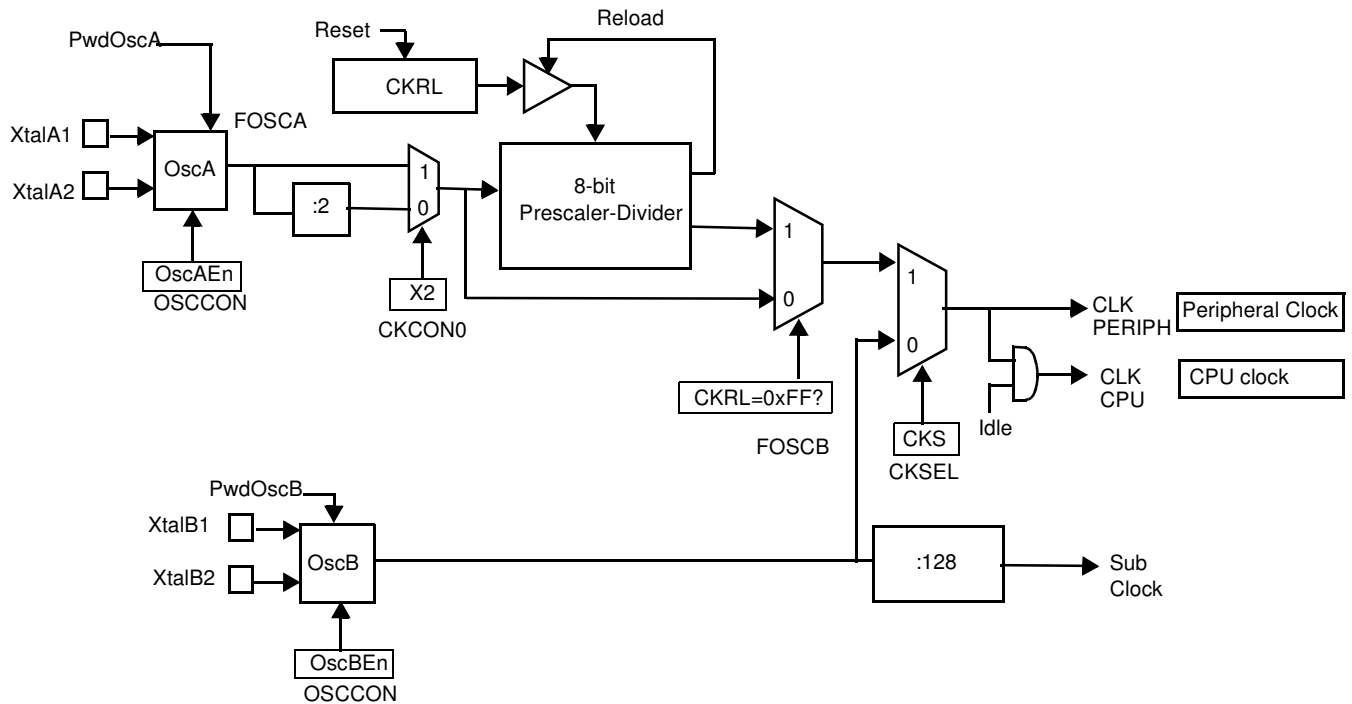


## Operating Modes

**Reset**
A hardware RESET puts the Clock generator in the following state:

The selected oscillator depends on OSC bit in Hardware Security Byte (HSB) (see HSB Table 84)

HSB.OSC = 1 (Oscillator A selected)
- OscAEn = 1 & OscBEn = 0: OscA is running, OscB is stopped.
- CKS = 1: OscA is selected for CPU.

HSB.OSC = 0 (Oscillator B selected)
- OscAEn = 0 & OscBEn = 1: OscB is running, OscA is stopped.
- CKS = 0: OscB is selected for CPU.

**Functional Modes**

*Normal Modes*
- CPU and Peripherals clock depend on the software selection using CKCON0, CKCON1 and CKRL registers
- CKS bit in CKSEL register selects either OscA or OscB
- CKRL register determines the frequency of the OscA clock.

• It is always possible to switch dynamically by software from OscA to OscB, and vice versa by changing CKS bit.

*Idle Modes*

• IDLE modes are achieved by using any instruction that writes into PCON.0 bit (IDL)

• IDLE modes A and B depend on previous software sequence, prior to writing into PCON.0 bit:

• IDLE MODE A: OscA is running (OscAEn = 1) and selected (CKS = 1)

• IDLE MODE B: OscB is running (OscBEn = 1) and selected (CKS = 0)

• The unused oscillator OscA or OscB can be stopped by software by clearing OscAEn or OscBEn respectively.

• IDLE mode can be canceled either by Reset, or by activation of any enabled interruption

• In both cases, PCON.0 bit (IDL) is cleared by hardware

• Exit from IDLE modes will leave Oscillators control bits (OscEnA, OscEnB, CKS) unchanged.

*Power Down Modes*

• POWER DOWN modes are achieved by using any instruction that writes into PCON.1 bit (PD)

• POWER DOWN modes A and B depend on previous software sequence, prior to writing into PCON.1 bit:

• Both OscA and OscB will be stopped.

• POWER DOWN mode can be cancelled either by a hardware Reset, an external interruption, or the keyboard interrupt.

• By Reset signal: The CPU will restart according to OSC bit in Hardware Security Bit (HSB) register.

• By INT0 or INT1 interruption, if enabled: (standard behavioral), request on Pads must be driven low enough to ensure correct restart of the oscillator which was selected when entering in Power down.

• By keyboard Interrupt if enabled: a hardware clear of the PCON.1 flag ensure the restart of the oscillator which was selected when entering in Power down.

**Table 18.** Overview

| PCON.1 | PCON.0 | OscBEn | OscAEn | CKS | Selected Mode | Comment |
|--------|--------|--------|--------|-----|---------------|---------|
| 0 | 0 | 0 | 1 | 1 | NORMAL MODE A, OscB stopped | Default mode after power-up or Warm Reset |
| 0 | 0 | 1 | 1 | 1 | NORMAL MODE A, OscB running | Default mode after power-up or Warm Reset + OscB running |
| 0 | 0 | 1 | 0 | 0 | NORMAL MODE B, OscA stopped | OscB running and selected |
| 0 | 0 | 1 | 1 | 0 | NORMAL MODE B, OscA running | OscB running and selected + OscA running |
| X | X | 0 | 0 | X | INVALID | OscA & OscB cannot be stopped at the same time |
| X | X | X | 0 | 1 | INVALID | OscA must not be stopped, as used for CPU and peripherals |
| X | X | 0 | X | 0 | INVALID | OscB must not be stopped as used for CPU and peripherals |

**Table 18.**  Overview  (Continued)

| PCON.1 | PCON.0 | OscBEn | OscAEn | CKS | Selected Mode | Comment |
|---|---|---|---|---|---|---|
| 0 | 1 | X | 1 | 1 | IDLE MODE A | The CPU is off, OscA supplies the peripherals, OscB can be disabled (OscBEn = 0) |
| 0 | 1 | 1 | X | 0 | IDLE MODE B | The CPU is off, OscB supplies the peripherals, OscA can be disabled (OscAEn = 0) |
| 1 | X | X | 1 | X | POWER DOWN MODE | The CPU and peripherals are off, OscA and OscB are stopped |

## Design Considerations

**Oscillators Control**

- PwdOscA and PwdOscB signals are generated in the Clock generator and used to control the hard blocks of oscillators A and B.
- PwdOscA ='1' stops OscA
- PwdOscB ='1' stops OscB
- The following tables summarize the Operating modes:

| PCON.1 | OscAEn | PwdOscA | Comments |
|---|---|---|---|
| 0 | 1 | 0 | OscA running |
| 1 | X | 1 | OscA stopped by Power-down mode |
| 0 | 0 | 1 | OscA stopped by clearing OscAEn |

| PCON.1 | OscBEn | PwdOscB | Comments |
|---|---|---|---|
| 0 | 1 | 0 | OscB running |
| 1 | X | 1 | OscB stopped by Power-down mode |
| 0 | 0 | 1 | OscB stopped by clearing OscBEn |

**Prescaler Divider**

- A hardware RESET puts the prescaler divider in the following state:
  - CKRL = FFh: $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSCA}/2$ (Standard C51 feature)
- CKS signal selects OSCA or OSCB: $F_{CLK\ OUT} = F_{OSCA}$ or $F_{OSCB}$
- Any value between FFh down to 00h can be written by software into CKRL register in order to divide frequency of the selected oscillator:
  - CKRL = 00h: minimum frequency
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSCA}/1020$ (Standard Mode)
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSCA}/510$ (X2 Mode)
  - CKRL = FFh: maximum frequency
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSCA}/2$ (Standard Mode)
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSCA}$ (X2 Mode)

- $F_{CLK\ CPU}$ and $F_{CLK\ PERIPH}$, for CKRL<>0xFF
  In X2 Mode:

$$F_{CPU} = F_{CLKPERIPH} = \frac{F_{OSCA}}{2 \times (255 - CKRL)}$$

In X1 Mode:

$$F_{CPU} = F_{CLKPERIPH} = \frac{F_{OSCA}}{4 \times (255 - CKRL)}$$

**Timer 0: Clock Inputs**

**Figure 4.** Timer 0: Clock Inputs
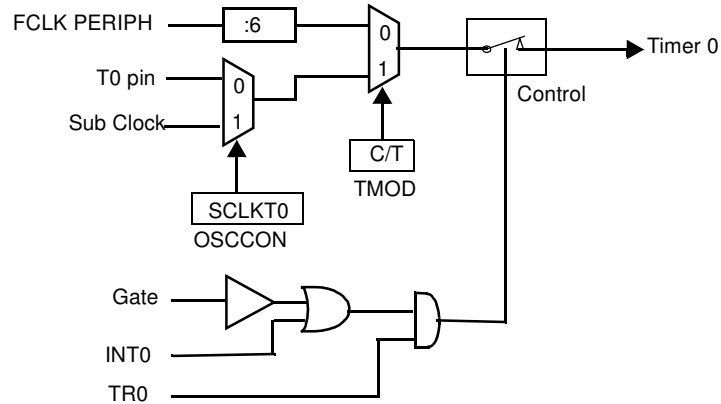


Note:    The SCLKT0 bit in OSCCON register allows to select Timer 0 Subsidiary clock.

SCLKT0 = 0: Timer 0 uses the standard T0 pin as clock input (Standard mode)

SCLKT0 = 1: Timer 0 uses the special Sub Clock as clock input, this feature can be use as periodic interrupt for time clock.

**Enhanced Features**    In comparison to the original 80C52, the AT89C51IC2 implements some new features, which are**:**

- The X2 option
- The Dual Data Pointer
- The extended RAM
- The Programmable Counter Array (PCA)
- The Hardware Watchdog
- The SPI interface
- The 2-wire interface
- The 4 level interrupt priority system
- The power-off flag
- The Power On Reset
- The ONCE mode
- The ALE disabling
- Some enhanced features are also located in the UART and the timer 2

**X2 Feature and OSCA Clock Generation**    The AT89C51IC2 core needs only 6 clock periods per machine cycle. This feature called "X2" provides the following advantages:

- Divide frequency crystals by 2 (cheaper crystals) while keeping same CPU power.
- Save power consumption while keeping same CPU power (oscillator power saving).
- Save power consumption by dividing dynamically the operating frequency by 2 in operating and idle modes.
- Increase CPU power by 2 while keeping same crystal frequency.

In order to keep the original C51 compatibility, a divider by 2 is inserted between the XTALA1 signal and the main clock input of the core (phase generator). This divider may be disabled by software.

**Description**    The clock for the whole circuit and peripherals is first divided by two before being used by the CPU core and the peripherals.

This allows any cyclic ratio to be accepted on XTALA1 input. In X2 mode, as this divider is bypassed, the signals on XTALA1 must have a cyclic ratio between 40 to 60%.

Figure 5. shows the clock generation block diagram.x2 bit is validated on the rising edge of the XTALA1÷2 to avoid glitches when switching from X2 to STD mode. Figure 6. shows the switching mode waveforms.

**Figure 5.** Clock Generation Diagram
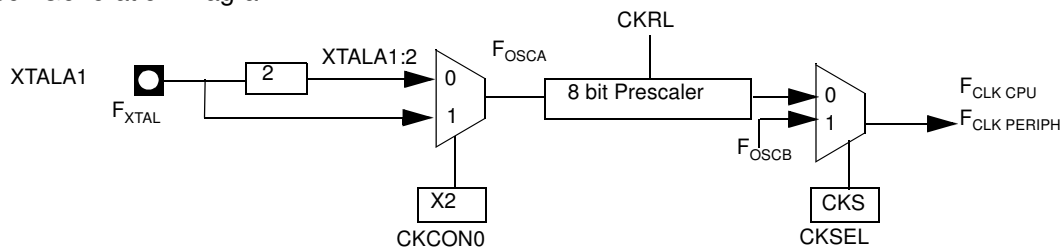
**Figure 6.** Mode Switching Waveforms
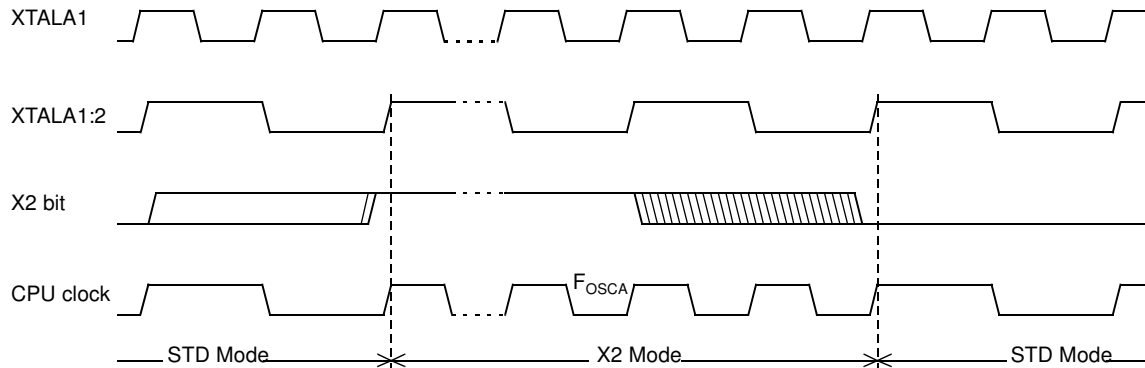


The X2 bit in the CKCON0 register (see Table 19) allow to switch from 12 clock periods per instruction to 6 clock periods and vice versa. At reset, the speed is setting according to X2 bit of Hardware Security Byte (HSB). By default, Standard mode is actived. Setting the X2 bit activates the X2 feature (X2 mode).

The T0X2, T1X2, T2X2, UartX2, PcaX2, WdX2 and I2CX2 bits in the CKCON0 register (See Table 19.) and SPIX2 bit in the CKCON1 register (see Table 20) allow to switch from standard peripheral speed (12 clock periods per peripheral clock cycle) to fast peripheral speed (6 clock periods per peripheral clock cycle). These bits are active only in X2 mode.

More information about the X2 mode can be found in the application note "How to take advantage of the X2 features in TS80C51 microcontroller?"

**Table 19.** CKCON0 Register

CKCON0 - Clock Control Register (8Fh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SPIX2 | WDX2 | PCAX2 | SIX2 | T2X2 | T1X2 | T0X2 | X2 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | I2CX2 | **2-wire clock** (This control bit is validated when the CPU clock X2 is set; when X2 is low, this bit has no effect)<br>Cleared to select 6 clock periods per peripheral clock cycle.<br>Set to select 12 clock periods per peripheral clock cycle. |
| 6 | WDX2 | **Watchdog clock** (This control bit is validated when the CPU clock X2 is set; when X2 is low, this bit has no effect)<br>Cleared to select 6 clock periods per peripheral clock cycle.<br>Set to select 12 clock periods per peripheral clock cycle. |
| 5 | PCAX2 | **Programmable Counter Array clock** (This control bit is validated when the CPU clock X2 is set; when X2 is low, this bit has no effect)<br>Cleared to select 6 clock periods per peripheral clock cycle.<br>Set to select 12 clock periods per peripheral clock cycle. |
| 4 | SIX2 | **Enhanced UART clock (Mode 0 and 2)** (This control bit is validated when the CPU clock X2 is set; when X2 is low, this bit has no effect)<br>Cleared to select 6 clock periods per peripheral clock cycle.<br>Set to select 12 clock periods per peripheral clock cycle. |
| 3 | T2X2 | **Timer2 clock** (This control bit is validated when the CPU clock X2 is set; when X2 is low, this bit has no effect)<br>Cleared to select 6 clock periods per peripheral clock cycle.<br>Set to select 12 clock periods per peripheral clock cycle. |
| 2 | T1X2 | **Timer1 clock** (This control bit is validated when the CPU clock X2 is set; when X2 is low, this bit has no effect)<br>Cleared to select 6 clock periods per peripheral clock cycle.<br>Set to select 12 clock periods per peripheral clock cycle. |
| 1 | T0X2 | **Timer0 clock** (This control bit is validated when the CPU clock X2 is set; when X2 is low, this bit has no effect)<br>Cleared to select 6 clock periods per peripheral clock cycle.<br>Set to select 12 clock periods per peripheral clock cycle. |
| 0 | X2 | **CPU clock**<br>Cleared to select 12 clock periods per machine cycle (STD mode) for CPU and all the peripherals.<br>Set to select 6clock periods per machine cycle (X2 mode) and to enable the individual peripherals "X2" bits.<br>Programmed by hardware after Power-up regarding Hardware Security Byte (HSB), Default setting, X2 is cleared. |

Reset Value = 0000 000'HSB.X2'b
Not bit addressable

**Table 20.** CKCON1 Register

CKCON1 - Clock Control Register (AFh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | **SPIX2** |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved** |
| 6 | - | **Reserved** |
| 5 | - | **Reserved** |
| 4 | - | **Reserved** |
| 3 | - | **Reserved** |
| 2 | - | **Reserved** |
| 1 | - | **Reserved** |
| 0 | SPIX2 | **SPI** (This control bit is validated when the CPU clock X2 is set; when X2 is low, this bit has no effect)<br>Clear to select 6 clock periods per peripheral clock cycle.<br>Set to select 12 clock periods per peripheral clock cycle. |

Reset Value = XXXX XXX0b
Not bit addressable

## Dual Data Pointer Register

The additional data pointer can be used to speed up code execution and reduce code size.

The dual DPTR structure is a way by which the chip will specify the address of an external data memory location. There are two 16-bit DPTR registers that address the external memory, and a single bit called DPS = AUXR1.0 (see Table 21) that allows the program code to switch between them (Refer to Figure 7).

**Figure 7.** Use of Dual Pointer
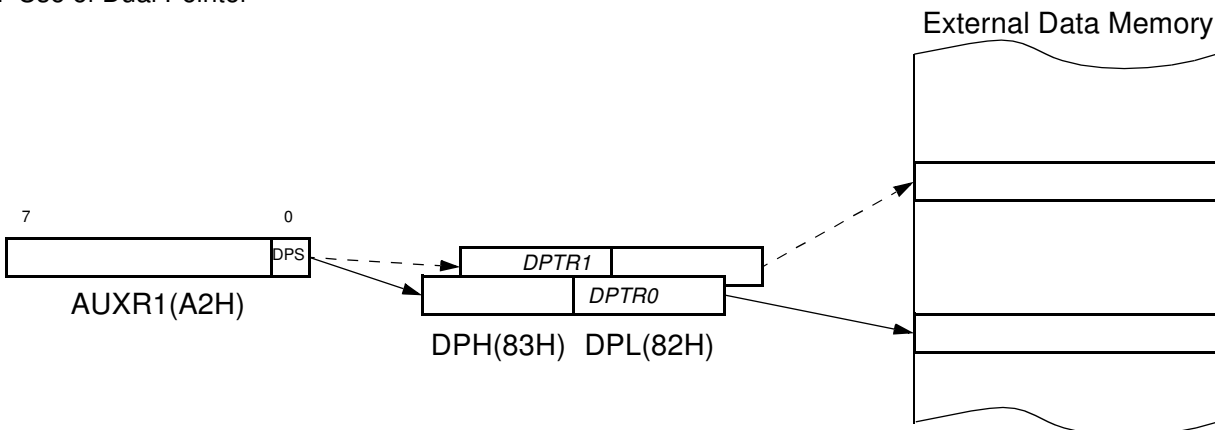


**Table 21.** AUXR1 register

AUXR1- Auxiliary Register 1(0A2h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | ENBOOT | - | GF3 | 0 | - | DPS |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 6 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 5 | ENBOOT | **Enable Boot Flash**<br>Cleared to disable boot rom.<br>Set to map the boot rom between F800h - 0FFFFh. |
| 4 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 3 | GF3 | This bit is a general purpose user flag.* |
| 2 | 0 | Always cleared. |
| 1 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 0 | DPS | **Data Pointer Selection**<br>Cleared to select DPTR0.<br>Set to select DPTR1. |

Reset Value: XXXX XX0X0b
Not bit addressable

Note:    *Bit 2 stuck at 0; this allows to use INC AUXR1 to toggle DPS without changing GF3.

ASSEMBLY LANGUAGE

```
; Block move using dual data pointers
; Modifies DPTR0, DPTR1, A and PSW
; note: DPS exits opposite of entry state
; unless an extra INC AUXR1 is added
;
00A2  AUXR1 EQU 0A2H
;
0000 909000MOV DPTR,#SOURCE ; address of SOURCE
0003 05A2 INC AUXR1 ; switch data pointers
0005 90A000 MOV DPTR,#DEST ; address of DEST
0008  LOOP:
0008 05A2 INC AUXR1 ; switch data pointers
000A E0 MOVX A,@DPTR ; get a byte from SOURCE
000B A3 INC DPTR ; increment SOURCE address
000C 05A2 INC AUXR1 ; switch data pointers
000E F0 MOVX @DPTR,A ; write the byte to DEST
000F A3 INC DPTR ; increment DEST address
0010 70F6JNZ LOOP ; check for 0 terminator
0012 05A2 INC AUXR1 ; (optional) restore DPS
```

INC is a short (2 bytes) and fast (12 clocks) way to manipulate the DPS bit in the AUXR1 SFR. However, note that the INC instruction does not directly force the DPS bit to a particular state, but simply toggles it. In simple routines, such as the block move example, only the fact that DPS is toggled in the proper sequence matters, not its actual value. In other words, the block move routine works the same whether DPS is '0' or '1' on entry. Observe that without the last instruction (INC AUXR1), the routine will exit with DPS in the opposite state.

## Expanded RAM (XRAM)

The AT89C51IC2 provides additional Bytes of random access memory (RAM) space for increased data parameter handling and high level language usage.

AT89C51IC2 devices have expanded RAM in external data space; maximum size and location are described in Table 22.

**Table 22.** Expanded RAM

|  | XRAM size | Address | |
|---|---|---|---|
|  |  | Start | End |
| AT89C51IC2 | 1024 | 00h | 3FFh |

The AT89C51IC2 has internal data memory that is mapped into four separate segments.

The four segments are:

1. The Lower 128 bytes of RAM (addresses 00h to 7Fh) are directly and indirectly addressable.

2. The Upper 128 bytes of RAM (addresses 80h to FFh) are indirectly addressable only.

3. The Special Function Registers, SFRs, (addresses 80h to FFh) are directly addressable only.

4. The expanded RAM bytes are indirectly accessed by MOVX instructions, and with the EXTRAM bit cleared in the AUXR register (see Table 22)

The lower 128 bytes can be accessed by either direct or indirect addressing. The Upper 128 bytes can be accessed by indirect addressing only. The Upper 128 bytes occupy the same address space as the SFR. That means they have the same address, but are physically separate from SFR space.

**Figure 8.** Internal and External Data Memory Address
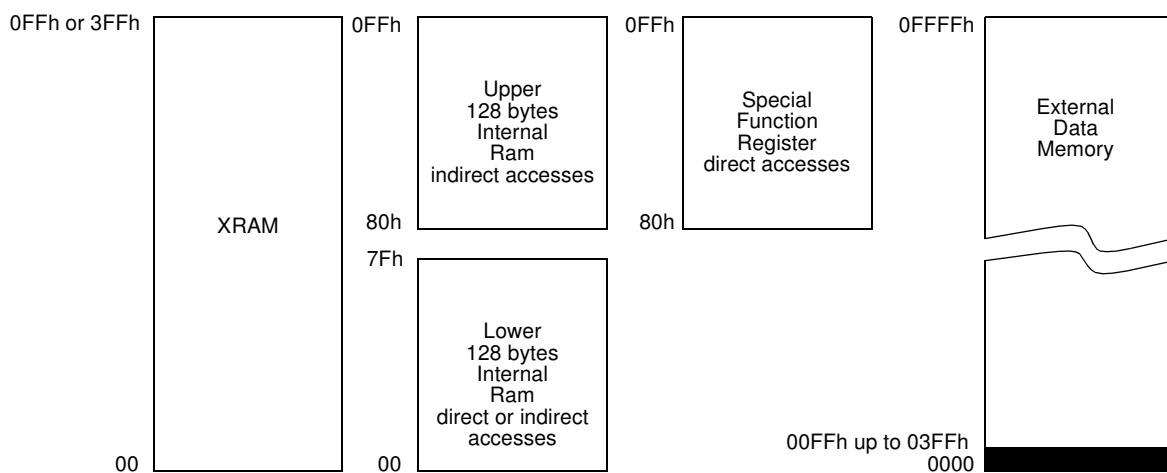


When an instruction accesses an internal location above address 7Fh, the CPU knows whether the access is to the upper 128 bytes of data RAM or to SFR space by the addressing mode used in the instruction.

• Instructions that use direct addressing access SFR space. For example: MOV 0A0H, # data, accesses the SFR at location 0A0h (which is P2).

**27**

- Instructions that use indirect addressing access the Upper 128 bytes of data RAM. For example: MOV @R0, # data where R0 contains 0A0h, accesses the data byte at address 0A0h, rather than P2 (whose address is 0A0h).

- The XRAM bytes can be accessed by indirect addressing, with EXTRAM bit cleared and MOVX instructions. This part of memory which is physically located on-chip, logically occupies the first bytes of external data memory. The bits XRS0 and XRS1 are used to hide a part of the available XRAM as explained in Table 22. This can be useful if external peripherals are mapped at addresses already used by the internal XRAM.

- With EXTRAM = 0, the XRAM is indirectly addressed, using the MOVX instruction in combination with any of the registers R0, R1 of the selected bank or DPTR. An access to XRAM will not affect ports P0, P2, P3.6 (WR) and P3.7 (RD). For example, with EXTRAM = 0, MOVX @R0, # data where R0 contains 0A0H, accesses the XRAM at address 0A0H rather than external memory. An access to external data memory locations higher than the accessible size of the XRAM will be performed with the MOVX DPTR instructions in the same way as in the standard 80C51, with P0 and P2 as data/address busses, and P3.6 and P3.7 as write and read timing signals. Accesses to XRAM above 0FFH can only be done by the use of DPTR.

- With EXTRAM = 1, MOVX @Ri and MOVX @DPTR will be similar to the standard 80C51. MOVX @ Ri will provide an eight-bit address multiplexed with data on Port0 and any output port pins can be used to output higher order address bits. This is to provide the external paging capability. MOVX @DPTR will generate a sixteen-bit address. Port2 outputs the high-order eight address bits (the contents of DPH) while Port0 multiplexes the low-order eight address bits (DPL) with data. MOVX @ Ri and MOVX @DPTR will generate either read or write signals on P3.6 (WR) and P3.7 (RD).

The stack pointer (SP) may be located anywhere in the 256 bytes RAM (lower and upper RAM) internal data memory. The stack may not be located in the XRAM.

The M0 bit allows to stretch the XRAM timings; if M0 is set, the read and write pulses are extended from 6 to 30 clock periods. This is useful to access external slow peripherals.

**Table 23.** AUXR Register

AUXR - Auxiliary Register (8Eh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | M0 | - | XRS1 | XRS0 | EXTRAM | AO |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 6 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 5 | M0 | **Pulse length**<br>Cleared to stretch MOVX control: the RD/ and the WR/ pulse length is 6 clock periods (default).<br>Set to stretch MOVX control: the RD/ and the WR/ pulse length is 30 clock periods. |
| 4 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 3 | XRS1 | **XRAM Size** |
| 2 | XRS0 | XRS1   XRS0   XRAM size<br>0       0       256 bytes (default)<br>0       1       512 bytes<br>1       0       768 bytes<br>1       1       1024 bytes |
| 1 | EXTRAM | **EXTRAM bit**<br>Cleared to access internal XRAM using movx @ Ri/ @ DPTR.<br>Set to access external memory.<br>Programmed by hardware after Power-up regarding Hardware Security Byte (HSB), default setting, XRAM selected. |
| 0 | AO | **ALE Output bit**<br>Cleared, ALE is emitted at a constant rate of 1/6 the oscillator frequency (or 1/3 if X2 mode is used) (default) Set, ALE is active only during a MOVX or MOVC instruction is used. |

Reset Value = XX0X 00'HSB.XRAM'0b
Not bit addressable

## Timer 2

The Timer 2 in the AT89C51IC2 is the standard C52 the Timer 2.
It is a 16-bit timer/counter: the count is maintained by two eight-bit timer registers, TH2 and TL2 are cascaded. It is controlled by T2CON (Table 24) and T2MOD (Table 25) registers. Timer 2 operation is similar to Timer 0 and Timer 1. C/$\overline{T2}$ selects $F_{OSC}$/12 (timer operation) or external pin T2 (counter operation) as the timer clock input. Setting TR2 allows TL2 to be incremented by the selected input.

Timer 2 has 3 operating modes: capture, autoreload and Baud Rate Generator. These modes are selected by the combination of RCLK, TCLK and CP/$\overline{RL2}$ (T2CON).

Refer to the Atmel 8-bit Microcontroller Hardware description for the description of Capture and Baud Rate Generator Modes.

Timer 2 includes the following enhancements:

•   Auto-reload mode with up or down counter
•   Programmable clock-output

## Auto-Reload Mode

The auto-reload mode configures timer 2 as a 16-bit timer or event counter with automatic reload. If DCEN bit in T2MOD is cleared, timer 2 behaves as in 80C52 (refer to the Atmel 8-bit Microcontroller Hardware description). If DCEN bit is set, timer 2 acts as an Up/down timer/counter as shown in Figure 9. In this mode the T2EX pin controls the direction of count.

When T2EX is high, timer 2 counts up. Timer overflow occurs at FFFFh which sets the TF2 flag and generates an interrupt request. The overflow also causes the 16-bit value in RCAP2H and RCAP2L registers to be loaded into the timer registers TH2 and TL2.

When T2EX is low, timer 2 counts down. Timer underflow occurs when the count in the timer registers TH2 and TL2 equals the value stored in RCAP2H and RCAP2L registers. The underflow sets TF2 flag and reloads FFFFh into the timer registers.

The EXF2 bit toggles when timer 2 overflows or underflows according to the direction of the count. EXF2 does not generate any interrupt. This bit can be used to provide 17-bit resolution.

**Figure 9.** Auto-Reload Mode Up/Down Counter (DCEN = 1)
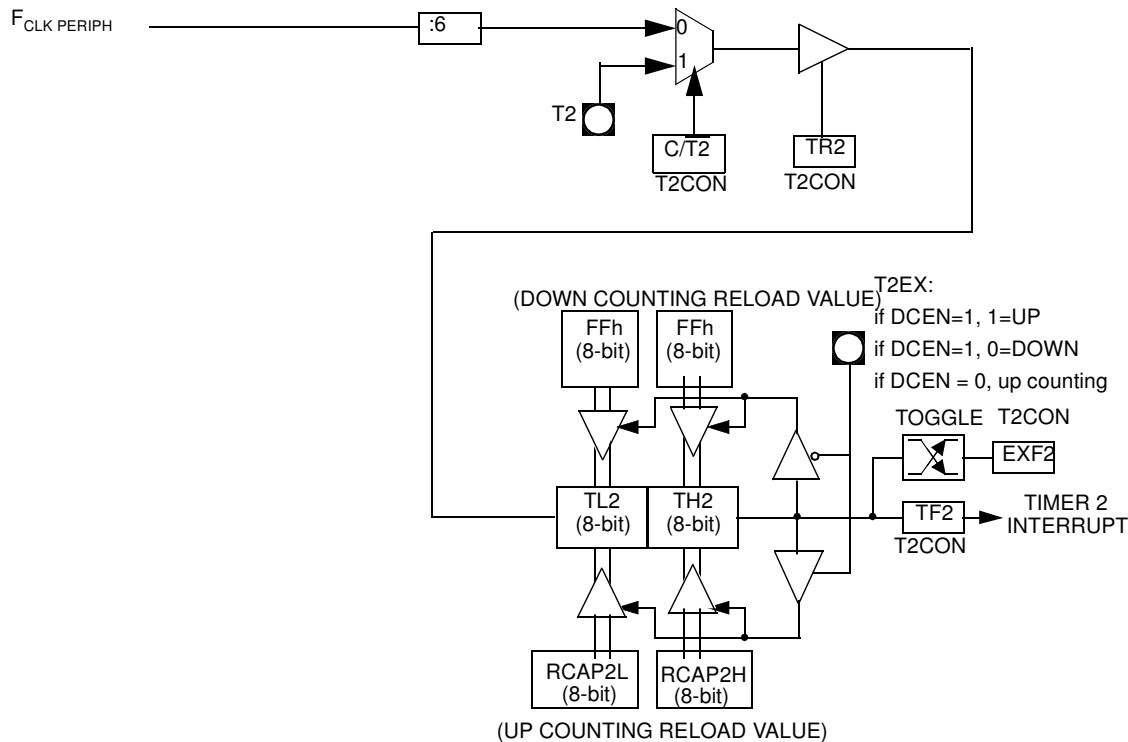


**Programmable Clock-Output**

In the clock-out mode, timer 2 operates as a 50%-duty-cycle, programmable clock generator (See Figure 10). The input clock increments TL2 at frequency $F_{CLK\ PERIPH}/2$. The timer repeatedly counts to overflow from a loaded value. At overflow, the contents of RCAP2H and RCAP2L registers are loaded into TH2 and TL2. In this mode, timer 2 overflows do not generate interrupts. The formula gives the clock-out frequency as a function of the system oscillator frequency and the value in the RCAP2H and RCAP2L registers:

$$Clock-OutFrequency = \frac{F_{CLKPERIPH}}{4 \times (65536 - RCAP2H/RCAP2L)}$$

For a 16 MHz system clock, timer 2 has a programmable frequency range of 61 Hz ($F_{CLK\ PERIPH}/2^{16)}$) to 4 MHz ($F_{CLK\ PERIPH}/4$). The generated clock signal is brought out to T2 pin (P1.0).

Timer 2 is programmed for the clock-out mode as follows:

- Set T2OE bit in T2MOD register.
- Clear C/$\overline{T2}$ bit in T2CON register.
- Determine the 16-bit reload value from the formula and enter it in RCAP2H/RCAP2L registers.
- Enter a 16-bit initial value in timer registers TH2/TL2. It can be the same as the reload value or a different one depending on the application.
- To start the timer, set TR2 run control bit in T2CON register.

It is possible to use timer 2 as a baud rate generator and a clock generator simultaneously. For this configuration, the baud rates and clock frequencies are not independent since both functions use the values in the RCAP2H and RCAP2L registers.

**Figure 10.** Clock-Out Mode C/$\overline{T2}$ = 0
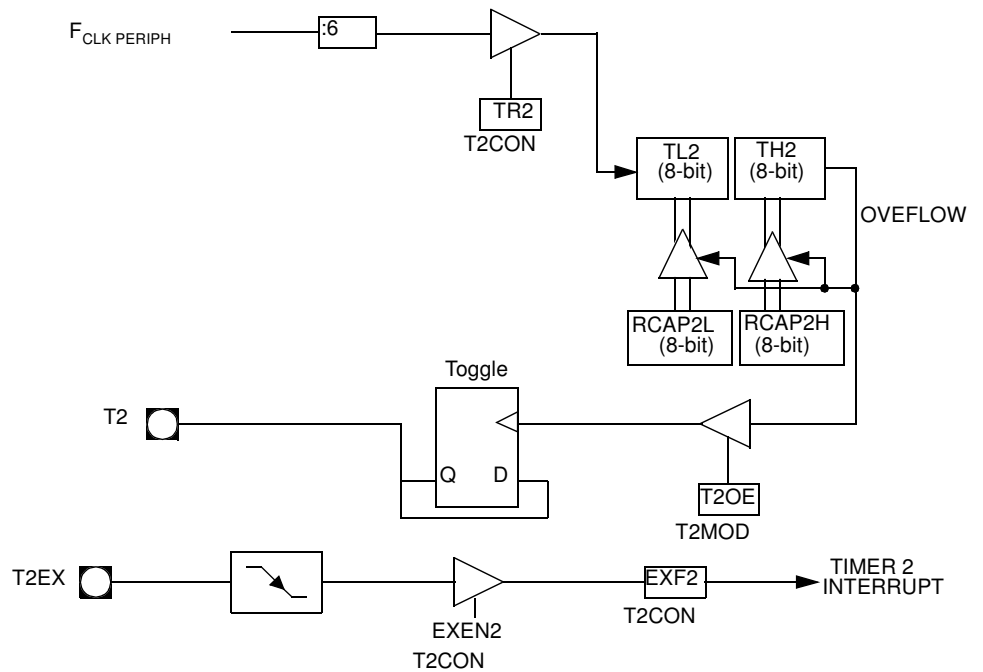
**Table 24.** T2CON Register

T2CON - Timer 2 Control Register (C8h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TF2 | EXF2 | RCLK | TCLK | EXEN2 | TR2 | C/T2# | CP/RL2# |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | TF2 | **Timer 2 overflow Flag**<br>Must be cleared by software.<br>Set by hardware on timer 2 overflow, if RCLK = 0 and TCLK = 0. |
| 6 | EXF2 | **Timer 2 External Flag**<br>Set when a capture or a reload is caused by a negative transition on T2EX pin if EXEN2=1.<br>When set, causes the CPU to vector to timer 2 interrupt routine when timer 2 interrupt is enabled.<br>Must be cleared by software. EXF2 doesn't cause an interrupt in Up/down counter mode (DCEN = 1) |
| 5 | RCLK | **Receive Clock bit**<br>Cleared to use timer 1 overflow as receive clock for serial port in mode 1 or 3.<br>Set to use timer 2 overflow as receive clock for serial port in mode 1 or 3. |
| 4 | TCLK | **Transmit Clock bit**<br>Cleared to use timer 1 overflow as transmit clock for serial port in mode 1 or 3.<br>Set to use timer 2 overflow as transmit clock for serial port in mode 1 or 3. |
| 3 | EXEN2 | **Timer 2 External Enable bit**<br>Cleared to ignore events on T2EX pin for timer 2 operation.<br>Set to cause a capture or reload when a negative transition on T2EX pin is detected, if timer 2 is not used to clock the serial port. |
| 2 | TR2 | **Timer 2 Run control bit**<br>Cleared to turn off timer 2.<br>Set to turn on timer 2. |
| 1 | C/T2# | **Timer/Counter 2 select bit**<br>Cleared for timer operation (input from internal clock system: $F_{CLK\ PERIPH}$).<br>Set for counter operation (input from T2 input pin, falling edge trigger). Must be 0 for clock out mode. |
| 0 | CP/RL2# | **Timer 2 Capture/Reload bit**<br>If RCLK=1 or TCLK=1, CP/RL2# is ignored and timer is forced to auto-reload on timer 2 overflow.<br>Cleared to auto-reload on timer 2 overflows or negative transitions on T2EX pin if EXEN2=1.<br>Set to capture on negative transitions on T2EX pin if EXEN2=1. |

Reset Value = 0000 0000b
Bit addressable

**Table 25.** T2MOD Register

T2MOD - Timer 2 Mode Control Register (C9h)

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

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 6 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 3 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 2 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 1 | T2OE | **Timer 2 Output Enable bit**<br>Cleared to program P1.0/T2 as clock input or I/O port.<br>Set to program P1.0/T2 as clock output. |
| 0 | DCEN | **Down Counter Enable bit**<br>Cleared to disable timer 2 as up/down counter.<br>Set to enable timer 2 as up/down counter. |

Reset Value = XXXX XX00b
Not bit addressable

## Programmable Counter Array PCA

The PCA provides more timing capabilities with less CPU intervention than the standard timer/counters. Its advantages include reduced software overhead and improved accuracy. The PCA consists of a dedicated timer/counter which serves as the time base for an array of five compare/capture modules. Its clock input can be programmed to count any one of the following signals:

- Peripheral clock frequency ($F_{CLK\ PERIPH}$) $\div$ 6
- Peripheral clock frequency ($F_{CLK\ PERIPH}$) $\div$ 2
- Timer 0 overflow
- External input on ECI (P1.2)

Each compare/capture modules can be programmed in any one of the following modes:

- rising and/or falling edge capture
- software timer
- high-speed output
- pulse width modulator

Module 4 can also be programmed as a watchdog timer (See Section "PCA Watchdog Timer", page 46).

When the compare/capture modules are programmed in the capture mode, software timer, or high speed output mode, an interrupt can be generated when the module executes its function. All five modules plus the PCA timer overflow share one interrupt vector.

The PCA timer/counter and compare/capture modules share Port 1 for external I/O. These pins are listed below. If the port is not used for the PCA, it can still be used for standard I/O.

| PCA component | External I/O Pin |
|---|---|
| 16-bit Counter | P1.2 / ECI |
| 16-bit Module 0 | P1.3 / CEX0 |
| 16-bit Module 1 | P1.4 / CEX1 |
| 16-bit Module 2 | P1.5 / CEX2 |
| 16-bit Module 3 | P1.6 / CEX3 |

The PCA timer is a common time base for all five modules (See Figure 11). The timer count source is determined from the CPS1 and CPS0 bits in the CMOD register (Table 26) and can be programmed to run at:

- 1/6 the peripheral clock frequency ($F_{CLK\ PERIPH}$)
- 1/2 the peripheral clock frequency ($F_{CLK\ PERIPH}$)
- The Timer 0 overflow
- The input on the ECI pin (P1.2)

**Figure 11.** PCA Timer/Counter
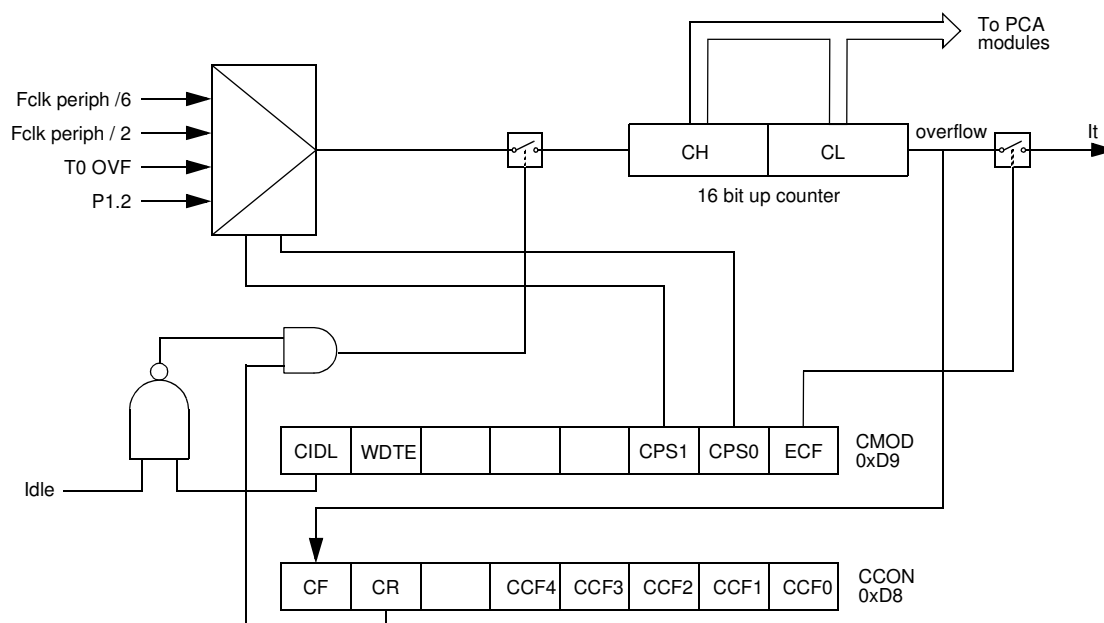
**Table 26.** CMOD Register

CMOD - PCA Counter Mode Register (D9h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CIDL | WDTE | - | - | - | CPS1 | CPS0 | ECF |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | CIDL | **Counter Idle Control**<br>Cleared to program the PCA Counter to continue functioning during idle Mode.<br>Set to program PCA to be gated off during idle. |
| 6 | WDTE | **Watchdog Timer Enable**<br>Cleared to disable Watchdog Timer function on PCA Module 4.<br>Set to enable Watchdog Timer function on PCA Module 4. |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 3 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 2 | CPS1 | **PCA Count Pulse Select** |
| 1 | CPS0 | CPS1    CPS0Selected PCA input<br>0         0 Internal clock fCLK PERIPH/6<br>0         1Internal clock fCLK PERIPH/2<br>1         0Timer 0 Overflow<br>1         1 External clock at ECI/P1.2 pin (max rate = fCLK PERIPH/ 4) |
| 0 | ECF | **PCA Enable Counter Overflow Interrupt**<br>Cleared to disable CF bit in CCON to inhibit an interrupt.<br>Set to enable CF bit in CCON to generate an interrupt. |

Reset Value = 00XX X000b
Not bit addressable

The CMOD register includes three additional bits associated with the PCA (See Figure 11 and Table 26).

• The CIDL bit which allows the PCA to stop during idle mode.

• The WDTE bit which enables or disables the watchdog function on module 4.

• The ECF bit which when set causes an interrupt and the PCA overflow flag CF (in the CCON SFR) to be set when the PCA timer overflows.

The CCON register contains the run control bit for the PCA and the flags for the PCA timer (CF) and each module (Refer to Table 27).

• Bit CR (CCON.6) must be set by software to run the PCA. The PCA is shut off by clearing this bit.

• Bit CF: The CF bit (CCON.7) is set when the PCA counter overflows and an interrupt will be generated if the ECF bit in the CMOD register is set. The CF bit can only be cleared by software.

• Bits 0 through 4 are the flags for the modules (bit 0 for module 0, bit 1 for module 1, etc.) and are set by hardware when either a match or a capture occurs. These flags also can only be cleared by software.

**Table 27.** CCON Register

CCON - PCA Counter Control Register (D8h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CF | **CR** | - | **CCF4** | **CCF3** | **CCF2** | **CCF1** | **CCF0** |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | CF | **PCA Counter Overflow flag**<br>Set by hardware when the counter rolls over. CF flags an interrupt if bit ECF in CMOD is set. CF<br>may be set by either hardware or software but can only be cleared by software. |
| 6 | CR | **PCA Counter Run control bit**<br>Must be cleared by software to turn the PCA counter off.<br>Set by software to turn the PCA counter on. |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | CCF4 | **PCA Module 4 interrupt flag**<br>Must be cleared by software.<br>Set by hardware when a match or capture occurs. |
| 3 | CCF3 | **PCA Module 3 interrupt flag**<br>Must be cleared by software.<br>Set by hardware when a match or capture occurs. |
| 2 | CCF2 | **PCA Module 2 interrupt flag**<br>Must be cleared by software.<br>Set by hardware when a match or capture occurs. |
| 1 | CCF1 | **PCA Module 1 interrupt flag**<br>Must be cleared by software.<br>Set by hardware when a match or capture occurs. |
| 0 | CCF0 | **PCA Module 0 interrupt flag**<br>Must be cleared by software.<br>Set by hardware when a match or capture occurs. |

Reset Value = 00X0 0000b
Not bit addressable

The watchdog timer function is implemented in module 4 (See Figure 14).

The PCA interrupt system is shown in Figure 12.

**Figure 12.** PCA Interrupt System
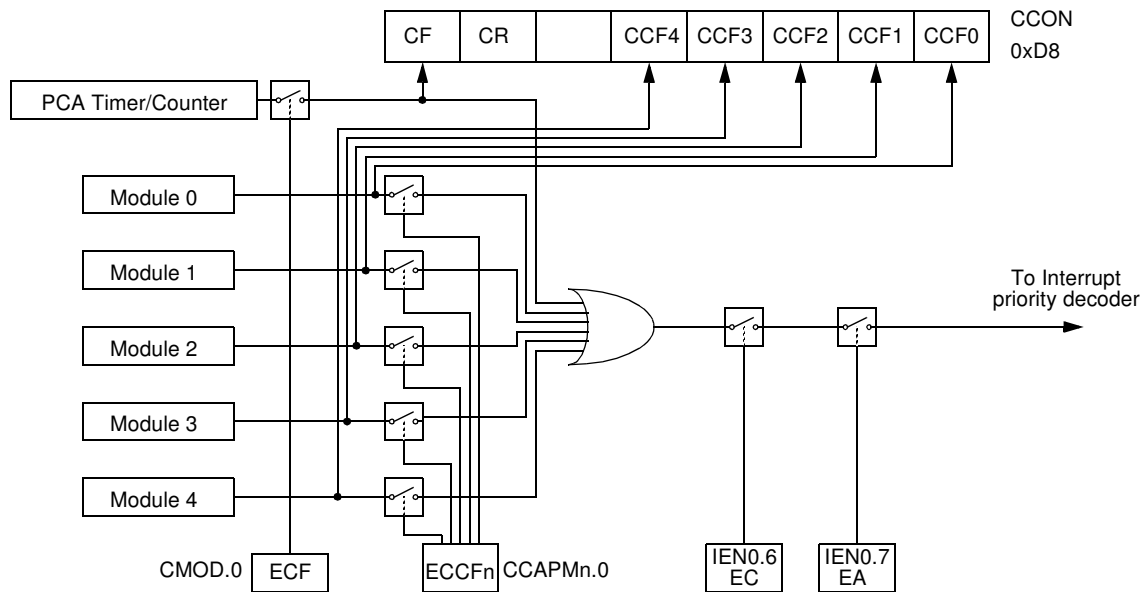


**PCA Modules:** each one of the five compare/capture modules has six possible functions. It can perform:

- 16-bit Capture, positive-edge triggered
- 16-bit Capture, negative-edge triggered
- 16-bit Capture, both positive and negative-edge triggered
- 16-bit Software Timer
- 16-bit High Speed Output
- 8-bit Pulse Width Modulator

In addition, module 4 can be used as a Watchdog Timer.

Each module in the PCA has a special function register associated with it. These registers are: CCAPM0 for module 0, CCAPM1 for module 1, etc. (See Table 28). The registers contain the bits that control the mode that each module will operate in.

- The ECCF bit (CCAPMn.0 where n=0, 1, 2, 3, or 4 depending on the module) enables the CCF flag in the CCON SFR to generate an interrupt when a match or compare occurs in the associated module.
- PWM (CCAPMn.1) enables the pulse width modulation mode.
- The TOG bit (CCAPMn.2) when set causes the CEX output associated with the module to toggle when there is a match between the PCA counter and the module's capture/compare register.
- The match bit MAT (CCAPMn.3) when set will cause the CCFn bit in the CCON register to be set when there is a match between the PCA counter and the module's capture/compare register.
- The next two bits CAPN (CCAPMn.4) and CAPP (CCAPMn.5) determine the edge that a capture input will be active on. The CAPN bit enables the negative edge, and the CAPP bit enables the positive edge. If both bits are set both edges will be enabled and a capture will occur for either transition.
- The last bit in the register ECOM (CCAPMn.6) when set enables the comparator function.

Table 28 shows the CCAPMn settings for the various PCA functions.

**Table 28.** CCAPMn Registers (n = 0-4)

CCAPM0 - PCA Module 0 Compare/Capture Control Register (0DAh)

CCAPM1 - PCA Module 1 Compare/Capture Control Register (0DBh)

CCAPM2 - PCA Module 2 Compare/Capture Control Register (0DCh)

CCAPM3 - PCA Module 3 Compare/Capture Control Register (0DDh)

CCAPM4 - PCA Module 4 Compare/Capture Control Register (0DEh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | ECOMn | CAPPn | CAPNn | MATn | TOGn | PWMn | ECCFn |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 6 | ECOMn | **Enable Comparator**<br>Cleared to disable the comparator function.<br>Set to enable the comparator function. |
| 5 | CAPPn | **Capture Positive**<br>Cleared to disable positive edge capture.<br>Set to enable positive edge capture. |
| 4 | CAPNn | **Capture Negative**<br>Cleared to disable negative edge capture.<br>Set to enable negative edge capture. |
| 3 | MATn | **Match**<br>When MATn = 1, a match of the PCA counter with this module's compare/capture register causes the<br>CCFn bit in CCON to be set, flagging an interrupt. |
| 2 | TOGn | **Toggle**<br>When TOGn = 1, a match of the PCA counter with this module's compare/capture register causes the<br>CEXn pin to toggle. |
| 1 | PWMn | **Pulse Width Modulation Mode**<br>Cleared to disable the CEXn pin to be used as a pulse width modulated output.<br>Set to enable the CEXn pin to be used as a pulse width modulated output. |
| 0 | CCF0 | **Enable CCF interrupt**<br>Cleared to disable compare/capture flag CCFn in the CCON register to generate an interrupt.<br>Set to enable compare/capture flag CCFn in the CCON register to generate an interrupt. |

Reset Value = X000 0000b
Not bit addressable

**Table 29.** PCA Module Modes (CCAPMn Registers)

| ECOMn | CAPPn | CAPNn | MATn | TOGn | PWMm | ECCFn | Module Function |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | No Operation |
| X | 1 | 0 | 0 | 0 | 0 | X | 16-bit capture by a positive-edge trigger on CEXn |
| X | 0 | 1 | 0 | 0 | 0 | X | 16-bit capture by a negative trigger on CEXn |
| X | 1 | 1 | 0 | 0 | 0 | X | 16-bit capture by a transition on CEXn |
| 1 | 0 | 0 | 1 | 0 | 0 | X | 16-bit Software Timer / Compare mode. |
| 1 | 0 | 0 | 1 | 1 | 0 | X | 16-bit High Speed Output |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 8-bit PWM |
| 1 | 0 | 0 | 1 | X | 0 | X | Watchdog Timer (module 4 only) |

There are two additional registers associated with each of the PCA modules. They are CCAPnH and CCAPnL and these are the registers that store the 16-bit count when a capture occurs or a compare should occur. When a module is used in the PWM mode these registers are used to control the duty cycle of the output (See Table 30 & Table 31).

**Table 30.** CCAPnH Registers (n = 0-4)

CCAP0H - PCA Module 0 Compare/Capture Control Register High (0FAh)

CCAP1H - PCA Module 1 Compare/Capture Control Register High (0FBh)

CCAP2H - PCA Module 2 Compare/Capture Control Register High (0FCh)

CCAP3H - PCA Module 3 Compare/Capture Control Register High (0FDh)

CCAP4H - PCA Module 4 Compare/Capture Control Register High (0FEh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7-0 | - | **PCA Module n Compare/Capture Control**<br>CCAPnH Value |

Reset Value = 0000 0000b
Not bit addressable

**Table 31.** CCAPnL Registers (n = 0-4)

CCAP0L - PCA Module 0 Compare/Capture Control Register Low (0EAh)

CCAP1L - PCA Module 1 Compare/Capture Control Register Low (0EBh)

CCAP2L - PCA Module 2 Compare/Capture Control Register Low (0ECh)

CCAP3L - PCA Module 3 Compare/Capture Control Register Low (0EDh)

CCAP4L - PCA Module 4 Compare/Capture Control Register Low (0EEh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7-0 | - | **PCA Module n Compare/Capture Control**<br>CCAPnL Value |

Reset Value = 0000 0000b
Not bit addressable

**Table 32.** CH Register

CH - PCA Counter Register High (0F9h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7-0 | - | **PCA counter**<br>CH Value |

Reset Value = 0000 0000b
Not bit addressable

**Table 33.** CL Register

CL - PCA Counter Register Low (0E9h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7-0 | - | **PCA Counter**<br>CL Value |

Reset Value = 0000 0000b
Not bit addressable

**PCA Capture Mode**

To use one of the PCA modules in the capture mode either one or both of the CCAPM bits CAPN and CAPP for that module must be set. The external CEX input for the module (on port 1) is sampled for a transition. When a valid transition occurs the PCA hardware loads the value of the PCA counter registers (CH and CL) into the module's capture registers (CCAPnL and CCAPnH). If the CCFn bit for the module in the CCON SFR and the ECCFn bit in the CCAPMn SFR are set then an interrupt will be generated (Refer to Figure 13).

**Figure 13.** PCA Capture Mode
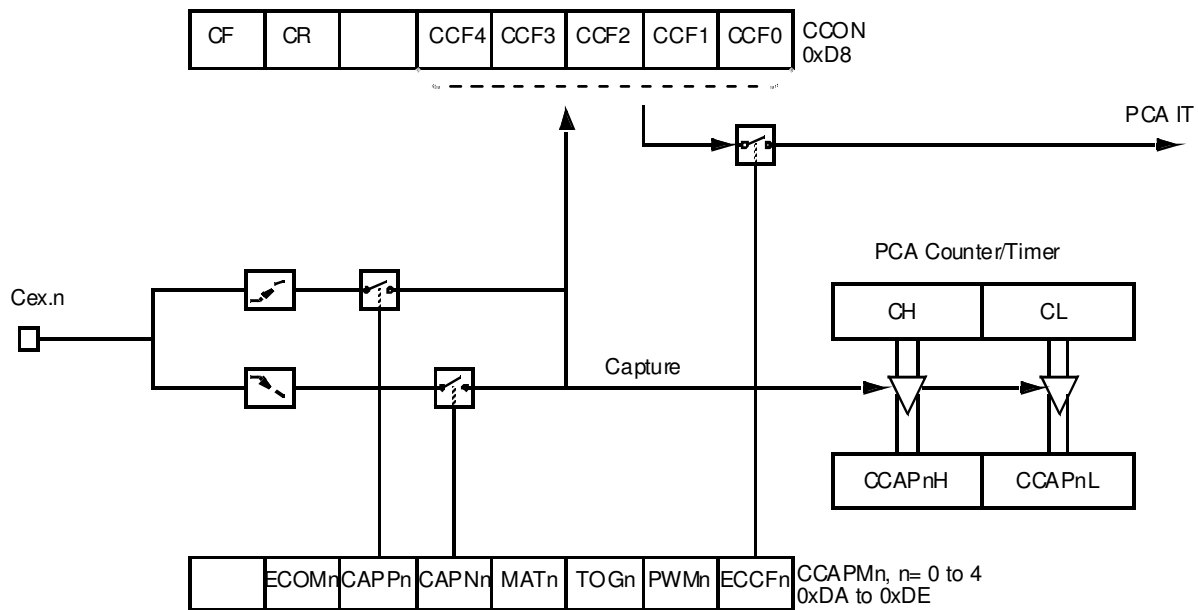


**16-bit Software Timer/ Compare Mode**

The PCA modules can be used as software timers by setting both the ECOM and MAT bits in the modules CCAPMn register. The PCA timer will be compared to the module's capture registers and when a match occurs an interrupt will occur if the CCFn (CCON SFR) and the ECCFn (CCAPMn SFR) bits for the module are both set (See Figure 14).

**Figure 14.** PCA Compare Mode and PCA Watchdog Timer
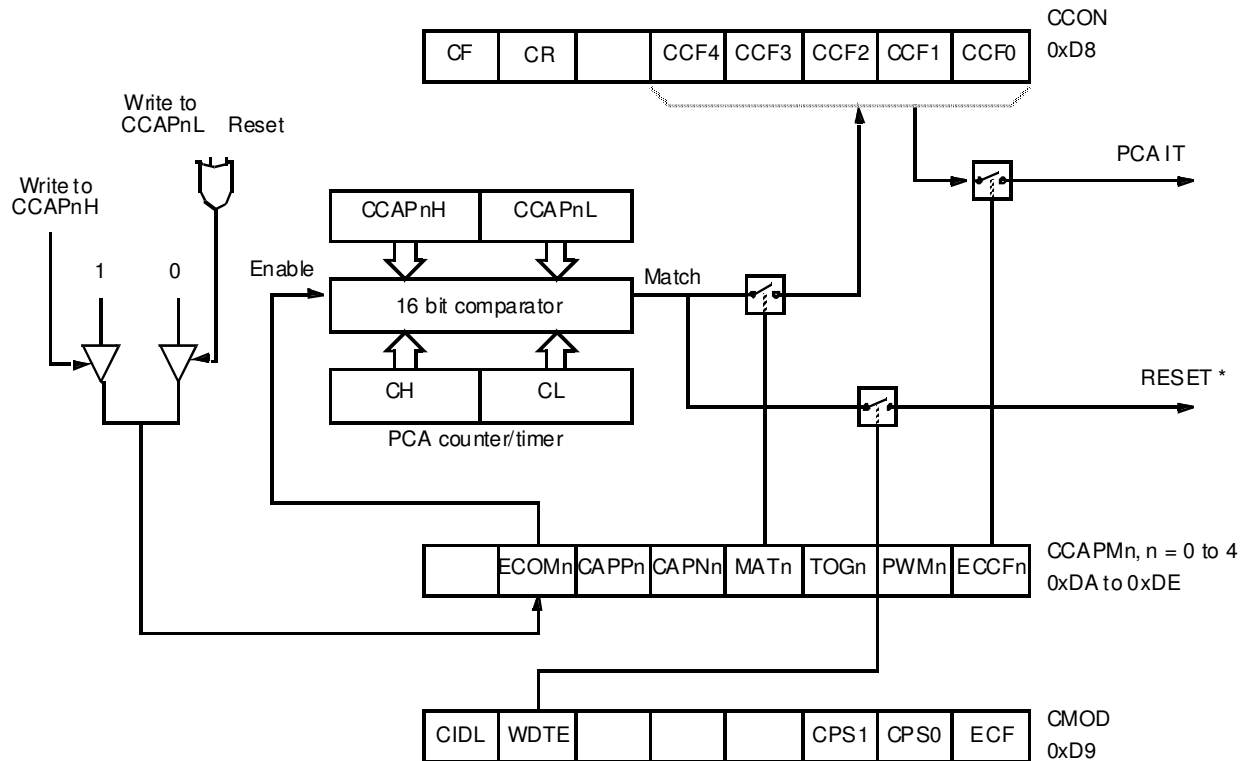


Before enabling ECOM bit, CCAPnL and CCAPnH should be set with a non zero value, otherwise an unwanted match could happen. Writing to CCAPnH will set the ECOM bit.

Once ECOM set, writing CCAPnL will clear ECOM so that an unwanted match doesn't occur while modifying the compare value. Writing to CCAPnH will set ECOM. For this reason, user software should write CCAPnL first, and then CCAPnH. Of course, the ECOM bit can still be controlled by accessing to CCAPMn register.

**High Speed Output Mode**    In this mode the CEX output (on port 1) associated with the PCA module will toggle each time a match occurs between the PCA counter and the module's capture registers. To activate this mode the TOG, MAT, and ECOM bits in the module's CCAPMn SFR must be set (See Figure 15).

A prior write must be done to CCAPnL and CCAPnH before writing the ECOMn bit.

**Figure 15.** PCA High Speed Output Mode
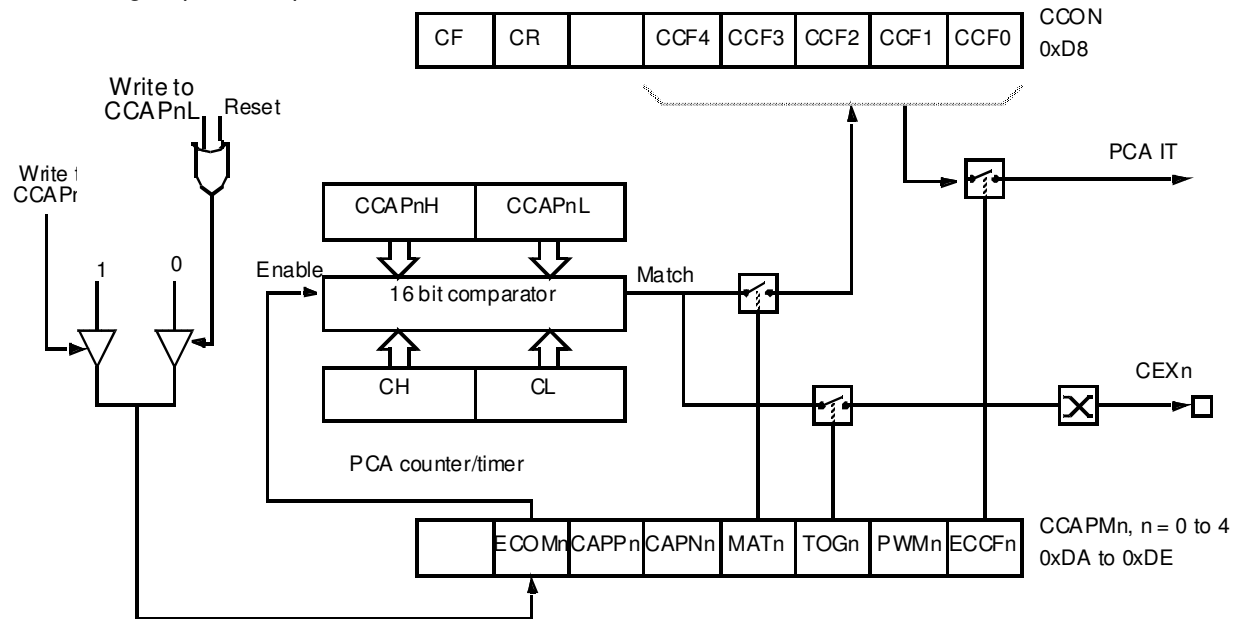


Before enabling ECOM bit, CCAPnL and CCAPnH should be set with a non zero value, otherwise an unwanted match could happen.

Once ECOM set, writing CCAPnL will clear ECOM so that an unwanted match doesn't occur while modifying the compare value. Writing to CCAPnH will set ECOM. For this reason, user software should write CCAPnL first, and then CCAPnH. Of course, the ECOM bit can still be controlled by accessing to CCAPMn register.

**Pulse Width Modulator Mode**

All of the PCA modules can be used as PWM outputs. Figure 16 shows the PWM function. The frequency of the output depends on the source for the PCA timer. All of the modules will have the same frequency of output because they all share the PCA timer. The duty cycle of each module is independently variable using the module's capture register CCAPLn. When the value of the PCA CL SFR is less than the value in the module's CCAPLn SFR the output will be low, when it is equal to or greater than the output will be high. When CL overflows from FF to 00, CCAPLn is reloaded with the value in CCAPHn. This allows updating the PWM without glitches. The PWM and ECOM bits in the module's CCAPMn register must be set to enable the PWM mode.

**Figure 16.** PCA PWM Mode
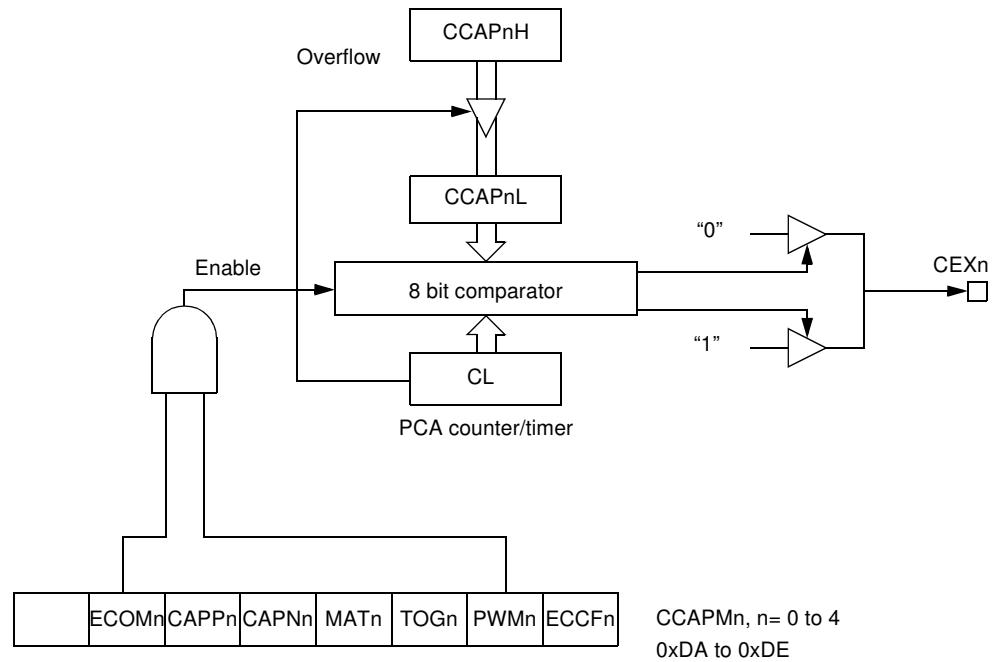


**PCA Watchdog Timer**

An on-board watchdog timer is available with the PCA to improve the reliability of the system without increasing chip count. Watchdog timers are useful for systems that are susceptible to noise, power glitches, or electrostatic discharge. Module 4 is the only PCA module that can be programmed as a watchdog. However, this module can still be used for other modes if the watchdog is not needed. Figure 14 shows a diagram of how the watchdog works. The user pre-loads a 16-bit value in the compare registers. Just like the other compare modes, this 16-bit value is compared to the PCA timer value. If a match is allowed to occur, an internal reset will be generated. This will not cause the RST pin to be driven high.

In order to hold off the reset, the user has three options:

1. periodically change the compare value so it will never match the PCA timer,

2. periodically change the PCA timer value so it will never match the compare values, or

3. disable the watchdog by clearing the WDTE bit before a match occurs and then re-enable it.

The first two options are more reliable because the watchdog timer is never disabled as in option #3. If the program counter ever goes astray, a match will eventually occur and cause an internal reset. The second option is also not recommended if other PCA modules are being used. Remember, the PCA timer is the time base for all modules; changing the time base for other modules would not be a good idea. Thus, in most applications the first solution is the best option.

This watchdog timer won't generate a reset out on the reset pin.

## Serial I/O Port

The serial I/O port in the AT89C51IC2 is compatible with the serial I/O port in the 80C52. It provides both synchronous and asynchronous communication modes. It operates as a Universal Asynchronous Receiver and Transmitter (UART) in three full-duplex modes (Modes 1, 2 and 3). Asynchronous transmission and reception can occur simultaneously and at different baud rates

Serial I/O port includes the following enhancements:

- Framing error detection
- Automatic address recognition

## Framing Error Detection

Framing bit error detection is provided for the three asynchronous modes (modes 1, 2 and 3). To enable the framing bit error detection feature, set SMOD0 bit in PCON register (See Figure 17).

**Figure 17.** Framing Error Block Diagram
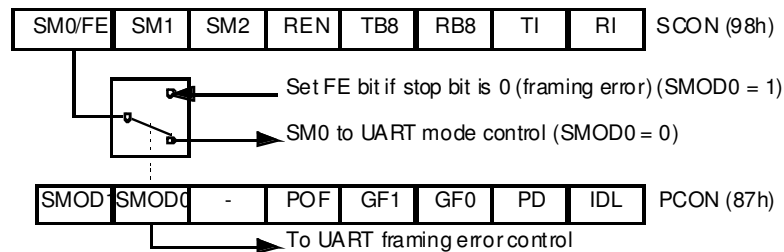


When this feature is enabled, the receiver checks each incoming data frame for a valid stop bit. An invalid stop bit may result from noise on the serial lines or from simultaneous transmission by two CPUs. If a valid stop bit is not found, the Framing Error bit (FE) in SCON register (See Table 37.) bit is set.

Software may examine FE bit after each reception to check for data errors. Once set, only software or a reset can clear FE bit. Subsequently received frames with valid stop bits cannot clear FE bit. When FE feature is enabled, RI rises on stop bit instead of the last data bit (See Figure 18. and Figure 19.).

**Figure 18.** UART Timings in Mode 1
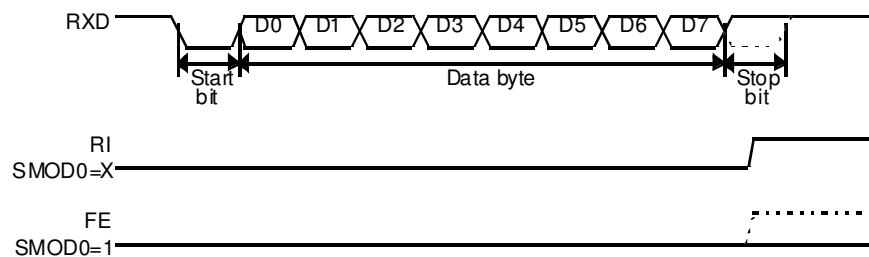
**Figure 19.** UART Timings in Modes 2 and 3
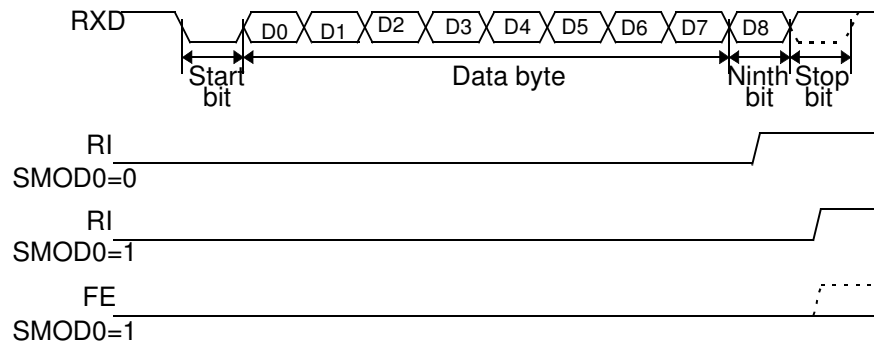
**Automatic Address Recognition**

The automatic address recognition feature is enabled when the multiprocessor communication feature is enabled (SM2 bit in SCON register is set).

Implemented in hardware, automatic address recognition enhances the multiprocessor communication feature by allowing the serial port to examine the address of each incoming command frame. Only when the serial port recognizes its own address, the receiver sets RI bit in SCON register to generate an interrupt. This ensures that the CPU is not interrupted by command frames addressed to other devices.

If desired, you may enable the automatic address recognition feature in mode 1. In this configuration, the stop bit takes the place of the ninth data bit. Bit RI is set only when the received command frame address matches the device's address and is terminated by a valid stop bit.

To support automatic address recognition, a device is identified by a given address and a broadcast address.

Note: The multiprocessor communication and automatic address recognition features cannot be enabled in mode 0 (i.e. setting SM2 bit in SCON register in mode 0 has no effect).

**Given Address**

Each device has an individual address that is specified in SADDR register; the SADEN register is a mask byte that contains don't-care bits (defined by zeros) to form the device's given address. The don't-care bits provide the flexibility to address one or more slaves at a time. The following example illustrates how a given address is formed.

To address a device by its individual address, the SADEN mask byte must be `1111 1111b`.

For example:

```
SADDR    0101 0110b
SADEN    1111 1100b
Given    0101 01XXb
```

The following is an example of how to use given addresses to address different slaves:

```
Slave A:  SADDR    1111 0001b
          SADEN    1111 1010b
          Given    1111 0X0Xb


Slave B:  SADDR    1111 0011b
          SADEN    1111 1001b
          Given    1111 0XX1b


Slave C:  SADDR    1111 0010b
          SADEN    1111 1101b
          Given    1111 00X1b
```

The SADEN byte is selected so that each slave may be addressed separately.

For slave A, bit 0 (the LSB) is a don't-care bit; for slaves B and C, bit 0 is a 1. To communicate with slave A only, the master must send an address where bit 0 is clear (e.g. `1111 0000b`).

For slave A, bit 1 is a 1; for slaves B and C, bit 1 is a don't care bit. To communicate with slaves B and C, but not slave A, the master must send an address with bits 0 and 1 both set (e.g. `1111 0011b`).

To communicate with slaves A, B and C, the master must send an address with bit 0 set, bit 1 clear, and bit 2 clear (e.g. `1111 0001b`).

**Broadcast Address**

A broadcast address is formed from the logical OR of the SADDR and SADEN registers with zeros defined as don't-care bits, e.g.:

```
    SADDR 0101 0110b
    SADEN 1111 1100b
 Broadcast =SADDR OR SADEN1111 111Xb
```

The use of don't-care bits provides flexibility in defining the broadcast address, however in most applications, a broadcast address is FFh. The following is an example of using broadcast addresses:

```
Slave A:SADDR1111 0001b
    SADEN1111 1010b
    Broadcast1111 1X11b,

Slave B:SADDR1111 0011b
    SADEN1111 1001b
    Broadcast1111 1X11B,

Slave C:SADDR=1111 0010b
    SADEN1111 1101b
    Broadcast1111 1111b
```

For slaves A and B, bit 2 is a don't care bit; for slave C, bit 2 is set. To communicate with all of the slaves, the master must send an address FFh. To communicate with slaves A and B, but not slave C, the master can send and address FBh.

**Reset Addresses**

On reset, the SADDR and SADEN registers are initialized to 00h, i.e. the given and broadcast addresses are `XXXX XXXXb` (all don't-care bits). This ensures that the serial port will reply to any address, and so, that it is backwards compatible with the 80C51 microcontrollers that do not support automatic address recognition.

**Table 34.** SADEN Register

SADEN - Slave Address Mask Register (B9h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|   |   |   |   |   |   |   |   |

Reset Value = 0000 0000b
Not bit addressable

**Table 35.** SADDR Register

SADDR - Slave Address Register (A9h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|   |   |   |   |   |   |   |   |

Reset Value = 0000 0000b
Not bit addressable

## Baud Rate Selection for UART for mode 1 and 3

The Baud Rate Generator for transmit and receive clocks can be selected separately via the T2CON and BDRCON registers.

**Figure 20.** Baud Rate selection
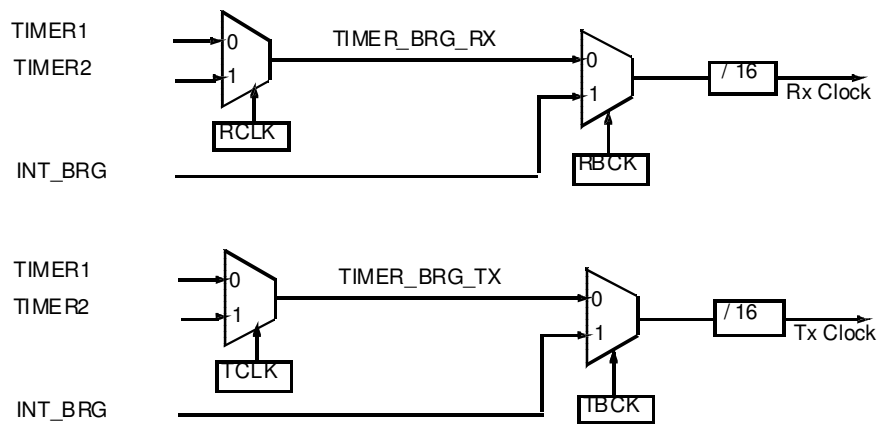


**Table 36.** Baud Rate Selection Table UART

| TCLK (T2CON) | RCLK (T2CON) | TBCK (BDRCON) | RBCK (BDRCON) | Clock Source UART Tx | Clock Source UART Rx |
|:---:|:---:|:---:|:---:|:---:|:---:|
| 0 | 0 | 0 | 0 | Timer 1 | Timer 1 |
| 1 | 0 | 0 | 0 | Timer 2 | Timer 1 |
| 0 | 1 | 0 | 0 | Timer 1 | Timer 2 |
| 1 | 1 | 0 | 0 | Timer 2 | Timer 2 |
| X | 0 | 1 | 0 | INT_BRG | Timer 1 |
| X | 1 | 1 | 0 | INT_BRG | Timer 2 |
| 0 | X | 0 | 1 | Timer 1 | INT_BRG |
| 1 | X | 0 | 1 | Timer 2 | INT_BRG |
| X | X | 1 | 1 | INT_BRG | INT_BRG |

**Internal Baud Rate Generator (BRG)**

When the internal Baud Rate Generator is used, the Baud Rates are determined by the BRG overflow depending on the BRL reload value, the value of SPD bit (Speed Mode) in BDRCON register and the value of the SMOD1 bit in PCON register.

**Figure 21.** Internal Baud Rate
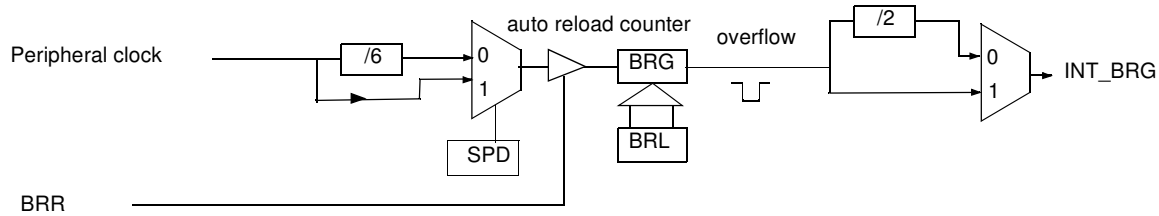


- The baud rate for UART is token by formula:

$$BaudRate = \frac{2_{SMOD} \times F_{CLKPERIPH}}{2 \times 2 \times 6 \langle 1 - SPD \rangle \times 16 \times [256 - (BRL)]}$$

$$(BRL) = 256 - \frac{2_{SMOD1} \times F_{CLKPERIPH}}{2 \times 2 \times 6_{(1-SPD)} \times 16 \times BaudRate}$$

**Table 37.** SCON Register

SCON - Serial Control Register (98h)

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

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | FE | **Framing Error bit (SMOD0=1)**<br>Clear to reset the error state, not cleared by a valid stop bit.<br>Set by hardware when an invalid stop bit is detected.<br>SMOD0 must be set to enable access to the FE bit. |
|   | SM0 | **Serial port Mode bit 0**<br>Refer to SM1 for serial port mode selection.<br>SMOD0 must be cleared to enable access to the SM0 bit. |
| 6 | SM1 | **Serial port Mode bit 1**<br>SM0  SM1    Mode     Baud Rate<br>0    0    Shift Register  $F_{XTAL}$/12 (or $F_{XTAL}$/6 in mode X2)<br>0    1    8-bit UART    Variable<br>1    0    9-bit UART    $F_{XTAL}$/64 or $F_{XTAL}$/32<br>1    1    9-bit UART    Variable |
| 5 | SM2 | **Serial port Mode 2 bit / Multiprocessor Communication Enable bit**<br>Clear to disable multiprocessor communication feature.<br>Set to enable multiprocessor communication feature in mode 2 and 3, and eventually mode 1. This bit should be cleared in mode 0. |
| 4 | REN | **Reception Enable bit**<br>Clear to disable serial reception.<br>Set to enable serial reception. |
| 3 | TB8 | **Transmitter Bit 8 / Ninth bit to transmit in modes 2 and 3**<br>o transmit a logic 0 in the 9th bit.<br>Set to transmit a logic 1 in the 9th bit. |
| 2 | RB8 | **Receiver Bit 8 / Ninth bit received in modes 2 and 3**<br>Cleared by hardware if 9th bit received is a logic 0.<br>Set by hardware if 9th bit received is a logic 1.<br>In mode 1, if SM2 = 0, RB8 is the received stop bit. In mode 0 RB8 is not used. |
| 1 | TI | **Transmit Interrupt flag**<br>Clear to acknowledge interrupt.<br>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. |
| 0 | RI | **Receive Interrupt flag**<br>Clear to acknowledge interrupt.<br>Set by hardware at the end of the 8th bit time in mode 0, see Figure 18. and Figure 19. in the other modes. |

Reset Value = 0000 0000b

Bit addressable

**Table 38.** Example of computed value when X2=1, SMOD1=1, SPD=1

| Baud Rates | $F_{OSCA}$ = 16.384 MHz | | $F_{OSCA}$ = 24MHz | |
|---|---|---|---|---|
| | BRL | Error (%) | BRL | Error (%) |
| 115200 | 247 | 1.23 | 243 | 0.16 |
| 57600 | 238 | 1.23 | 230 | 0.16 |
| 38400 | 229 | 1.23 | 217 | 0.16 |
| 28800 | 220 | 1.23 | 204 | 0.16 |
| 19200 | 203 | 0.63 | 178 | 0.16 |
| 9600 | 149 | 0.31 | 100 | 0.16 |
| 4800 | 43 | 1.23 | - | - |

**Table 39.** Example of computed value when X2=0, SMOD1=0, SPD=0

| Baud Rates | $F_{OSCA}$ = 16.384 MHz | | $F_{OSCA}$ = 24MHz | |
|---|---|---|---|---|
| | BRL | Error (%) | BRL | Error (%) |
| 4800 | 247 | 1.23 | 243 | 0.16 |
| 2400 | 238 | 1.23 | 230 | 0.16 |
| 1200 | 220 | 1.23 | 202 | 3.55 |
| 600 | 185 | 0.16 | 152 | 0.16 |

The baud rate generator can be used for mode 1 or 3 (refer to Figure 20.), but also for mode 0 for UART, thanks to the bit SRC located in BDRCON register (Table 46.)

**UART Registers**

**Table 40.** SADEN Register

SADEN - Slave Address Mask Register for UART (B9h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|   |   |   |   |   |   |   |   |

Reset Value = 0000 0000b

**Table 41.** SADDR Register

SADDR - Slave Address Register for UART (A9h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|   |   |   |   |   |   |   |   |

Reset Value = 0000 0000b

**Table 42.** SBUF Register

SBUF - Serial Buffer Register for UART (99h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|   |   |   |   |   |   |   |   |

Reset Value = XXXX XXXXb

**Table 43.** BRL Register

BRL - Baud Rate Reload Register for the internal baud rate generator, UART (9Ah)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|   |   |   |   |   |   |   |   |

Reset Value = 0000 0000b

**Table 44.** T2CON Register

T2CON - Timer 2 Control Register (C8h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TF2 | EXF2 | RCLK | TCLK | EXEN2 | TR2 | C/T2# | CP/RL2# |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | TF2 | **Timer 2 overflow Flag**<br>Must be cleared by software.<br>Set by hardware on timer 2 overflow, if RCLK = 0 and TCLK = 0. |
| 6 | EXF2 | **Timer 2 External Flag**<br>Set when a capture or a reload is caused by a negative transition on T2EX pin if EXEN2=1.<br>When set, causes the CPU to vector to timer 2 interrupt routine when timer 2 interrupt is enabled.<br>Must be cleared by software. EXF2 doesn't cause an interrupt in Up/down counter mode (DCEN = 1) |
| 5 | RCLK | **Receive Clock bit for UART**<br>Cleared to use timer 1 overflow as receive clock for serial port in mode 1 or 3.<br>Set to use timer 2 overflow as receive clock for serial port in mode 1 or 3. |
| 4 | TCLK | **Transmit Clock bit for UART**<br>Cleared to use timer 1 overflow as transmit clock for serial port in mode 1 or 3.<br>Set to use timer 2 overflow as transmit clock for serial port in mode 1 or 3. |
| 3 | EXEN2 | **Timer 2 External Enable bit**<br>Cleared to ignore events on T2EX pin for timer 2 operation.<br>Set to cause a capture or reload when a negative transition on T2EX pin is detected, if timer 2 is not used to clock the serial port. |
| 2 | TR2 | **Timer 2 Run control bit**<br>Cleared to turn off timer 2.<br>Set to turn on timer 2. |
| 1 | C/T2# | **Timer/Counter 2 select bit**<br>Cleared for timer operation (input from internal clock system: $F_{CLK\ PERIPH}$).<br>Set for counter operation (input from T2 input pin, falling edge trigger). Must be 0 for clock out mode. |
| 0 | CP/RL2# | **Timer 2 Capture/Reload bit**<br>If RCLK=1 or TCLK=1, CP/RL2# is ignored and timer is forced to auto-reload on timer 2 overflow.<br>Cleared to auto-reload on timer 2 overflows or negative transitions on T2EX pin if EXEN2=1.<br>Set to capture on negative transitions on T2EX pin if EXEN2=1. |

Reset Value = 0000 0000b
Bit addressable

**Table 45.** PCON Register

PCON - Power Control Register (87h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SMOD1 | SMOD0 | - | POF | GF1 | GF0 | PD | IDL |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | SMOD1 | **Serial port Mode bit 1 for UART**<br>Set to select double baud rate in mode 1, 2 or 3. |
| 6 | SMOD0 | **Serial port Mode bit 0 for UART**<br>Cleared to select SM0 bit in SCON register.<br>Set to select FE bit in SCON register. |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | POF | **Power-Off Flag**<br>Cleared to recognize next reset type.<br>Set by hardware when VCC rises from 0 to its nominal voltage. Can also be set by software. |
| 3 | GF1 | **General purpose Flag**<br>Cleared by user for general purpose usage.<br>Set by user for general purpose usage. |
| 2 | GF0 | **General purpose Flag**<br>Cleared by user for general purpose usage.<br>Set by user for general purpose usage. |
| 1 | PD | **Power-Down mode bit**<br>Cleared by hardware when reset occurs.<br>Set to enter power-down mode. |
| 0 | IDL | **Idle mode bit**<br>Cleared by hardware when interrupt or reset occurs.<br>Set to enter idle mode. |

Reset Value = 00X1 0000b
Not bit addressable

Power-off flag reset value will be 1 only after a power on (cold reset). A warm reset doesn't affect the value of this bit.

**Table 46.** BDRCON Register

BDRCON - Baud Rate Control Register (9Bh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | BRR | TBCK | RBCK | SPD | SRC |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 6 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | BRR | **Baud Rate Run Control bit**<br>Cleared to stop the internal Baud Rate Generator.<br>Set to start the internal Baud Rate Generator. |
| 3 | TBCK | **Transmission Baud rate Generator Selection bit for UART**<br>Cleared to select Timer 1 or Timer 2 for the Baud Rate Generator.<br>Set to select internal Baud Rate Generator. |
| 2 | RBCK | **Reception Baud Rate Generator Selection bit for UART**<br>Cleared to select Timer 1 or Timer 2 for the Baud Rate Generator.<br>Set to select internal Baud Rate Generator. |
| 1 | SPD | **Baud Rate Speed Control bit for UART**<br>Cleared to select the SLOW Baud Rate Generator.<br>Set to select the FAST Baud Rate Generator. |
| 0 | SRC | **Baud Rate Source select bit in Mode 0 for UART**<br>Cleared to select $F_{OSC}$/12 as the Baud Rate Generator ($F_{CLK\ PERIPH}$/6 in X2 mode).<br>Set to select the internal Baud Rate Generator for UARTs in mode 0. |

Reset Value = XXX0 0000b
Not bit addressable

**Interrupt System**

The AT89C51IC2 has a total of 10 interrupt vectors: two external interrupts ($\overline{INT0}$ and $\overline{INT1}$), three timer interrupts (timers 0, 1 and 2), the serial port interrupt, SPI interrupt, Two Wire Interface (I2C) interrupt, Keyboard interrupt and the PCA global interrupt. These interrupts are shown in Figure 22.

**Figure 22.** Interrupt Control System
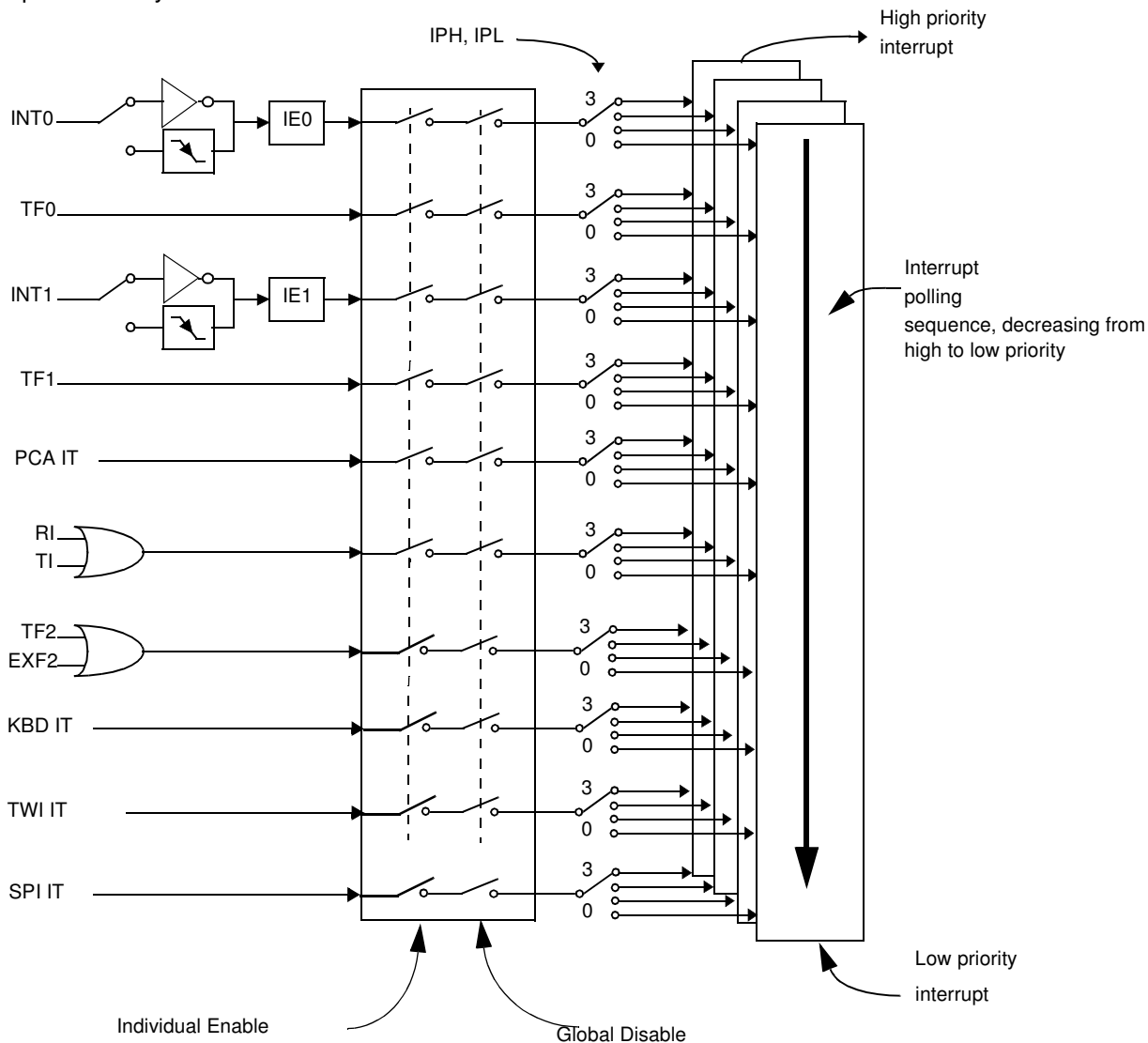


Each of the interrupt sources can be individually enabled or disabled by setting or clearing a bit in the Interrupt Enable register (Table 51 and Table 49). This register also contains a global disable bit, which must be cleared to disable all interrupts at once.

Each interrupt source can also be individually programmed to one out of four priority levels by setting or clearing a bit in the Interrupt Priority register (Table 52) and in the Interrupt Priority High register (Table 50 and Table 51) shows the bit values and priority levels associated with each combination.

**Registers**

The PCA interrupt vector is located at address 0033H, the SPI interrupt vector is located at address 0043H, the I2C interrupt vector at 0043H and Keyboard interrupt vector is located at address 003BH. All other vectors addresses are the same as standard C52 devices.

**Table 47.** Priority Level Bit Values

| IPH.x | IPL.x | Interrupt Level Priority |
|---|---|---|
| 0 | 0 | 0 (Lowest) |
| 0 | 1 | 1 |
| 1 | 0 | 2 |
| 1 | 1 | 3 (Highest) |

A low-priority interrupt can be interrupted by a high priority interrupt, but not by another low-priority interrupt. A high-priority interrupt can't be interrupted by any other interrupt source.

If two interrupt requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence.

**Table 48.** IENO Register

IEN0 - Interrupt Enable Register (A8h)

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

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | EA | **Enable All interrupt bit**<br>Cleared to disable all interrupts.<br>Set to enable all interrupts. |
| 6 | EC | **PCA interrupt enable bit**<br>Cleared to disable.<br>Set to enable. |
| 5 | ET2 | **Timer 2 overflow interrupt Enable bit**<br>Cleared to disable timer 2 overflow interrupt.<br>Set to enable timer 2 overflow interrupt. |
| 4 | ES | **Serial port Enable bit**<br>Cleared to disable serial port interrupt.<br>Set to enable serial port interrupt. |
| 3 | ET1 | **Timer 1 overflow interrupt Enable bit**<br>Cleared to disable timer 1 overflow interrupt.<br>Set to enable timer 1 overflow interrupt. |
| 2 | EX1 | **External interrupt 1 Enable bit**<br>Cleared to disable external interrupt 1.<br>Set to enable external interrupt 1. |
| 1 | ET0 | **Timer 0 overflow interrupt Enable bit**<br>Cleared to disable timer 0 overflow interrupt.<br>Set to enable timer 0 overflow interrupt. |
| 0 | EX0 | **External interrupt 0 Enable bit**<br>Cleared to disable external interrupt 0.<br>Set to enable external interrupt 0. |

Reset Value = 0000 0000b
Bit addressable

**Table 49.** IPL0 Register

IPL0 - Interrupt Priority Register (B8h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | PPCL | PT2L | PSL | PT1L | PX1L | PT0L | PX0L |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 6 | PPCL | **PCA interrupt Priority bit**<br>Refer to PPCH for priority level. |
| 5 | PT2L | **Timer 2 overflow interrupt Priority bit**<br>Refer to PT2H for priority level. |
| 4 | PSL | **Serial port Priority bit**<br>Refer to PSH for priority level. |
| 3 | PT1L | **Timer 1 overflow interrupt Priority bit**<br>Refer to PT1H for priority level. |
| 2 | PX1L | **External interrupt 1 Priority bit**<br>Refer to PX1H for priority level. |
| 1 | PT0L | **Timer 0 overflow interrupt Priority bit**<br>Refer to PT0H for priority level. |
| 0 | PX0L | **External interrupt 0 Priority bit**<br>Refer to PX0H for priority level. |

Reset Value = X000 0000b
Bit addressable

**Table 50.** IPH0 Register

IPH0 - Interrupt Priority High Register (B7h)

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

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 6 | PPCH | **PCA interrupt Priority high bit.**<br>PPCHPPCLPriority Level<br>0　0　　Lowest<br>0　1<br>1　0<br>1　1　　Highest |
| 5 | PT2H | **Timer 2 overflow interrupt Priority High bit**<br>PT2HPT2LPriority Level<br>0　0　　Lowest<br>0　1<br>1　0<br>1　1　　Highest |
| 4 | PSH | **Serial port Priority High bit**<br>PSH PSLPriority Level<br>0　0　　Lowest<br>0　1<br>1　0<br>1　1　　Highest |
| 3 | PT1H | **Timer 1 overflow interrupt Priority High bit**<br>PT1HPT1L Priority Level<br>0　0　　 Lowest<br>0　1<br>1　0<br>1　1　　Highest |
| 2 | PX1H | **External interrupt 1 Priority High bit**<br>PX1HPX1LPriority Level<br>0　0　　Lowest<br>0　1<br>1　0<br>1　1　　Highest |
| 1 | PT0H | **Timer 0 overflow interrupt Priority High bit**<br>PT0HPT0LPriority Level<br>0　0　　Lowest<br>0　1<br>1　0<br>1　1　　Highest |
| 0 | PX0H | **External interrupt 0 Priority High bit**<br>PX0H PX0LPriority Level<br>0　0　　Lowest<br>0　1<br>1　0<br>1　1　　Highest |

Reset Value = X000 0000b
Not bit addressable

**Table 51.** IEN1 Register

IEN1 - Interrupt Enable Register (B1h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | ESPI | ETWI | KBD |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved** |
| 6 | - | **Reserved** |
| 5 | - | **Reserved** |
| 4 | - | **Reserved** |
| 3 | - | **Reserved** |
| 2 | ESPI | **SPI interrupt Enable bit**<br>Cleared to disable SPI interrupt.<br>Set to enable SPI interrupt. |
| 1 | ETWI | **TWI interrupt Enable bit**<br>Cleared to disable TWI interrupt.<br>Set to enable TWI interrupt. |
| 0 | KBD | **Keyboard interrupt Enable bit**<br>Cleared to disable keyboard interrupt.<br>Set to enable keyboard interrupt. |

Reset Value = XXXX X000b
Bit addressable

**Table 52.** IPL1 Register

IPL1 - Interrupt Priority Register (B2h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | SPIL | TWIL | KBDL |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|-------------|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 6 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 3 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 2 | SPIL | **SPI interrupt Priority bit**<br>Refer to SPIH for priority level. |
| 1 | TWIL | **TWI interrupt Priority bit**<br>Refer to TWIH for priority level. |
| 0 | KBDL | **Keyboard interrupt Priority bit**<br>Refer to KBDH for priority level. |

Reset Value = XXXX X000b
Bit addressable

**Table 53.** IPH1 Register

IPH1 - Interrupt Priority High Register (B3h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | SPIH | TWIH | KBDH |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 6 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 3 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 2 | SPIH | **SPI interrupt Priority High bit**<br><u>SPIHSPILPriority Level</u><br>0  0    Lowest<br>0  1<br>1  0<br>1  1    Highest |
| 1 | TWIH | **TWI interrupt Priority High bit**<br><u>TWIHTWILPriority Level</u><br>0  0    Lowest<br>0  1<br>1  0<br>1  1    Highest |
| 0 | KBDH | **Keyboard interrupt Priority High bit**<br><u>KB DHKBDLPriority Level</u><br>0  0     Lowest<br>0  1<br>1  0<br>1  1    Highest |

Reset Value = XXXX X000b
Not bit addressable

## Interrupt Sources and Vector Addresses

**Table 54.** Interrupt Sources and Vector Addresses

| Number | Polling Priority | Interrupt Source | Interrupt Request | Vector Address |
|--------|------------------|------------------|-------------------|----------------|
| 0 | 0 | Reset | | 0000h |
| 1 | 1 | INT0 | IE0 | 0003h |
| 2 | 2 | Timer 0 | TF0 | 000Bh |
| 3 | 3 | INT1 | IE1 | 0013h |
| 4 | 4 | Timer 1 | IF1 | 001Bh |
| 5 | 6 | UART | RI+TI | 0023h |
| 6 | 7 | Timer 2 | TF2+EXF2 | 002Bh |
| 7 | 5 | PCA | CF + CCFn (n = 0-4) | 0033h |
| 8 | 8 | Keyboard | KBDIT | 003Bh |
| 9 | 9 | TWI | TWIIT | 0043h |
| 10 | 10 | SPI | SPIIT | 004Bh |

**Power Management**

Two power reduction modes are implemented in the AT89C51IC2: the Idle mode and the Power-down mode. These modes are detailed in the following sections. In addition to these power reduction modes, the clocks of the core and peripherals can be dynamically divided by 2 using the X2 mode detailed in Section "Enhanced Features".

**Reset**

In order to start-up (cold reset) or to restart (warm reset) properly the microcontroller, an high level has to be applied on the RST pin. A bad level leads to a wrong initialization of the internal registers like SFRs, Program Counter… and to unpredictable behavior of the microcontroller. A proper device reset initializes the AT89C51IC2 and vectors the CPU to address 0000h. RST input has a pull-down resistor allowing power-on reset by simply connecting an external capacitor to $V_{DD}$ as shown in Figure 23. A warm reset can be applied either directly on the RST pin or indirectly by an internal reset source such as the watchdog timer. Resistor value and input characteristics are discussed in the Section "DC Characteristics" of the AT89C51IC2 datasheet.

**Figure 23.**  Reset Circuitry and Power-On Reset



RST input circuitry                                    Power-on Reset
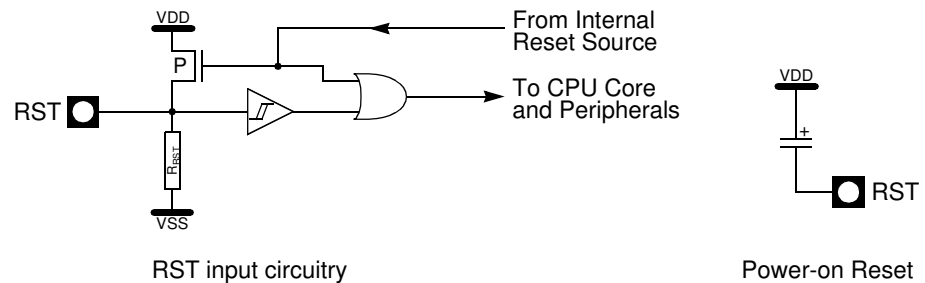
**Cold Reset**

2 conditions are required before enabling a CPU start-up:

*   $V_{DD}$ must reach the specified $V_{DD}$ range
*   The level on X1 input pin must be outside the specification ($V_{IH}$, $V_{IL}$)

If one of these 2 conditions are not met, the microcontroller does not start correctly and can execute an instruction fetch from anywhere in the program space. An active level applied on the RST pin must be maintained till both of the above conditions are met. A reset is active when the level $V_{IH1}$ is reached and when the pulse width covers the period of time where $V_{DD}$ and the oscillator are not stabilized. 2 parameters have to be taken into account to determine the reset pulse width:

*   $V_{DD}$ rise time,
*   Oscillator startup time.

To determine the capacitor value to implement, the highest value of these 2 parameters has to be chosen. Table 1 gives some capacitor values examples for a minimum $R_{RST}$ of 50 KΩ and different oscillator startup and $V_{DD}$ rise times.

**Table 1.** Minimum Reset Capacitor Value for a 50 kΩ Pull-down Resistor[1]

| Oscillator Start-Up Time | VDD Rise Time | | |
|---|---|---|---|
| | 1 ms | 10 ms | 100 ms |
| 5 ms | 820 nF | 1.2 µF | 12 µF |
| 20 ms | 2.7 µF | 3.9 µF | 12 µF |

Note: These values assume $V_{DD}$ starts from 0V to the nominal value. If the time between 2 on/off sequences is too fast, the power-supply de-coupling capacitors may not be fully discharged, leading to a bad reset sequence.

**Warm Reset**

To achieve a valid reset, the reset signal must be maintained for at least 2 machine cycles (24 oscillator clock periods) while the oscillator is running. The number of clock periods is mode independent (X2 or X1).

**Watchdog Reset**

As detailed in Section "Hardware Watchdog Timer", page 102, the WDT generates a 96-clock period pulse on the RST pin. In order to properly propagate this pulse to the rest of the application in case of external capacitor or power-supply supervisor circuit, a 1 kΩ resistor must be added as shown Figure 24.

**Figure 24.** Reset Circuitry for WDT Reset-out Usage
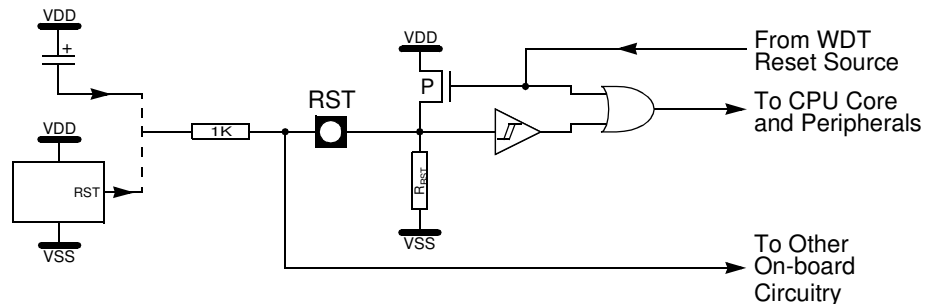
**Reset Recommendation to Prevent Flash Corruption**

An example of bad initialization situation may occur in an instance where the bit ENBOOT in AUXR1 register is initialized from the hardware bit BLJB upon reset. Since this bit allows mapping of the bootloader in the code area, a reset failure can be critical.

If one wants the ENBOOT cleared in order to unmap the boot from the code area (yet due to a bad reset) the bit ENBOOT in SFRs may be set. If the value of Program Counter is accidently in the range of the boot memory addresses then a Flash access (write or erase) may corrupt the Flash on-chip memory.

It is recommended to use an external reset circuitry featuring power supply monitoring to prevent system malfunction during periods of insufficient power supply voltage (power supply failure, power supply switched off).

**Idle Mode**

An instruction that sets PCON.0 indicates that it is the last instruction to be executed before going into Idle mode. In Idle mode, the internal clock signal is gated off to the CPU, but not to the interrupt, Timer, and Serial Port functions. The CPU status is preserved in its entirety: the Stack Pointer, Program Counter, Program Status Word, Accumulator and all other registers maintain their data during idle. The port pins hold the logical states they had at the time Idle was activated. ALE and PSEN hold at logic high level.

There are two ways to terminate the Idle mode. Activation of any enabled interrupt will cause PCON.0 to be cleared by hardware, terminating the Idle mode. The interrupt will be serviced, and following RETI the next instruction to be executed will be the one following the instruction that put the device into idle.

The flag bits GF0 and GF1 can be used to give an indication if an interrupt occurred during normal operation or during idle. For example, an instruction that activates idle can also set one or both flag bits. When idle is terminated by an interrupt, the interrupt service routine can examine the flag bits.

The other way of terminating the Idle mode is with a hardware reset. Since the clock oscillator is still running, the hardware reset needs to be held active for only two machine cycles (24 oscillator periods) to complete the reset.

**Power-down Mode**

To save maximum power, a Power-down mode can be invoked by software (see PCON register).

In Power-down mode, the oscillator is stopped and the instruction that invoked Power-down mode is the last instruction executed. The internal RAM and SFRs retain their value until the Power-down mode is terminated. $V_{CC}$ can be lowered to save further power. Either a hardware reset or an external interrupt can cause an exit from Power-down. 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.

Only external interrupts $\overline{INT0}$, $\overline{INT1}$ and Keyboard Interrupts are useful to exit from Power-down. For that, interrupt must be enabled and configured as level or edge sensitive interrupt input. When Keyboard Interrupt occurs after a power down mode, 1024 clocks are necessary to exit to power down mode and enter in operating mode.

Holding the pin low restarts the oscillator but bringing the pin high completes the exit as detailed in Figure 25. When both interrupts are enabled, the oscillator restarts as soon as one of the two inputs is held low and power down exit will be completed when the first input will be released. In this case, the higher priority interrupt service routine is executed. Once the interrupt is serviced, the next instruction to be executed after RETI will be the one following the instruction that puts the AT89C51IC2 into Power-down mode.

**Figure 25.** Power-down Exit Waveform
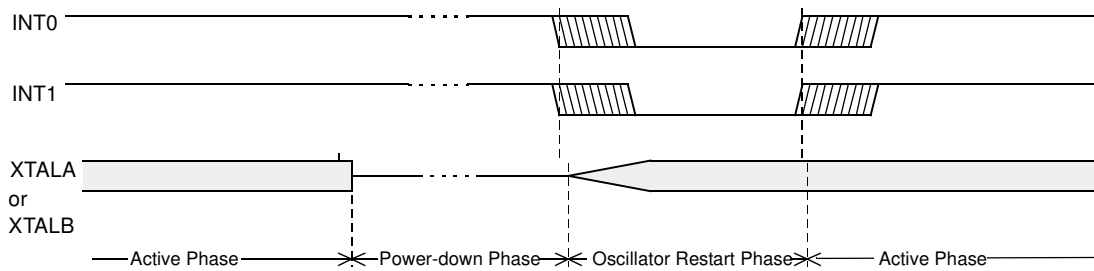


Exit from Power-down by reset redefines all the SFRs, exit from Power-down by external interrupt does no affect the SFRs.

Exit from Power-down by either reset or external interrupt does not affect the internal RAM content.

Note: If idle mode is activated with Power-down mode (IDL and PD bits set), the exit sequence is unchanged, when execution is vectored to interrupt, PD and IDL bits are cleared and idle mode is not entered.

Table 55 shows the state of ports during idle and power-down modes.

**Table 55.** State of Ports

| Mode | Program Memory | ALE | PSEN | PORT0 | PORT1 | PORT2 | PORT3 |
|------|---------------|-----|------|-------|-------|-------|-------|
| Idle | Internal | 1 | 1 | Port Data[1] | Port Data | Port Data | Port Data |
| Idle | External | 1 | 1 | Floating | Port Data | Address | Port Data |
| Power Down | Internal | 0 | 0 | Port Data[1] | Port Data | Port Data | Port Data |
| Power Down | External | 0 | 0 | Floating | Port Data | Port Data | Port Data |

Port 0 can force a 0 level. A "one" will leave port floating.

## Serial Port Interface (SPI)

The Serial Peripheral Interface Module (SPI) allows full-duplex, synchronous, serial communication between the MCU and peripheral devices, including other MCUs.

### Features

Features of the SPI Module include the following:

- Full-duplex, three-wire synchronous transfers
- Master or Slave operation
- Eight programmable Master clock rates
- Serial clock with programmable polarity and phase
- Master Mode fault error flag with MCU interrupt capability
- Write collision flag protection

### Signal Description

Figure 26 shows a typical SPI bus configuration using one Master controller and many Slave peripherals. The bus is made of three wires connecting all the devices.

**Figure 26.** SPI Master/Slaves Interconnection
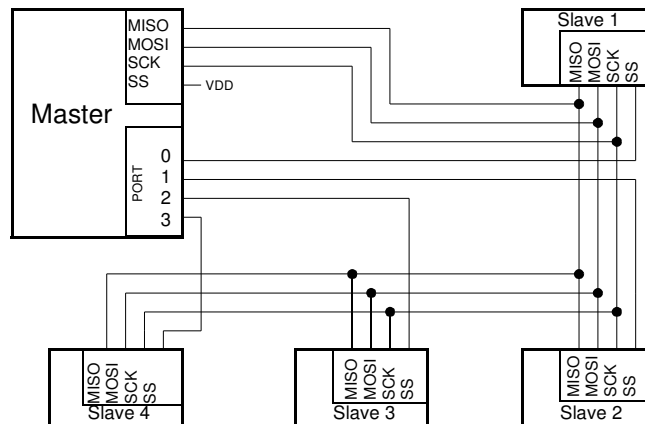


The Master device selects the individual Slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the Slave devices.

**Master Output Slave Input (MOSI)**

This 1-bit signal is directly connected between the Master Device and a Slave Device. The MOSI line is used to transfer data in series from the Master to the Slave. Therefore, it is an output signal from the Master, and an input signal to a Slave. A Byte (8-bit word) is transmitted most significant bit (MSB) first, least significant bit (LSB) last.

**Master Input Slave Output (MISO)**

This 1-bit signal is directly connected between the Slave Device and a Master Device. The MISO line is used to transfer data in series from the Slave to the Master. Therefore, it is an output signal from the Slave, and an input signal to the Master. A Byte (8-bit word) is transmitted most significant bit (MSB) first, least significant bit (LSB) last.

**SPI Serial Clock (SCK)**

This signal is used to synchronize the data movement both in and out of the devices through their MOSI and MISO lines. It is driven by the Master for eight clock cycles which allows to exchange one Byte on the serial lines.

**Slave Select ($\overline{SS}$)**

Each Slave peripheral is selected by one Slave Select pin ($\overline{SS}$). This signal must stay low for any message for a Slave. It is obvious that only one Master ($\overline{SS}$ high level) can

drive the network. The Master may select each Slave device by software through port pins (Figure 27). To prevent bus conflicts on the MISO line, only one slave should be selected at a time by the Master for a transmission.

In a Master configuration, the $\overline{SS}$ line can be used in conjunction with the MODF flag in the SPI Status register (SPSTA) to prevent multiple masters from driving MOSI and SCK (see Error conditions).

A high level on the $\overline{SS}$ pin puts the MISO line of a Slave SPI in a high-impedance state.

The $\overline{SS}$ pin could be used as a general-purpose if the following conditions are met:

- The device is configured as a Master and the SSDIS control bit in SPCON is set. This kind of configuration can be found when only one Master is driving the network and there is no way that the $\overline{SS}$ pin could be pulled low. Therefore, the MODF flag in the SPSTA will never be set[1].

- The Device is configured as a Slave with CPHA and SSDIS control bits set[2]. This kind of configuration can happen when the system comprises one Master and one Slave only. Therefore, the device should always be selected and there is no reason that the Master uses the $\overline{SS}$ pin to select the communicating Slave device.

Note: 1. Clearing SSDIS control bit does not clear MODF.

2. Special care should be taken not to set SSDIS control bit when CPHA = '0' because in this mode, the $\overline{SS}$ is used to start the transmission.

**Baud Rate**

In Master mode, the baud rate can be selected from a baud rate generator which is controlled by three bits in the SPCON register: SPR2, SPR1 and SPR0.The Master clock is selected from one of seven clock rates resulting from the division of the internal clock by 2, 4, 8, 16, 32, 64 or 128.

Table 56 gives the different clock rates selected by SPR2:SPR1:SPR0.

**Table 56.** SPI Master Baud Rate Selection

| SPR2 | SPR1 | SPR0 | Clock Rate | Baud Rate Divisor (BD) |
|------|------|------|------------|------------------------|
| 0 | 0 | 0 | $F_{CLK\ PERIPH}$ /2 | 2 |
| 0 | 0 | 1 | $F_{CLK\ PERIPH}$ /4 | 4 |
| 0 | 1 | 0 | $F_{CLK\ PERIPH}$/8 | 8 |
| 0 | 1 | 1 | $F_{CLK\ PERIPH}$ /16 | 16 |
| 1 | 0 | 0 | $F_{CLK\ PERIPH}$ /32 | 32 |
| 1 | 0 | 1 | $F_{CLK\ PERIPH}$ /64 | 64 |
| 1 | 1 | 0 | $F_{CLK\ PERIPH}$ /128 | 128 |
| 1 | 1 | 1 | Don't Use | No BRG |

**Functional Description**    Figure 27 shows a detailed structure of the SPI Module.

**Figure 27.** SPI Module Block Diagram
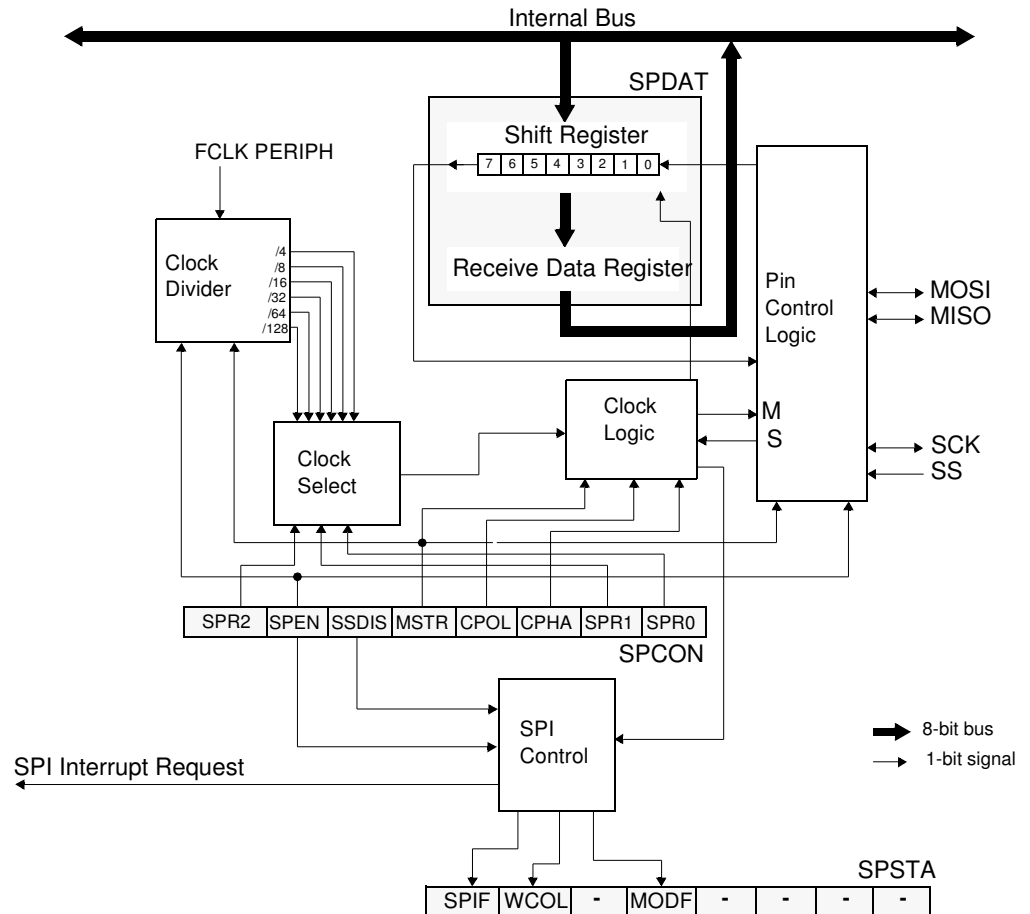


**Operating Modes**    The Serial Peripheral Interface can be configured in one of the two modes: Master mode or Slave mode. The configuration and initialization of the SPI Module is made through one register:

• The Serial Peripheral Control register (SPCON)

Once the SPI is configured, the data exchange is made using:

• SPCON
• The Serial Peripheral STAtus register (SPSTA)
• The Serial Peripheral DATa register (SPDAT)

During an SPI transmission, data is simultaneously transmitted (shifted out serially) and received (shifted in serially). A serial clock line (SCK) synchronizes shifting and sampling on the two serial data lines (MOSI and MISO). A Slave Select line (SS) allows individual selection of a Slave SPI device; Slave devices that are not selected do not interfere with SPI bus activities.

When the Master device transmits data to the Slave device via the MOSI line, the Slave device responds by sending data to the Master device via the MISO line. This implies full-duplex transmission with both data out and data in synchronized with the same clock (Figure 28).

**Figure 28.** Full-Duplex Master-Slave Interconnection
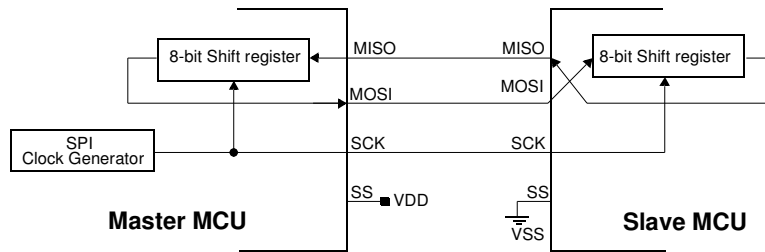


*Master Mode*

The SPI operates in Master mode when the Master bit, MSTR [1], in the SPCON register is set. Only one Master SPI device can initiate transmissions. Software begins the transmission from a Master SPI Module by writing to the Serial Peripheral Data Register (SPDAT). If the shift register is empty, the Byte is immediately transferred to the shift register. The Byte begins shifting out on MOSI pin under the control of the serial clock, SCK. Simultaneously, another Byte shifts in from the Slave on the Master's MISO pin. The transmission ends when the Serial Peripheral transfer data flag, SPIF, in SPSTA becomes set. At the same time that SPIF becomes set, the received Byte from the Slave is transferred to the receive data register in SPDAT. Software clears SPIF by reading the Serial Peripheral Status register (SPSTA) with the SPIF bit set, and then reading the SPDAT.

*Slave Mode*

The SPI operates in Slave mode when the Master bit, MSTR [2], in the SPCON register is cleared. Before a data transmission occurs, the Slave Select pin, $\overline{SS}$, of the Slave device must be set to '0'. $\overline{SS}$ must remain low until the transmission is complete.

In a Slave SPI Module, data enters the shift register under the control of the SCK from the Master SPI Module. After a Byte enters the shift register, it is immediately transferred to the receive data register in SPDAT, and the SPIF bit is set. To prevent an overflow condition, Slave software must then read the SPDAT before another Byte enters the shift register [3]. A Slave SPI must complete the write to the SPDAT (shift register) at least one bus cycle before the Master SPI starts a transmission. If the write to the data register is late, the SPI transmits the data already in the shift register from the previous transmission. The maximum SCK frequency allowed in slave mode is $F_{CLK\ PERIPH}$ /4.

**Transmission Formats**

Software can select any of four combinations of serial clock (SCK) phase and polarity using two bits in the SPCON: the Clock Polarity (CPOL [4]) and the Clock Phase (CPHA[4]). CPOL defines the default SCK line level in idle state. It has no significant effect on the transmission format. CPHA defines the edges on which the input data are sampled and the edges on which the output data are shifted (Figure 29 and Figure 30). The clock phase and polarity should be identical for the Master SPI device and the communicating Slave device.

1.    The SPI Module should be configured as a Master before it is enabled (SPEN set). Also, the Master SPI should be configured before the Slave SPI.
2.    The SPI Module should be configured as a Slave before it is enabled (SPEN set).
3.    The maximum frequency of the SCK for an SPI configured as a Slave is the bus clock speed.
4.    Before writing to the CPOL and CPHA bits, the SPI should be disabled (SPEN = '0').
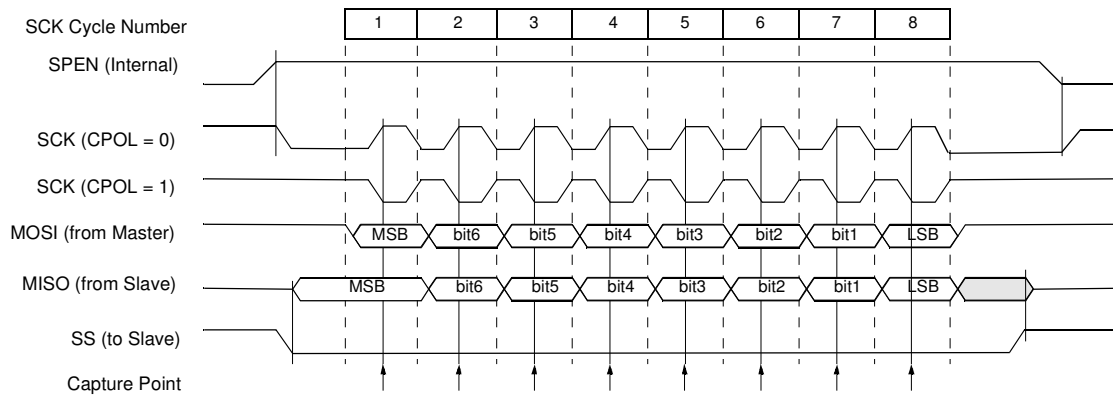
**Figure 29.** Data Transmission Format (CPHA = 0)



**Figure 30.** Data Transmission Format (CPHA = 1)
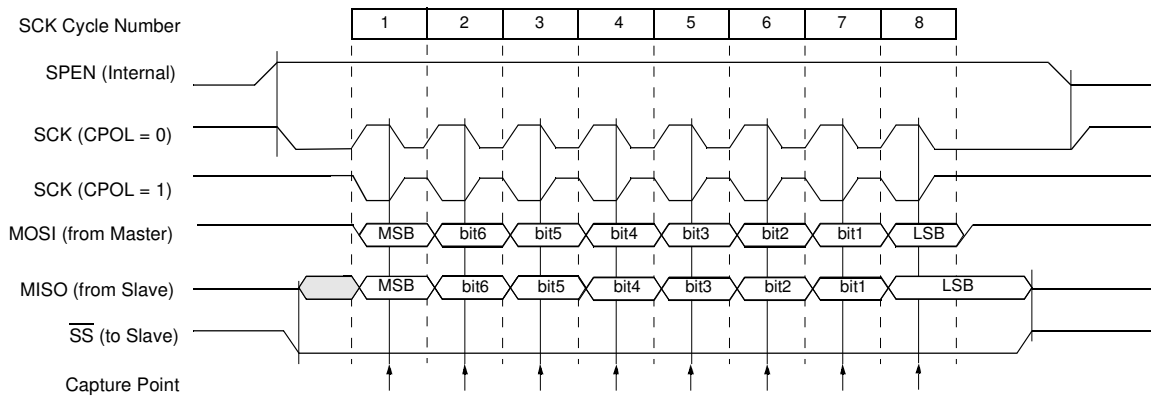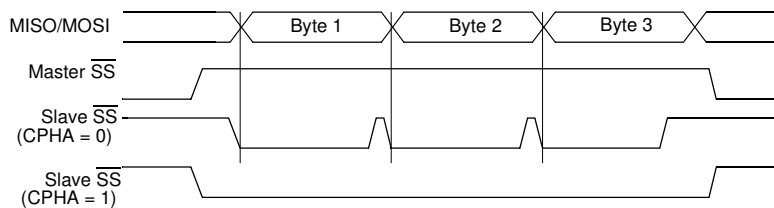


**Figure 31.** CPHA/$\overline{SS}$ Timing



As shown in Figure 29, the first SCK edge is the MSB capture strobe. Therefore, the Slave must begin driving its data before the first SCK edge, and a falling edge on the $\overline{SS}$ pin is used to start the transmission. The $\overline{SS}$ pin must be toggled high and then low between each Byte transmitted (Figure 31).

Figure 30 shows an SPI transmission in which CPHA is '1'. In this case, the Master begins driving its MOSI pin on the first SCK edge. Therefore, the Slave uses the first SCK edge as a start transmission signal. The $\overline{SS}$ pin can remain low between transmissions (Figure 31). This format may be preffered in systems having only one Master and only one Slave driving the MISO data line.

**Error Conditions**

The following flags in the SPSTA signal SPI error conditions:

*Mode Fault (MODF)*

Mode Fault error in Master mode SPI indicates that the level on the Slave Select ($\overline{SS}$) pin is inconsistent with the actual mode of the device. MODF is set to warn that there may be a multi-master conflict for system control. In this case, the SPI system is affected in the following ways:

• An SPI receiver/error CPU interrupt request is generated

• The SPEN bit in SPCON is cleared. This disables the SPI

• The MSTR bit in SPCON is cleared

When $\overline{SS}$ Disable (SSDIS) bit in the SPCON register is cleared, the MODF flag is set when the $\overline{SS}$ signal becomes '0'.

However, as stated before, for a system with one Master, if the $\overline{SS}$ pin of the Master device is pulled low, there is no way that another Master attempts to drive the network. In this case, to prevent the MODF flag from being set, software can set the SSDIS bit in the SPCON register and therefore making the $\overline{SS}$ pin as a general-purpose I/O pin.

Clearing the MODF bit is accomplished by a read of SPSTA register with MODF bit set, followed by a write to the SPCON register. SPEN Control bit may be restored to its original set state after the MODF bit has been cleared.

*Write Collision (WCOL)*

A Write Collision (WCOL) flag in the SPSTA is set when a write to the SPDAT register is done during a transmit sequence.

WCOL does not cause an interruption, and the transfer continues uninterrupted.

Clearing the WCOL bit is done through a software sequence of an access to SPSTA and an access to SPDAT.

*Overrun Condition*

An overrun condition occurs when the Master device tries to send several data Bytes and the Slave devise has not cleared the SPIF bit issuing from the previous data Byte transmitted. In this case, the receiver buffer contains the Byte sent after the SPIF bit was last cleared. A read of the SPDAT returns this Byte. All others Bytes are lost.

This condition is not detected by the SPI peripheral.

*SS Error Flag (SSERR)*

A Synchronous Serial Slave Error occurs when $\overline{SS}$ goes high before the end of a received data in slave mode. SSERR does not cause in interruption, this bit is cleared by writing 0 to SPEN bit (reset of the SPI state machine).

**Interrupts**

Two SPI status flags can generate a CPU interrupt requests:

**Table 57.** SPI Interrupts

| Flag | Request |
|------|---------|
| SPIF (SP data transfer) | SPI Transmitter Interrupt request |
| MODF (Mode Fault) | SPI Receiver/Error Interrupt Request (if SSDIS = '0') |

Serial Peripheral data transfer flag, SPIF: This bit is set by hardware when a transfer has been completed. SPIF bit generates transmitter CPU interrupt requests.

Mode Fault flag, MODF: This bit becomes set to indicate that the level on the SS is inconsistent with the mode of the SPI. MODF with SSDIS reset, generates receiver/error CPU interrupt requests. When SSDIS is set, no MODF interrupt request is generated.

Figure 32 gives a logical view of the above statements.

**Figure 32.** SPI Interrupt Requests Generation
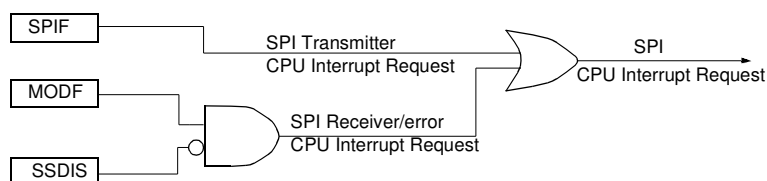


**Registers**

There are three registers in the Module that provide control, status and data storage functions. These registers are describes in the following paragraphs.

*Serial Peripheral Control Register (SPCON)*

- The Serial Peripheral Control Register does the following:
- Selects one of the Master clock rates
- Configure the SPI Module as Master or Slave
- Selects serial clock polarity and phase
- Enables the SPI Module
- Frees the SS pin for a general-purpose

Table 58 describes this register and explains the use of each bit

**Table 58.** SPCON Register

SPCON - Serial Peripheral Control Register (0C3H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SPR2 | SPEN | SSDIS | MSTR | CPOL | CPHA | SPR1 | SPR0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | SPR2 | **Serial Peripheral Rate 2**<br>Bit with SPR1 and SPR0 define the clock rate. |
| 6 | SPEN | **Serial Peripheral Enable**<br>Cleared to disable the SPI interface.<br>Set to enable the SPI interface. |
| 5 | SSDIS | **$\overline{SS}$ Disable**<br>Cleared to enable $\overline{SS}$ in both Master and Slave modes.<br>Set to disable $\overline{SS}$ in both Master and Slave modes. In Slave mode, this bit has no effect if CPHA ='0'. When SSDIS is set, no MODF interrupt request is generated. |
| 4 | MSTR | **Serial Peripheral Master**<br>Cleared to configure the SPI as a Slave.<br>Set to configure the SPI as a Master. |
| 3 | CPOL | **Clock Polarity**<br>Cleared to have the SCK set to '0' in idle state.<br>Set to have the SCK set to '1' in idle low. |
| 2 | CPHA | **Clock Phase**<br>Cleared to have the data sampled when the SCK leaves the idle state (see CPOL).<br>Set to have the data sampled when the SCK returns to idle state (see CPOL). |

| Bit Number | Bit Mnemonic | Description | | | |
|---|---|---|---|---|---|
| 1 | SPR1 | **SPR2** **SPR1** **SPR0** **Serial Peripheral Rate** | | | |
| | | 0 | 0 | 0 | $F_{CLK\ PERIPH}/2$ |
| | | 0 | 0 | 1 | $F_{CLK\ PERIPH}/4$ |
| | | 0 | 1 | 0 | $F_{CLK\ PERIPH}/8$ |
| | | 0 | 1 | 1 | $F_{CLK\ PERIPH}/16$ |
| 0 | SPR0 | 1 | 0 | 0 | $F_{CLK\ PERIPH}/32$ |
| | | 1 | 0 | 1 | $F_{CLK\ PERIPH}/64$ |
| | | 1 | 1 | 0 | $F_{CLK\ PERIPH}/128$ |
| | | 1 | 1 | 1 | Invalid |

Reset Value = 0001 0100b

Not bit addressable

*Serial Peripheral Status Register (SPSTA)*

The Serial Peripheral Status Register contains flags to signal the following conditions:

- Data transfer complete
- Write collision
- Inconsistent logic level on $\overline{SS}$ pin (mode fault error)

Table 59 describes the SPSTA register and explains the use of every bit in the register.

**Table 59.** SPSTA Register

SPSTA - Serial Peripheral Status and Control register (0C4H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SPIF | WCOL | SSERR | MODF | - | - | - | - |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | SPIF | **Serial Peripheral Data Transfer Flag**<br>Cleared by hardware to indicate data transfer is in progress or has been approved by a clearing sequence.<br>Set by hardware to indicate that the data transfer has been completed. |
| 6 | WCOL | **Write Collision Flag**<br>Cleared by hardware to indicate that no collision has occurred or has been approved by a clearing sequence.<br>Set by hardware to indicate that a collision has been detected. |
| 5 | SSERR | **Synchronous Serial Slave Error Flag**<br>Set by hardware when $\overline{SS}$ is deasserted before the end of a received data.<br>Cleared by disabling the SPI (clearing SPEN bit in SPCON). |
| 4 | MODF | **Mode Fault**<br>Cleared by hardware to indicate that the $\overline{SS}$ pin is at appropriate logic level, or has been approved by a clearing sequence.<br>Set by hardware to indicate that the $\overline{SS}$ pin is at inappropriate logic level. |
| 3 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 2 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 1 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 0 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |

Reset Value = 00X0 XXXXb

Not Bit addressable

*Serial Peripheral DATa Register (SPDAT)*

The Serial Peripheral Data Register (Table 60) is a read/write buffer for the receive data register. A write to SPDAT places data directly into the shift register. No transmit buffer is available in this model.

A Read of the SPDAT returns the value located in the receive buffer and not the content of the shift register.

**Table 60.** SPDAT Register

SPDAT - Serial Peripheral Data Register (0C5H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |

Reset Value = Indeterminate

R7:R0:  Receive data bits

SPCON, SPSTA and SPDAT registers may be read and written at any time while there is no on-going exchange. However, special care should be taken when writing to them while a transmission is on-going:

• Do not change SPR2, SPR1 and SPR0

• Do not change CPHA and CPOL

• Do not change MSTR

• Clearing SPEN would immediately disable the peripheral

• Writing to the SPDAT will cause an overflow.

# Keyboard Interface

The AT89C51IC2 implements a keyboard interface allowing the connection of a 8 x n matrix keyboard. It is based on 8 inputs with programmable interrupt capability on both high or low level. These inputs are available as alternate function of P1 and allow to exit from idle and power down modes.

The keyboard interface interfaces with the C51 core through 3 special function registers: KBLS, the Keyboard Level Selection register (Table 63), KBE, The Keyboard interrupt Enable register (Table 62), and KBF, the Keyboard Flag register (Table 61).

**Interrupt**

The keyboard inputs are considered as 8 independent interrupt sources sharing the same interrupt vector. An interrupt enable bit (KBD in IEN1) allows global enable or disable of the keyboard interrupt (see Figure 33). As detailed in Figure 34 each keyboard input has the capability to detect a programmable level according to KBLS.x bit value. Level detection is then reported in interrupt flags KBF.x that can be masked by software using KBE.x bits.

This structure allow keyboard arrangement from 1 by n to 8 by n matrix and allow usage of P1 inputs for other purpose.

**Figure 33.** Keyboard Interface Block Diagram
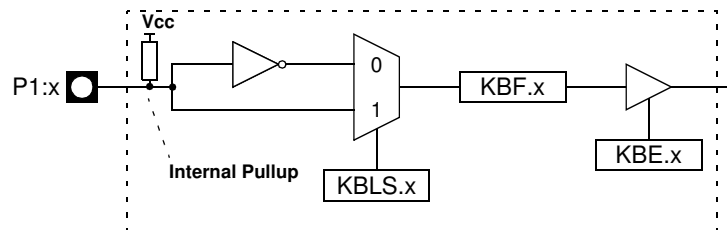


**Figure 34.** Keyboard Input Circuitry
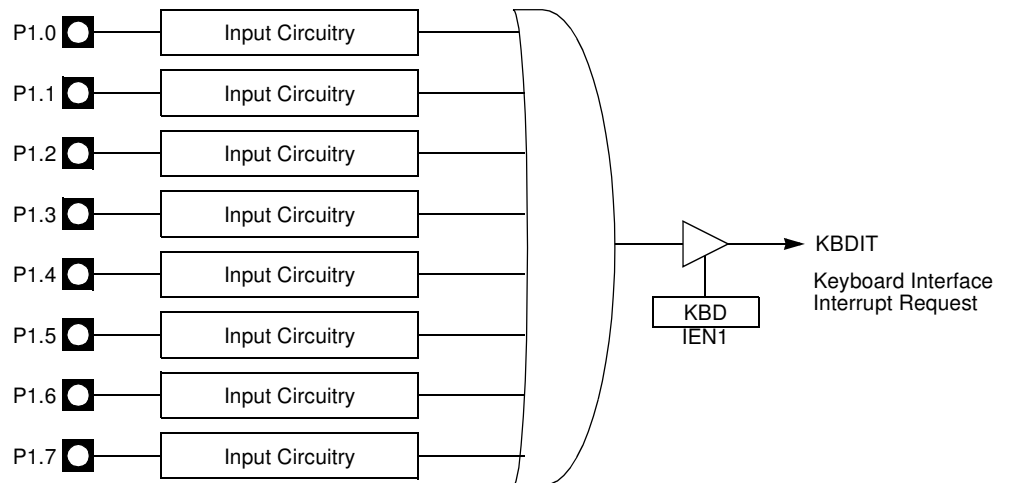


**Power Reduction Mode**

P1 inputs allow exit from idle and power down modes as detailed in Section "Power Management", page 67.

**Registers**

**Table 61. KBF Register**

KBF-Keyboard Flag Register (9Eh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| KBF7 | KBF6 | KBF5 | KBF4 | KBF3 | KBF2 | KBF1 | KBF0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | KBF7 | **Keyboard line 7 flag**<br>Set by hardware when the Port line 7 detects a programmed level. It generates a Keyboard interrupt request if the KBKBIE.7 bit in KBIE register is set.<br>Must be cleared by software. |
| 6 | KBF6 | **Keyboard line 6 flag**<br>Set by hardware when the Port line 6 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.6 bit in KBIE register is set.<br>Must be cleared by software. |
| 5 | KBF5 | **Keyboard line 5 flag**<br>Set by hardware when the Port line 5 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.5 bit in KBIE register is set.<br>Must be cleared by software. |
| 4 | KBF4 | **Keyboard line 4 flag**<br>Set by hardware when the Port line 4 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.4 bit in KBIE register is set.<br>Must be cleared by software. |
| 3 | KBF3 | **Keyboard line 3 flag**<br>Set by hardware when the Port line 3 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.3 bit in KBIE register is set.<br>Must be cleared by software. |
| 2 | KBF2 | **Keyboard line 2 flag**<br>Set by hardware when the Port line 2 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.2 bit in KBIE register is set.<br>Must be cleared by software. |
| 1 | KBF1 | **Keyboard line 1 flag**<br>Set by hardware when the Port line 1 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.1 bit in KBIE register is set.<br>Must be cleared by software. |
| 0 | KBF0 | **Keyboard line 0 flag**<br>Set by hardware when the Port line 0 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.0 bit in KBIE register is set.<br>Must be cleared by software. |

Reset Value= 0000 0000b

This register is read only access, all flags are automatically cleared by reading the register.

**Table 62.** KBE Register

KBE-Keyboard Input Enable Register (9Dh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| KBE7 | KBE6 | KBE5 | KBE4 | KBE3 | KBE2 | KBE1 | KBE0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | KBE7 | **Keyboard line 7 Enable bit**<br>Cleared to enable standard I/O pin.<br>Set to enable KBF.7 bit in KBF register to generate an interrupt request. |
| 6 | KBE6 | **Keyboard line 6 Enable bit**<br>Cleared to enable standard I/O pin.<br>Set to enable KBF.6 bit in KBF register to generate an interrupt request. |
| 5 | KBE5 | **Keyboard line 5 Enable bit**<br>Cleared to enable standard I/O pin.<br>Set to enable KBF.5 bit in KBF register to generate an interrupt request. |
| 4 | KBE4 | **Keyboard line 4 Enable bit**<br>Cleared to enable standard I/O pin.<br>Set to enable KBF.4 bit in KBF register to generate an interrupt request. |
| 3 | KBE3 | **Keyboard line 3 Enable bit**<br>Cleared to enable standard I/O pin.<br>Set to enable KBF.3 bit in KBF register to generate an interrupt request. |
| 2 | KBE2 | **Keyboard line 2 Enable bit**<br>Cleared to enable standard I/O pin.<br>Set to enable KBF.2 bit in KBF register to generate an interrupt request. |
| 1 | KBE1 | **Keyboard line 1 Enable bit**<br>Cleared to enable standard I/O pin.<br>Set to enable KBF.1 bit in KBF register to generate an interrupt request. |
| 0 | KBE0 | **Keyboard line 0 Enable bit**<br>Cleared to enable standard I/O pin.<br>Set to enable KBF.0 bit in KBF register to generate an interrupt request. |

Reset Value= 0000 0000b

**Table 63.** KBLS Register

KBLS-Keyboard Level Selector Register (9Ch)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| KBLS7 | KBLS6 | KBLS5 | KBLS4 | KBLS3 | KBLS2 | KBLS1 | KBLS0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | KBLS7 | **Keyboard line 7 Level Selection bit**<br>Cleared to enable a low level detection on Port line 7.<br>Set to enable a high level detection on Port line 7. |
| 6 | KBLS6 | **Keyboard line 6 Level Selection bit**<br>Cleared to enable a low level detection on Port line 6.<br>Set to enable a high level detection on Port line 6. |
| 5 | KBLS5 | **Keyboard line 5 Level Selection bit**<br>Cleared to enable a low level detection on Port line 5.<br>Set to enable a high level detection on Port line 5. |
| 4 | KBLS4 | **Keyboard line 4 Level Selection bit**<br>Cleared to enable a low level detection on Port line 4.<br>Set to enable a high level detection on Port line 4. |
| 3 | KBLS3 | **Keyboard line 3 Level Selection bit**<br>Cleared to enable a low level detection on Port line 3.<br>Set to enable a high level detection on Port line 3. |
| 2 | KBLS2 | **Keyboard line 2 Level Selection bit**<br>Cleared to enable a low level detection on Port line 2.<br>Set to enable a high level detection on Port line 2. |
| 1 | KBLS1 | **Keyboard line 1 Level Selection bit**<br>Cleared to enable a low level detection on Port line 1.<br>Set to enable a high level detection on Port line 1. |
| 0 | KBLS0 | **Keyboard line 0 Level Selection bit**<br>Cleared to enable a low level detection on Port line 0.<br>Set to enable a high level detection on Port line 0. |

Reset Value= 0000 0000b

## 2-wire Interface (TWI)

This section describes the 2-wire interface. In the rest of the section SSLC means Two-wire. The 2-wire bus is a bi-directional 2-wire serial communication standard. It is designed primarily for simple but efficient integrated circuit (IC) control. The system is comprised of two lines, SCL (Serial Clock) and SDA (Serial Data) that carry information between the ICs connected to them. The serial data transfer is limited to 400Kbit/s in standard mode. Various communication configuration can be designed using this bus. Figure 35 shows a typical 2-wire bus configuration. All the devices connected to the bus can be master and slave.

**Figure 35.** 2-wire Bus Configuration
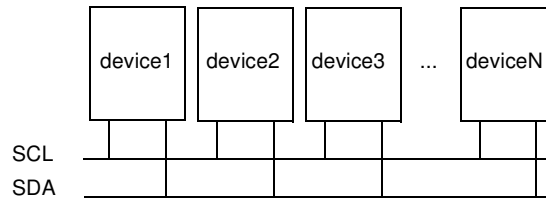
**Figure 36.** Block Diagram

4301D–8051–02/08
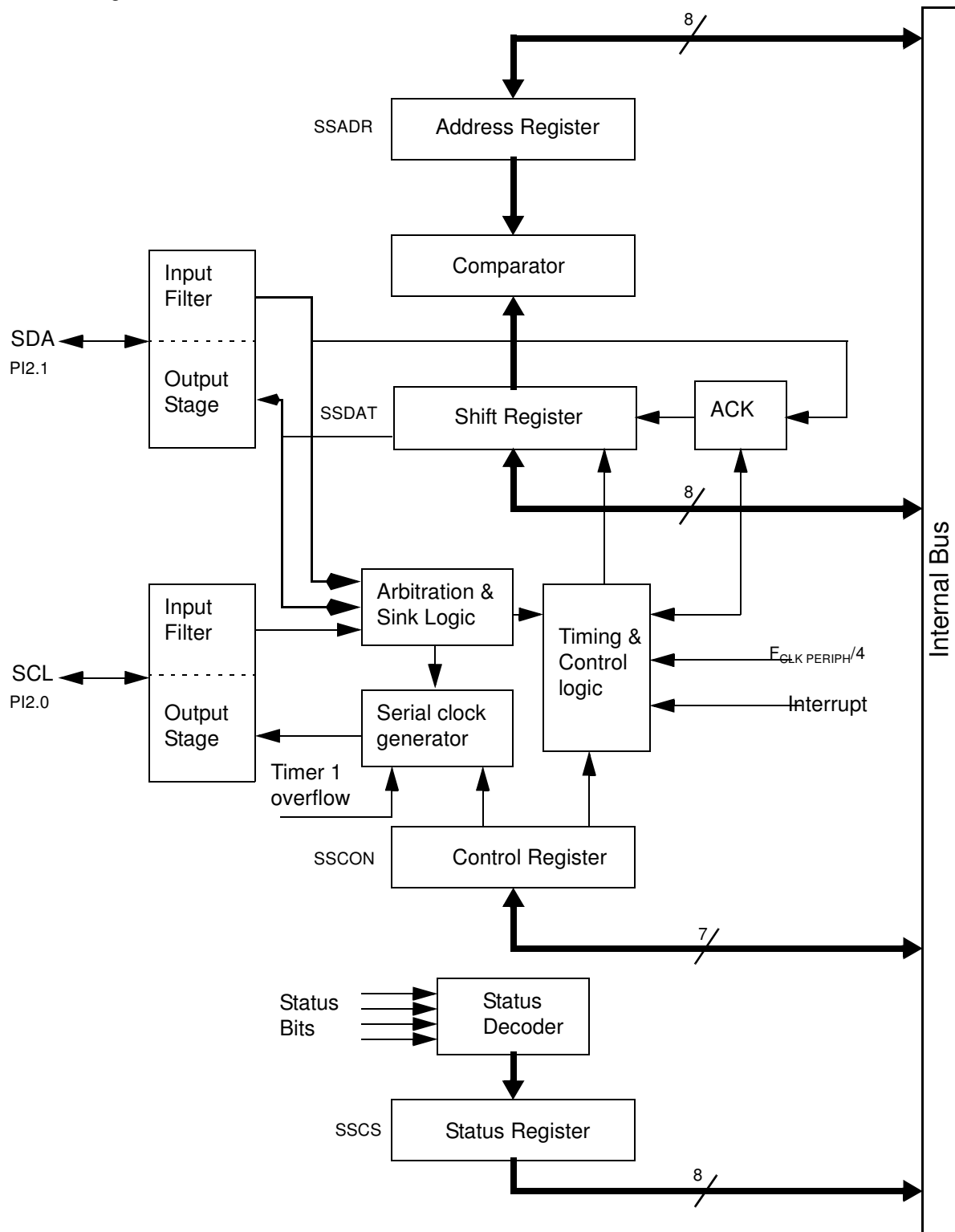
**Description**

The CPU interfaces to the 2-wire logic via the following four 8-bit special function registers: the Synchronous Serial Control register (SSCON; Table 73), the Synchronous Serial Data register (SSDAT; Table 74), the Synchronous Serial Control and Status register (SSCS; Table 75) and the Synchronous Serial Address register (SSADR Table 78).

SSCON is used to enable SSLC, to program the bit rate (see Table 66), to enable slave modes, to acknowledge or not a received data, to send a START or a STOP condition on the 2-wire bus, and to acknowledge a serial interrupt. A hardware reset disables SSLC.

In write mode, SSCS is used to select the 2-wire interface and to select the bit rate source. In read mode, SSCS contains a status code which reflects the status of the 2-wire logic and the 2-wire bus. The three least significant bits are always zero. The five most significant bits contains the status code. There are 26 possible status codes. When SSCS contains F8h, no relevant state information is available and no serial interrupt is requested. A valid status code is available in SSCS one machine cycle after SI is set by hardware and is still present one machine cycle after SI has been reset by software. Table 68.to Table 72. give the status for the master modes and miscellaneous states.

SSDAT contains a byte of serial data to be transmitted or a byte which has just been received. It is addressable while it is not in process of shifting a byte. This occurs when 2-wire logic is in a defined state and the serial interrupt flag is set. Data in SSDAT remains stable as long as SI is set. While data is being shifted out, data on the bus is simultaneously shifted in; SSDAT always contains the last byte present on the bus.

SSADR may be loaded with the 7-bit slave address (7 most significant bits) to which SSLC will respond when programmed as a slave transmitter or receiver. The LSB is used to enable general call address (00h) recognition.

Figure 37 shows how a data transfer is accomplished on the 2-wire bus.

**Figure 37.** Complete data transfer on 2-wire bus
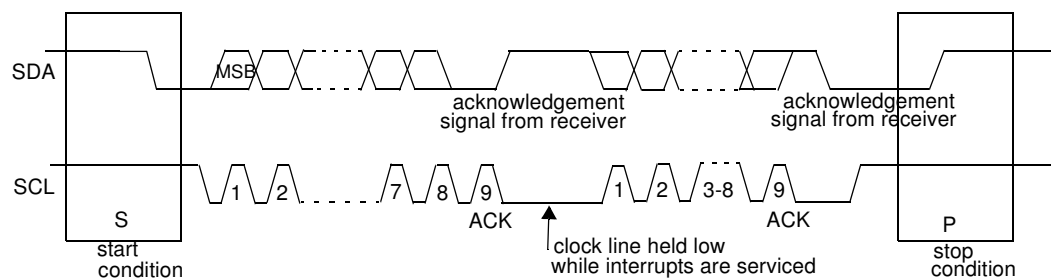


The four operating modes are:
- Master Transmitter
- Master Receiver
- Slave transmitter
- Slave receiver

Data transfer in each mode of operation is shown in Table 68 to Table 72 and Figure 38. to Figure 41.. These figures contain the following abbreviations:

S : START condition

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 Figure 38 to Figure 41, circles are used to indicate when the serial interrupt flag is set. The numbers in the circles show the status code held in SSCS. 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 the serial interrupt routine is entered, the status code in SSCS 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 Table 68 to Table 72.

**Master Transmitter Mode**

In the master transmitter mode, a number of data bytes are transmitted to a slave receiver (Figure 38). Before the master transmitter mode can be entered, SSCON must be initialised as follows:

**Table 64.** SSCON Initialization

| CR2 | SSIE | STA | STO | SI | AA | CR1 | CR0 |
|---|---|---|---|---|---|---|---|
| bit rate | 1 | 0 | 0 | 0 | X | bit rate | bit rate |

CR0, CR1 and CR2 define the internal serial bit rate if external bit rate generator is not used. SSIE must be set to enable SSLC. STA, STO and SI must be cleared.

The master transmitter mode may now be entered by setting the STA bit. The 2-wire logic will now test the 2-wire bus and generate a START condition as soon as the bus becomes free. When a START condition is transmitted, the serial interrupt flag (SI bit in SSCON) is set, and the status code in SSCS will be 08h. This status must be used to vector to an interrupt routine that loads SSDAT with the slave address and the data direction bit (SLA+W).

When the slave address and the direction bit have been transmitted and an acknowledgement bit has been received, SI is set again and a number of status code in SSCS 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 code is detailed in Table 68. This scheme is repeated until a STOP condition is transmitted.

SSIE, CR2, CR1 and CR0 are not affected by the serial transfer and are referred to Table 7 to Table 11. After a repeated START condition (state 10h) SSLC may switch to the master receiver mode by loading SSDAT with SLA+R.

**Master Receiver Mode**

In the master receiver mode, a number of data bytes are received from a slave transmitter (Figure 39). The transfer is initialized as in the master transmitter mode. When the START condition has been transmitted, the interrupt routine must load SSDAT with the 7-bit slave address and the data direction bit (SLA+R). The serial interrupt flag SI must then be cleared before the serial transfer can continue.

When the slave address and the direction bit have been transmitted and an acknowledgement bit has been received, the serial interrupt flag is set again and a number of status code in SSCS are possible. There 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 code is detailed in Table 69. This scheme is repeated until a STOP condition is transmitted.

SSIE, CR2, CR1 and CR0 are not affected by the serial transfer and are referred to Table 7 to Table 11. After a repeated START condition (state 10h) SSLC may switch to the master transmitter mode by loading SSDAT with SLA+W.

**Slave Receiver Mode**    In the slave receiver mode, a number of data bytes are received from a master transmitter (Figure 40). To initiate the slave receiver mode, SSADR and SSCON must be loaded as follows:

**Table 65.** SSADR: slave receiver mode initialization

| A6 | A5 | A4 | A3 | A2 | A1 | A0 | GC |
|----|----|----|----|----|----|----|----|
| own slave address | | | | | | | |

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

**Table 66.** SSCON: slave receiver mode initialization

| CR2 | SSIE | STA | STO | SI | AA | CR1 | CR0 |
|-----|------|-----|-----|----|----|-----|-----|
| bit rate | 1 | 0 | 0 | 0 | 1 | bit rate | bit rate |

CR0, CR1 and CR2 have no effect in the slave mode. SSIE must be set to enable SSLC. The AA bit must be set to enable the own slave address or the general call address acknowledgement. STA, STO and SI must be cleared.

When SSADR and SSCON have been initialised, SSLC waits until it is addressed by its own slave address followed by the data direction bit which must be at logic 0 (W) for SSLC to operate in the slave receiver mode. After its own slave address and the W bit have been received, the serial interrupt flag is set and a valid status code can be read from SSCS. This status code is used to vector to an interrupt service routine.The appropriate action to be taken for each of these status code is detailed in Table 70. The slave receiver mode may also be entered if arbitration is lost while SSLC is in the master mode (states 68h and 78h).

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

**Slave Transmitter Mode**    In the slave transmitter mode, a number of data bytes are transmitted to a master receiver (Figure 41). Data transfer is initialized as in the slave receiver mode. When SSADR and SSCON have been initialized, SSLC waits until it is addressed by its own

slave address followed by the data direction bit which must be at logic 1 (R) for SSLC to operate in the slave transmitter mode. After its own slave address and the R bit have been received, the serial interrupt flag is set and a valid status code can be read from SSCS. This status code is used to vector to an interrupt service routine. The appropriate action to be taken for each of these status code is detailed in Table 71. The slave transmitter mode may also be entered if arbitration is lost while SSLC is in the master mode.

If the AA bit is reset during a transfer, SSLC will transmit the last byte of the transfer and enter state C0h or C8h. SSLC is switched to the not addressed slave mode and will ignore the master receiver if it continues the transfer. Thus the master receiver receives all 1's as serial data. While AA is reset, SSLC does not respond to its own slave address. However, the 2-wire bus is still monitored and address recognition may be resume at any time by setting AA. This means that the AA bit may be used to temporarily isolate SSLC from the 2-wire bus.

**Miscellaneous States**

There are two SSCS codes that do not correspond to a define SSLC hardware state (Table 72 ). These codes are discuss hereafter.

Status F8h indicates that no relevant information is available because the serial interrupt flag is not set yet. This occurs between other states and when SSLC is not involved in a serial transfer.

Status 00h indicates that a bus error has occurred during an SSLC serial transfer. A bus error is caused when a START or a STOP condition occurs at an illegal position in the format frame. Examples of such illegal positions happen during the serial transfer of an address byte, a data byte, or an acknowledge bit. When a bus error occurs, SI is set. To recover from a bus error, the STO flag must be set and SI must be cleared. This causes SSLC to enter the not addressed slave mode and to clear the STO flag (no other bits in SSCON are affected). The SDA and SCL lines are released and no STOP condition is transmitted.

**Notes**

SSLC interfaces to the external 2-wire bus via two port pins: SCL (serial clock line) and SDA (serial data line). To avoid low level asserting on these lines when SSLC is enabled, the output latches of SDA and SLC must be set to logic 1.

**Table 67.** Bit frequency configuration

| | | | Bit Frequency ( kHz) | | |
|---|---|---|---|---|---|
| CR2 | CR1 | CR0 | $F_{OSCA}$= 12 MHz | $F_{OSCA}$ = 16 MHz | $F_{OSCA}$ divided by |
| 0 | 0 | 0 | 47 | 62.5 | 256 |
| 0 | 0 | 1 | 53.5 | 71.5 | 224 |
| 0 | 1 | 0 | 62.5 | 83 | 192 |
| 0 | 1 | 1 | 75 | 100 | 160 |
| 1 | 0 | 0 | - | - | Unused |
| 1 | 0 | 1 | 100 | 133.3 | 120 |
| 1 | 1 | 0 | 200 | 266.6 | 60 |
| 1 | 1 | 1 | 0.5 <. < 62.5 | 0.67 <. < 83 | 96 · (256 - reload valueTimer 1) (reload value range: 0-254 in mode 2) |

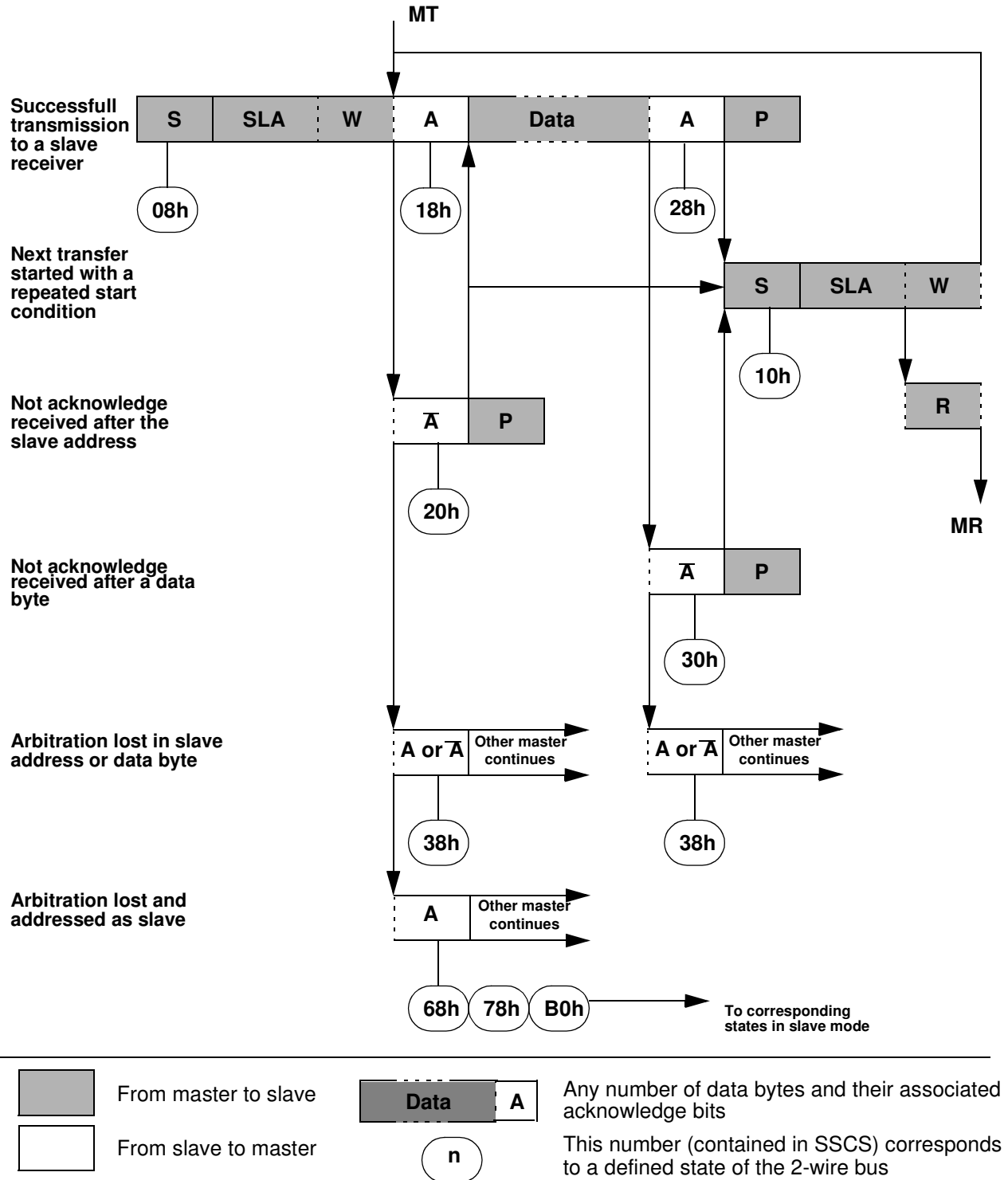**Figure 38.** Format and State in the Master Transmitter Mode

**Table 68.** Status in master transmitter mode

| Status Code SSSTA | Status of the Two-wire Bus and Two-wire Hardware | Application software response | | | | | Next Action Taken by Two-wire Hardware |
|---|---|---|---|---|---|---|---|
| | | To/From SSDAT | To SSCON | | | | |
| | | | SSSTA | SSSTO | SSI | SSAA | |
| 08h | A START condition has been transmitted | Write SLA+W | X | 0 | 0 | X | SLA+W will be transmitted. |
| 10h | A repeated START condition has been transmitted | Write SLA+W | X | 0 | 0 | X | SLA+W will be transmitted. |
| | | Write SLA+R | X | 0 | 0 | X | SLA+R will be transmitted.<br>Logic will switch to master receiver mode |
| 18h | SLA+W has been transmitted; ACK has been received | Write data byte | 0 | 0 | 0 | X | Data byte will be transmitted. |
| | | No SSDAT action | 1 | 0 | 0 | X | Repeated START will be transmitted. |
| | | No SSDAT action | 0 | 1 | 0 | X | STOP condition will be transmitted and SSSTO flag will be reset. |
| | | No SSDAT action | 1 | 1 | 0 | X | STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset. |
| 20h | SLA+W has been transmitted; NOT ACK has been received | Write data byte | 0 | 0 | 0 | X | Data byte will be transmitted. |
| | | No SSDAT action | 1 | 0 | 0 | X | Repeated START will be transmitted. |
| | | No SSDAT action | 0 | 1 | 0 | X | STOP condition will be transmitted and SSSTO flag will be reset. |
| | | No SSDAT action | 1 | 1 | 0 | X | STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset. |
| 28h | Data byte has been transmitted; ACK has been received | Write data byte | 0 | 0 | 0 | X | Data byte will be transmitted. |
| | | No SSDAT action | 1 | 0 | 0 | X | Repeated START will be transmitted. |
| | | No SSDAT action | 0 | 1 | 0 | X | STOP condition will be transmitted and SSSTO flag will be reset. |
| | | No SSDAT action | 1 | 1 | 0 | X | STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset. |
| 30h | Data byte has been transmitted; NOT ACK has been received | Write data byte | 0 | 0 | 0 | X | Data byte will be transmitted. |
| | | No SSDAT action | 1 | 0 | 0 | X | Repeated START will be transmitted. |
| | | No SSDAT action | 0 | 1 | 0 | X | STOP condition will be transmitted and SSSTO flag will be reset. |
| | | No SSDAT action | 1 | 1 | 0 | X | STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset. |
| 38h | Arbitration lost in SLA+W or data bytes | No SSDAT action | 0 | 0 | 0 | X | Two-wire bus will be released and not addressed slave mode will be entered. |
| | | No SSDAT action | 1 | 0 | 0 | X | A START condition will be transmitted when the bus becomes free. |

**Figure 39.** Format and State in the Master Receiver Mode
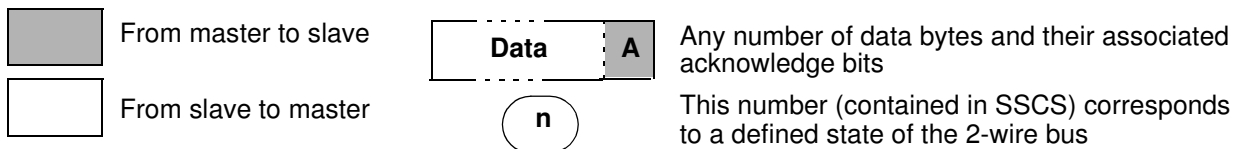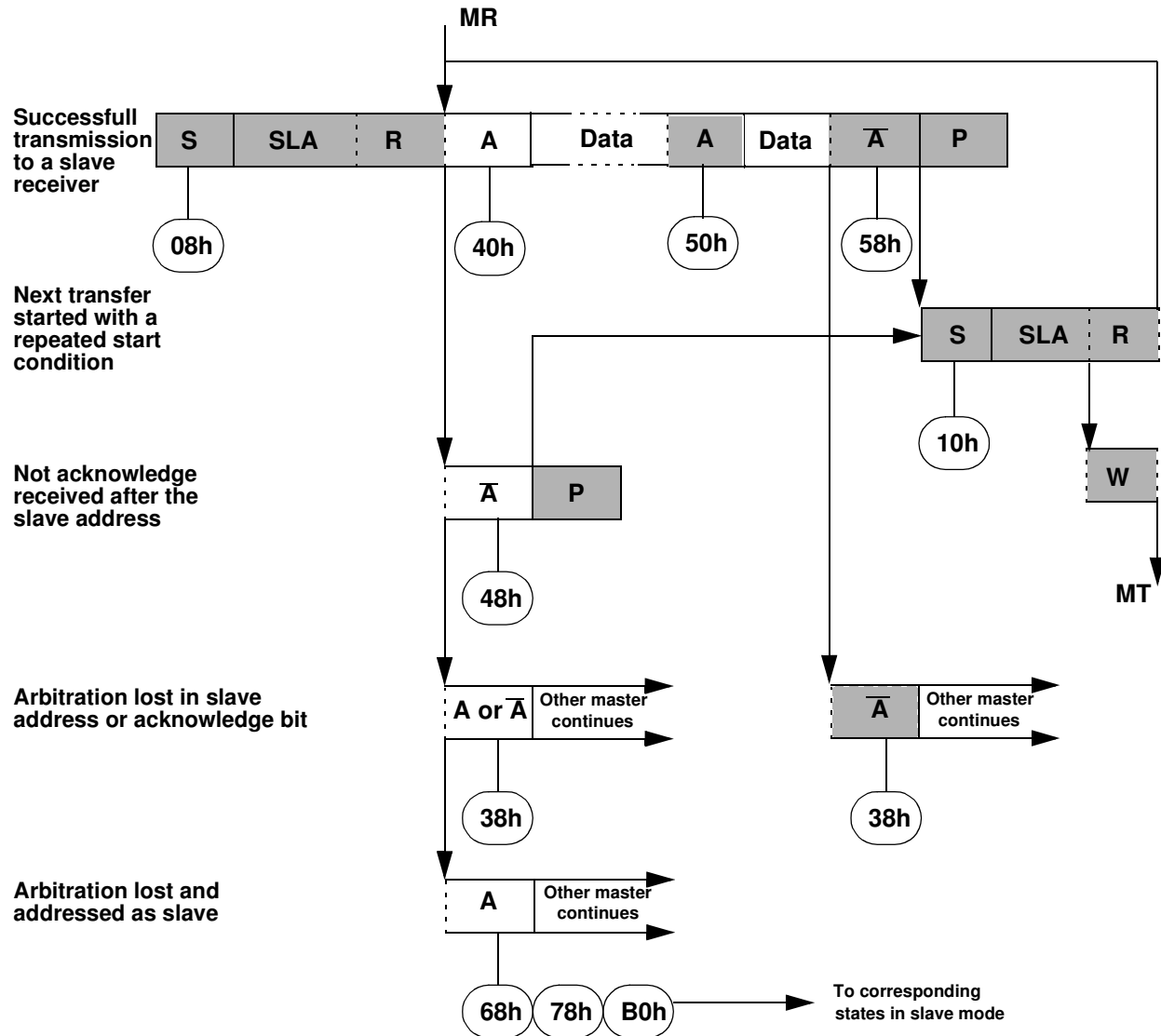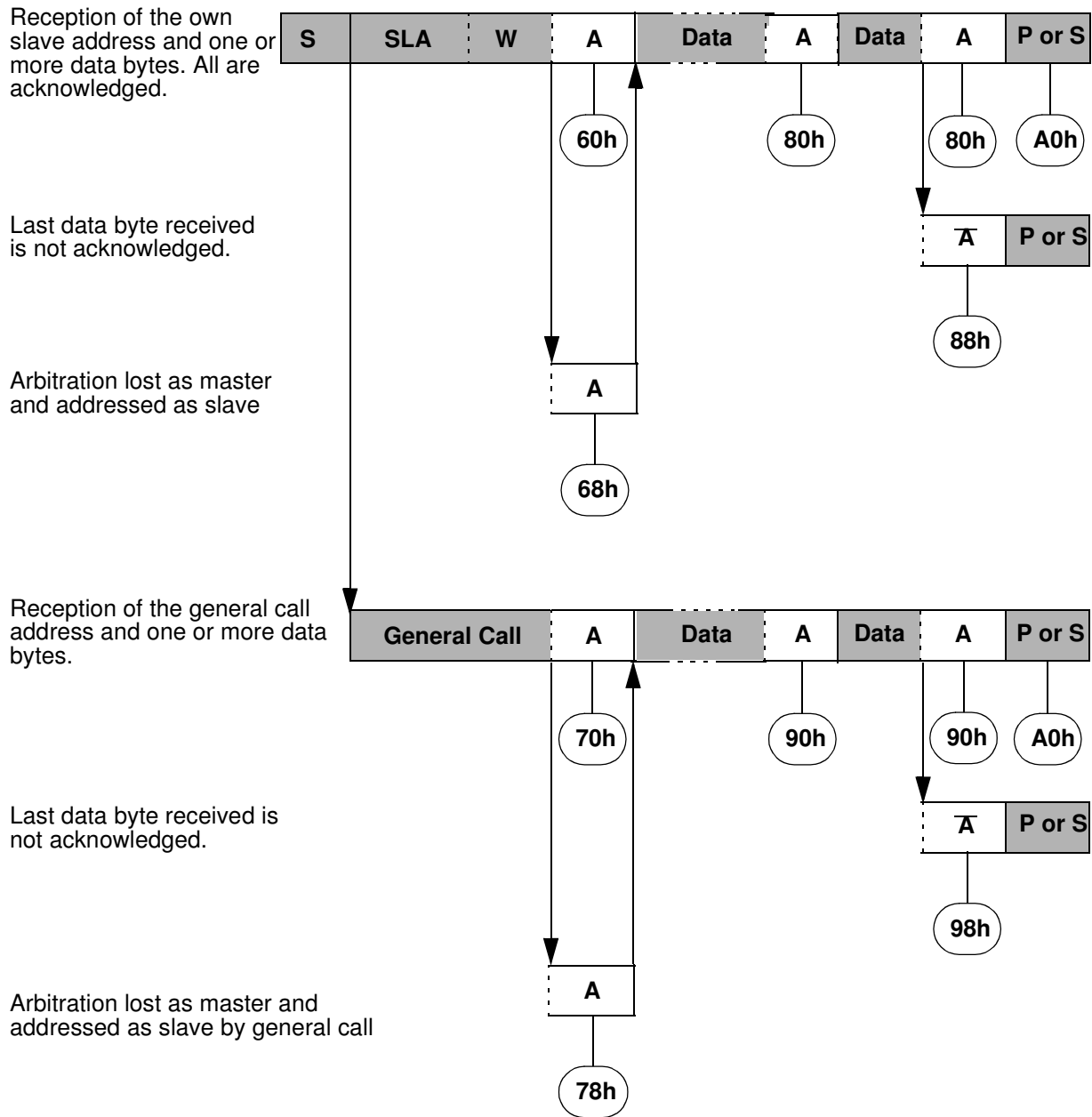
4301D–8051–02/08

**Table 69.** Status in master receiver mode

| Status Code SSSTA | Status of the Two-wire Bus and Two-wire Hardware | Application software response | | | | | Next Action Taken by Two-wire Hardware |
| | | To/From SSDAT | To SSCON | | | | |
| | | | SSSTA | SSSTO | SSI | SSAA | |
| 08h | A START condition has been transmitted | Write SLA+R | X | 0 | 0 | X | SLA+R will be transmitted. |
| 10h | A repeated START condition has been transmitted | Write SLA+R | X | 0 | 0 | X | SLA+R will be transmitted. |
| | | Write SLA+W | X | 0 | 0 | X | SLA+W will be transmitted. Logic will switch to master transmitter mode. |
| 38h | Arbitration lost in SLA+R or NOT ACK bit | No SSDAT action | 0 | 0 | 0 | X | Two-wire bus will be released and not addressed slave mode will be entered. |
| | | No SSDAT action | 1 | 0 | 0 | X | A START condition will be transmitted when the bus becomes free. |
| 40h | SLA+R has been transmitted; ACK has been received | No SSDAT action | 0 | 0 | 0 | 0 | Data byte will be received and NOT ACK will be returned. |
| | | No SSDAT action | 0 | 0 | 0 | 1 | Data byte will be received and ACK will be returned. |
| 48h | SLA+R has been transmitted; NOT ACK has been received | No SSDAT action | 1 | 0 | 0 | X | Repeated START will be transmitted. |
| | | No SSDAT action | 0 | 1 | 0 | X | STOP condition will be transmitted and SSSTO flag will be reset. |
| | | No SSDAT action | 1 | 1 | 0 | X | STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset. |
| 50h | Data byte has been received; ACK has been returned | Read data byte | 0 | 0 | 0 | 0 | Data byte will be received and NOT ACK will be returned. |
| | | Read data byte | 0 | 0 | 0 | 1 | Data byte will be received and ACK will be returned. |
| 58h | Data byte has been received; NOT ACK has been returned | Read data byte | 1 | 0 | 0 | X | Repeated START will be transmitted. |
| | | Read data byte | 0 | 1 | 0 | X | STOP condition will be transmitted and SSSTO flag will be reset. |
| | | Read data byte | 1 | 1 | 0 | X | STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset. |

**Figure 40.** Format and State in the Slave Receiver Mode

Reception of the own
slave address and one or
more data bytes. All are
acknowledged.

| S | SLA | W | A | Data | A | Data | A | P or S |

60h — 80h — 80h — A0h

Last data byte received
is not acknowledged.

A̅ | P or S

88h

Arbitration lost as master
and addressed as slave

A

68h

Reception of the general call
address and one or more data
bytes.

| General Call | A | Data | A | Data | A | P or S |

70h — 90h — 90h — A0h

Last data byte received is
not acknowledged.

A̅ | P or S

98h

Arbitration lost as master and
addressed as slave by general call

A

78h

From master to slave

From slave to master

| Data | A | Any number of data bytes and their associated acknowledge bits

n  This number (contained in SSCS) corresponds to a defined state of the 2-wire bus

**Table 70.** Status in slave receiver mode

| Status Code (SSCS) | Status of the 2-wire bus and 2-wire hardware | Application Software Response | | | | | | Next Action Taken By 2-wire Software |
|---|---|---|---|---|---|---|---|---|
| | | To/from SSDAT | To SSCON | | | | | |
| | | | STA | STO | SI | AA | | |
| 60h | Own SLA+W has been received; ACK has been returned | No SSDAT action or | X | 0 | 0 | 0 | | Data byte will be received and NOT ACK will be returned |
| | | No SSDAT action | X | 0 | 0 | 1 | | Data byte will be received and ACK will be returned |
| 68h | Arbitration lost in SLA+R/W as master; own SLA+W has been received; ACK has been returned | No SSDAT action or | X | 0 | 0 | 0 | | Data byte will be received and NOT ACK will be returned |
| | | No SSDAT action | X | 0 | 0 | 1 | | Data byte will be received and ACK will be returned |
| 70h | General call address has been received; ACK has been returned | No SSDAT action or | X | 0 | 0 | 0 | | Data byte will be received and NOT ACK will be returned |
| | | No SSDAT action | X | 0 | 0 | 1 | | Data byte will be received and ACK will be returned |
| 78h | Arbitration lost in SLA+R/W as master; general call address has been received; ACK has been returned | No SSDAT action or | X | 0 | 0 | 0 | | Data byte will be received and NOT ACK will be returned |
| | | No SSDAT action | X | 0 | 0 | 1 | | Data byte will be received and ACK will be returned |
| 80h | Previously addressed with own SLA+W; data has been received; ACK has been returned | No SSDAT action or | X | 0 | 0 | 0 | | Data byte will be received and NOT ACK will be returned |
| | | No SSDAT action | X | 0 | 0 | 1 | | Data byte will be received and ACK will be returned |
| 88h | Previously addressed with own SLA+W; data has been received; NOT ACK has been returned | Read data byte or | 0 | 0 | 0 | 0 | | Switched to the not addressed slave mode; no recognition of own SLA or GCA |
| | | Read data byte or | 0 | 0 | 0 | 1 | | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1 |
| | | Read data byte or | 1 | 0 | 0 | 0 | | Switched to the not addressed slave mode; no recognition of own SLA or GCA. A START condition will be transmitted when the bus becomes free |
| | | Read data byte | 1 | 0 | 0 | 1 | | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1. A START condition will be transmitted when the bus becomes free |
| 90h | Previously addressed with general call; data has been received; ACK has been returned | Read data byte or | X | 0 | 0 | 0 | | Data byte will be received and NOT ACK will be returned |
| | | Read data byte | X | 0 | 0 | 1 | | Data byte will be received and ACK will be returned |

| Status Code (SSCS) | Status of the 2-wire bus and 2-wire hardware | Application Software Response | | | | | | Next Action Taken By 2-wire Software |
|---|---|---|---|---|---|---|---|---|
| | | To/from SSDAT | To SSCON | | | | | |
| | | | STA | STO | SI | AA | | |
| 98h | Previously addressed with general call; data has been received; NOT ACK has been returned | Read data byte or | 0 | 0 | 0 | 0 | | Switched to the not addressed slave mode; no recognition of own SLA or GCA |
| | | Read data byte or | 0 | 0 | 0 | 1 | | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1 |
| | | Read data byte or | 1 | 0 | 0 | 0 | | Switched to the not addressed slave mode; no recognition of own SLA or GCA. A START condition will be transmitted when the bus becomes free |
| | | Read data byte | 1 | 0 | 0 | 1 | | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1. A START condition will be transmitted when the bus becomes free |
| A0h | A STOP condition or repeated START condition has been received while still addressed as slave | No SSDAT action or | 0 | 0 | 0 | 0 | | Switched to the not addressed slave mode; no recognition of own SLA or GCA |
| | | No SSDAT action or | 0 | 0 | 0 | 1 | | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1 |
| | | No SSDAT action or | 1 | 0 | 0 | 0 | | Switched to the not addressed slave mode; no recognition of own SLA or GCA. A START condition will be transmitted when the bus becomes free |
| | | No SSDAT action | 1 | 0 | 0 | 1 | | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1. A START condition will be transmitted when the bus becomes free |

**Figure 41.** Format and State in the Slave Transmitter Mode
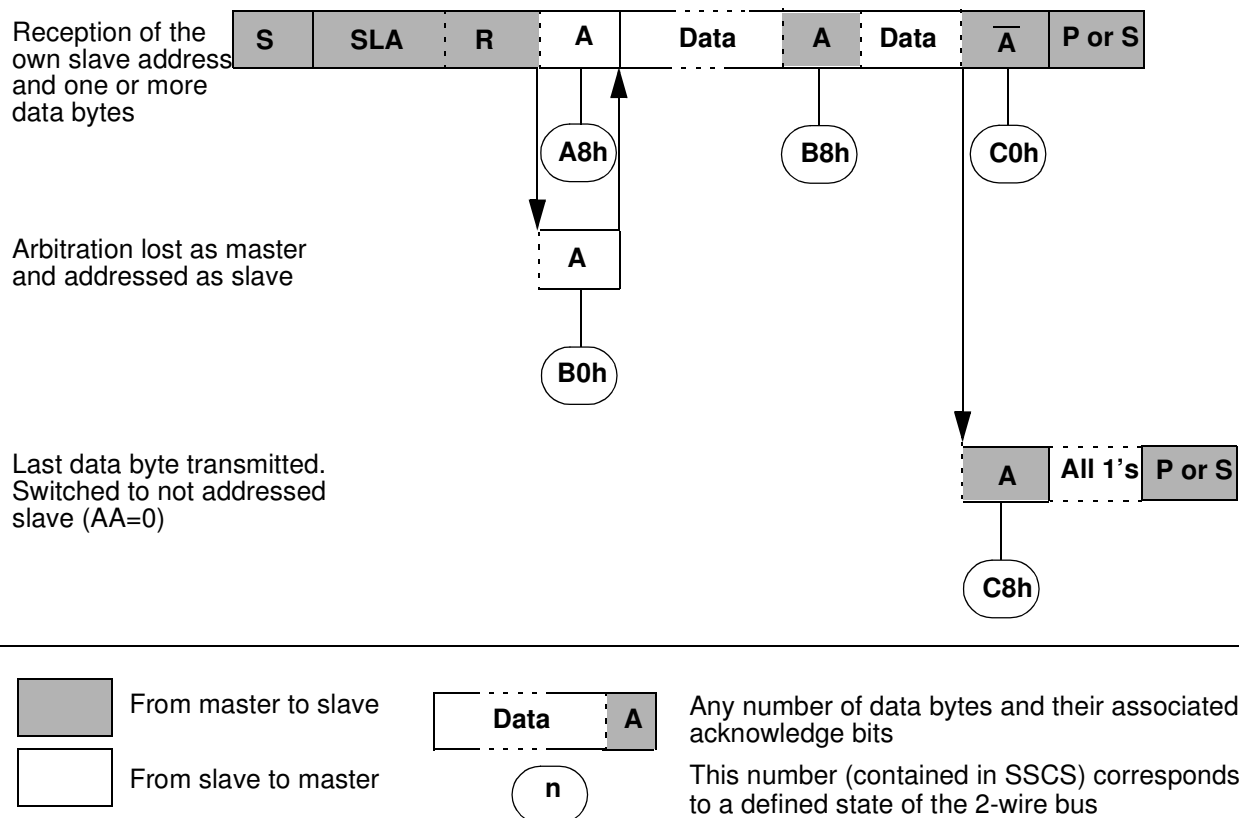


**Table 71.** Status in slave transmitter mode

| Status Code (SSCS) | Status of the 2-wire bus and 2-wire hardware | Application Software Response | | | | | |
|---|---|---|---|---|---|---|---|
| | | To/from SSDAT | To SSCON | | | | Next Action Taken By 2-wire Software |
| | | | STA | STO | SI | AA | |
| A8h | Own SLA+R has been received; ACK has been returned | Load data byte or | X | 0 | 0 | 0 | Last data byte will be transmitted and NOT ACK will be received |
| | | Load data byte | X | 0 | 0 | 1 | Data byte will be transmitted and ACK will be received |
| B0h | Arbitration lost in SLA+R/W as master; own SLA+R has been received; ACK has been returned | Load data byte or | X | 0 | 0 | 0 | Last data byte will be transmitted and NOT ACK will be received |
| | | Load data byte | X | 0 | 0 | 1 | Data byte will be transmitted and ACK will be received |
| B8h | Data byte in SSDAT has been transmitted; NOT ACK has been received | Load data byte or | X | 0 | 0 | 0 | Last data byte will be transmitted and NOT ACK will be received |
| | | Load data byte | X | 0 | 0 | 1 | Data byte will be transmitted and ACK will be received |

| Status Code (SSCS) | Status of the 2-wire bus and 2-wire hardware | Application Software Response | | | | | Next Action Taken By 2-wire Software |
|---|---|---|---|---|---|---|---|
| | | To/from SSDAT | To SSCON | | | | |
| | | | STA | STO | SI | AA | |
| C0h | Data byte in SSDAT has been transmitted; NOT ACK has been received | No SSDAT action or | 0 | 0 | 0 | 0 | Switched to the not addressed slave mode; no recognition of own SLA or GCA |
| | | No SSDAT action or | 0 | 0 | 0 | 1 | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1 |
| | | No SSDAT action or | 1 | 0 | 0 | 0 | Switched to the not addressed slave mode; no recognition of own SLA or GCA. A START condition will be transmitted when the bus becomes free |
| | | No SSDAT action | 1 | 0 | 0 | 1 | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1. A START condition will be transmitted when the bus becomes free |
| C8h | Last data byte in SSDAT has been transmitted (AA=0); ACK has been received | No SSDAT action or | 0 | 0 | 0 | 0 | Switched to the not addressed slave mode; no recognition of own SLA or GCA |
| | | No SSDAT action or | 0 | 0 | 0 | 1 | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1 |
| | | No SSDAT action or | 1 | 0 | 0 | 0 | Switched to the not addressed slave mode; no recognition of own SLA or GCA. A START condition will be transmitted when the bus becomes free |
| | | No SSDAT action | 1 | 0 | 0 | 1 | Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1. A START condition will be transmitted when the bus becomes free |

**Table 72.** Miscellaneous status

| Status Code (SSCS) | Status of the 2-wire bus and 2-wire hardware | Application Software Response | | | | | Next Action Taken By 2-wire Software |
|---|---|---|---|---|---|---|---|
| | | To/from SSDAT | To SSCON | | | | |
| | | | STA | STO | SI | AA | |
| F8h | No relevant state information available; SI= 0 | No SSDAT action | No SSCON action | | | | Wait or proceed current transfer |
| 00h | Bus error due to an illegal START or STOP condition | No SSDAT action | 0 | 1 | 0 | X | Only the internal hardware is affected, no STOP condition is sent on the bus. In all cases, the bus is released and STO is reset. |

**Registers**

**Table 73.** SSCON Register

SSCON - Synchronous Serial Control register (93h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CR2 | SSIE | STA | STO | SI | AA | CR1 | CR0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | CR2 | **Control Rate bit 2**<br>See Table 67. |
| 6 | SSIE | **Synchronous Serial Interface Enable bit**<br>Clear to disable SSLC.<br>Set to enable SSLC. |
| 5 | STA | **Start flag**<br>Set to send a START condition on the bus. |
| 4 | ST0 | **Stop flag**<br>Set to send a STOP condition on the bus. |
| 3 | SI | **Synchronous Serial Interrupt flag**<br>Set by hardware when a serial interrupt is requested.<br>Must be cleared by software to acknowledge interrupt. |
| 2 | AA | **Assert Acknowledge flag**<br>Clear in master and slave receiver modes, to force a not acknowledge (high level on SDA).<br>Clear to disable SLA or GCA recognition.<br>Set to recognise SLA or GCA (if GC set) for entering slave receiver or transmitter modes.<br>Set in master and slave receiver modes, to force an acknowledge (low level on SDA).<br>This bit has no effect when in master transmitter mode. |
| 1 | CR1 | **Control Rate bit 1**<br>See Table 67. |
| 0 | CR0 | **Control Rate bit 0**<br>See Table 67. |

**Table 74.** SSDAT (095h) - Syncrhonous Serial Data register (read/write)

| SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | SD7 | Address bit 7 or Data bit 7. |
| 6 | SD6 | Address bit 6 or Data bit 6. |
| 5 | SD5 | Address bit 5 or Data bit 5. |
| 4 | SD4 | Address bit 4 or Data bit 4. |
| 3 | SD3 | Address bit 3 or Data bit 3. |
| 2 | SD2 | Address bit 2 or Data bit 2. |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 1 | SD1 | Address bit 1 or Data bit 1. |
| 0 | SD0 | Address bit 0 (R/W) or Data bit 0. |

**Table 75.** SSCS (094h) read - Synchronous Serial Control and Status Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SC4 | SC3 | SC2 | SC1 | SC0 | 0 | 0 | 0 |

**Table 76.** SSCS Register: Read Mode - Reset Value = F8h

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 0 | 0 | Always zero |
| 1 | 0 | Always zero |
| 2 | 0 | Always zero |
| 3 | SC0 | **Status Code bit 0** See Table 68.to Table 72. |
| 4 | SC1 | **Status Code bit 1** See Table 68.to Table 72. |
| 5 | SC2 | **Status Code bit 2** See Table 68.to Table 72. |
| 6 | SC3 | **Status Code bit 3** See Table 68.to Table 72. |
| 7 | SC4 | **Status Code bit 4** See Table 68.to Table 72. |

**Table 77.** SSADR (096h) - Synchronus Serial Address Register (read/write)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |

**Table 78.** SSADR Register - Reset value = FEh

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | A7 | Slave Address bit 7 |
| 6 | A6 | Slave Address bit 6 |
| 5 | A5 | Slave Address bit 5 |
| 4 | A4 | Slave Address bit 4 |
| 3 | A3 | Slave Address bit 3 |
| 2 | A2 | Slave Address bit 2 |
| 1 | A1 | Slave Address bit 1 |
| 0 | GC | General Call bit<br>Clear to disable the general call address recognition.<br>Set to enable the general call address recognition. |

## Hardware Watchdog Timer

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upset. The WDT consists of a 14-bit counter and the WatchDog Timer ReSeT (WDTRST) SFR. The WDT is by default disabled from exiting reset. To enable the WDT, user must write 01EH and 0E1H in sequence to the WDTRST, SFR location 0A6H. When WDT is enabled, it will increment every machine cycle while the oscillator is running and there is no way to disable the WDT except through reset (either hardware reset or WDT overflow reset). When WDT overflows, it will drive an output RESET HIGH pulse at the RST-pin.

## Using the WDT

To enable the WDT, user must write 01EH and 0E1H in sequence to the WDTRST, SFR location 0A6H. When WDT is enabled, the user needs to service it by writing to 01EH and 0E1H to WDTRST to avoid WDT overflow. The 14-bit counter overflows when it reaches 16383 (3FFFH) and this will reset the device. When WDT is enabled, it will increment every machine cycle while the oscillator is running. This means the user must reset the WDT at least every 16383 machine cycle. To reset the WDT the user must write 01EH and 0E1H to WDTRST. WDTRST is a write only register. The WDT counter cannot be read or written. When WDT overflows, it will generate an output RESET pulse at the RST-pin. The RESET pulse duration is 96 x $T_{CLK\ PERIPH}$, where $T_{CLK\ PERIPH}= 1/F_{CLK\ PERIPH}$. To make the best use of the WDT, it should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset.

To have a more powerful WDT, a $2^7$ counter has been added to extend the Time-out capability, ranking from 16ms to 2s @ $F_{OSCA}$ = 12MHz. To manage this feature, refer to WDTPRG register description, Table 79.

**Table 79.** WDTRST Register

WDTRST - Watchdog Reset Register (0A6h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

Reset Value = XXXX XXXXb

Write only, this SFR is used to reset/enable the WDT by writing 01EH then 0E1H in sequence.

**Table 80.**  WDTPRG Register

WDTPRG - Watchdog Timer Out Register (0A7h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | S2 | S1 | S0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | |
| 6 | - | |
| 5 | - | **Reserved** <br> The value read from this bit is undetermined. Do not try to set this bit. |
| 4 | - | |
| 3 | - | |
| 2 | S2 | **WDT Time-out select bit 2** |
| 1 | S1 | **WDT Time-out select bit 1** |
| 0 | S0 | **WDT Time-out select bit 0** |
| | | <u>S2</u>  <u>S1</u> <u>S0 Selected Time-out</u> <br> 0    00    $(2^{14} - 1)$ machine cycles, 16. 3 ms @ $F_{OSCA}$ =12 MHz <br> 0    01    $(2^{15} - 1)$ machine cycles, 32.7 ms @ $F_{OSCA}$=12 MHz <br> 0    10 $(2^{16} - 1)$ machine cycles, 65. 5 ms @ $F_{OSCA}$=12 MHz <br> 0    11    $(2^{17} - 1)$ machine cycles, 131 ms @ $F_{OSCA}$=12 MHz <br> 1    00    $(2^{18} - 1)$ machine cycles, 262 ms @ $F_{OSCA}$=12 MHz <br> 1    01 $(2^{19} - 1)$ machine cycles, 542 ms @ $F_{OSCA}$=12 MHz <br> 1    10    $(2^{20} - 1)$ machine cycles, 1.05 s @ $F_{OSCA}$=12 MHz <br> 1    11    $(2^{21} - 1)$ machine cycles, 2.09 s @ $F_{OSCA}$=12 MHz |

Reset value XXXX X000

**WDT During Power Down and Idle**

In Power Down mode the oscillator stops, which means the WDT also stops. While in Power Down mode the user does not need to service the WDT. There are 2 methods of exiting Power Down mode:  by a hardware reset or via a level activated external interrupt which is enabled prior to entering Power Down mode. When Power Down is exited with hardware reset, servicing the WDT should occur as it normally should whenever the AT89C51IC2 is reset. Exiting Power Down with an interrupt is significantly different. The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service routine.

To ensure that the WDT does not overflow within a few states of exiting of powerdown, it is better to reset the WDT just before entering powerdown.

In the Idle mode, the oscillator continues to run. To prevent the WDT from resetting the AT89C51IC2 while in Idle mode, the user should always set up a timer that will periodically exit Idle, service the WDT, and re-enter Idle mode.

**Power-off Flag**    The power-off flag allows the user to distinguish between a "cold start" reset and a "warm start" reset.

A cold start reset is the one induced by $V_{CC}$ switch-on. A warm start reset occurs while $V_{CC}$ is still applied to the device and could be generated for example by an exit from power-down.

The power-off flag (POF) is located in PCON register (Table 81). POF is set by hardware when $V_{CC}$ rises from 0 to its nominal voltage. The POF can be set or cleared by software allowing the user to determine the type of reset.

**Table 81.** PCON Register

PCON - Power Control Register (87h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SMOD1 | SMOD0 | - | POF | GF1 | GF0 | PD | IDL |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | SMOD1 | **Serial port Mode bit 1**<br>Set to select double baud rate in mode 1, 2 or 3. |
| 6 | SMOD0 | **Serial port Mode bit 0**<br>Cleared to select SM0 bit in SCON register.<br>Set to select FE bit in SCON register. |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | POF | **Power-Off Flag**<br>Cleared to recognize next reset type.<br>Set by hardware when $V_{CC}$ rises from 0 to its nominal voltage. Can also be set by software. |
| 3 | GF1 | **General purpose Flag**<br>Cleared by user for general purpose usage.<br>Set by user for general purpose usage. |
| 2 | GF0 | **General purpose Flag**<br>Cleared by user for general purpose usage.<br>Set by user for general purpose usage. |
| 1 | PD | **Power-Down mode bit**<br>Cleared by hardware when reset occurs.<br>Set to enter power-down mode. |
| 0 | IDL | **Idle mode bit**<br>Cleared by hardware when interrupt or reset occurs.<br>Set to enter idle mode. |

Reset Value = 00X1 0000b
Not bit addressable

## ONCE(TM) Mode (ON Chip Emulation)

The ONCE mode facilitates testing and debugging of systems using AT89C51IC2 without removing the circuit from the board. The ONCE mode is invoked by driving certain pins of the AT89C51IC2; the following sequence must be exercised:

• Pull ALE low while the device is in reset (RST high) and $\overline{PSEN}$ is high.

• Hold ALE low as RST is deactivated.

While the AT89C51IC2 is in ONCE mode, an emulator or test CPU can be used to drive the circuit Table 82 shows the status of the port pins during ONCE mode.

Normal operation is restored when normal reset is applied.

**Table 82.** External Pin Status during ONCE Mode

| ALE | PSEN | Port 0 | Port 1 | Port 2 | Port 3 | Port I2 | XTALA1/2 | XTALB1/2 |
|------|------|--------|--------|--------|--------|---------|----------|----------|
| Weak pull-up | Weak pull-up | Float | Weak pull-up | Weak pull-up | Weak pull-up | Float | Active | Active |

(a) "Once" is a registered trademark of Intel Corporation.

## Reduced EMI Mode

The ALE signal is used to demultiplex address and data buses on port 0 when used with external program or data memory. Nevertheless, during internal code execution, ALE signal is still generated. In order to reduce EMI, ALE signal can be disabled by setting AO bit.

The AO bit is located in AUXR register at bit location 0. As soon as AO is set, ALE is no longer output but remains active during MOVX and MOVC instructions and external fetches. During ALE disabling, ALE pin is weakly pulled high.

**Table 83.** AUXR Register

AUXR - Auxiliary Register (8Eh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | M0 | - | XRS1 | XRS0 | EXTRAM | AO |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 6 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 5 | M0 | **Pulse length**<br>Cleared to stretch MOVX control: the RD/ and the WR/ pulse length is 6 clock periods (default).<br>Set to stretch MOVX control: the RD/ and the WR/ pulse length is 30 clock periods. |
| 4 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 3 | XRS1 | **XRAM Size** |
| 2 | XRS0 | XRS1XRS0XRAM size<br>0   0256 bytes (default)<br>0   1512 bytes<br>1   0768 bytes<br>1   11024 bytes |
| 1 | EXTRAM | **EXTRAM bit**<br>Cleared to access internal XRAM using movx @ Ri/ @ DPTR.<br>Set to access external memory.<br>Programmed by hardware after Power-up regarding Hardware Security Byte (HSB), default setting, XRAM selected. |
| 0 | AO | **ALE Output bit**<br>Cleared, ALE is emitted at a constant rate of 1/6 the oscillator frequency (or 1/3 if X2 mode is used).(default) Set, ALE is active only during a MOVX or MOVC instructione is used. |

## Flash EEPROM Memory

The Flash memory increases EPROM and ROM functionality with in-circuit electrical erasure and programming. It contains 32K Bytes of program memory organized in 128 or 256 pages of 128 Bytes. This memory is both parallel and serial In-system Programmable (ISP). ISP allows devices to alter their own program memory in the actual end product under software control. A default serial loader (bootloader) program allows ISP of the Flash.

The programming does not require external dedicated programming voltage. The necessary high programming voltage is generated on-chip using the standard $V_{CC}$ pins of the microcontroller.

## Features

- Flash EEPROM internal program memory.
- Boot vector allows user provided Flash loader code to reside anywhere in the Flash memory space. This configuration provides flexibility to the user.
- Default loader in Boot ROM allows programming via the serial port without the need of a user-provided loader.
- Up to 64K Byte external program memory if the internal program memory is disabled (EA = 0).
- Programming and erase voltage with standard 5V or 3V $V_{CC}$ supply.
- Read/Programming/Erase:
  – Byte-wise read without wait state
  – Byte or page erase and programming (10 ms)
- Typical programming time (32K Bytes) in 10 s
- Parallel programming with 87C51 compatible hardware interface to programmer
- Programmable security for the code in the Flash
- 10K write cycles
- 10 years data retention

## Flash Programming and Erasure

The 32K Bytes Flash is programmed by Bytes or by pages of 128 Bytes. It is not necessary to erase a Byte or a page before programming. The programming of a Byte or a page includes a self erase before programming.

There are three methods of programming the Flash memory:

- First, the on-chip ISP bootloader may be invoked which will use low level routines to program the pages. The interface used for serial downloading of Flash is the UART.
- Second, the Flash may be programmed or erased in the end-user application by calling low-level routines through a common entry point in the Boot ROM.
- Third, the Flash may be programmed using the parallel method by using a conventional EPROM programmer. The parallel programming method used by these devices is similar to that used by EPROM 87C51 but it is not identical and the commercially available programmers need to have support for the AT89C51IC2. The bootloader and the Application Programming Interface (API) routines are located in the BOOT ROM.

**Flash Registers and Memory Map**

The AT89C51IC2 Flash memory uses several registers for its management:

- Hardware registers can only be accessed through the parallel programming modes which are handled by the parallel programmer.
- Software registers are in a special page of the Flash memory which can be accessed through the API or with the parallel programming modes. This page, called "Extra Flash Memory", is not in the internal Flash program memory addressing space.

**Hardware Register**

The only hardware register of the AT89C51IC2 is called Hardware Security Byte (HSB).

**Table 84.** Hardware Security Byte (HSB)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| X2 | BLJB | OSC | - | XRAM | LB2 | LB1 | LB0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | X2 | **X2 Mode** <br> Programmed ('0' value) to force X2 mode (6 clocks per instruction) after reset. <br> Unprogrammed ('1' Value) to force X1 mode, Standard Mode, after reset (Default). |
| 6 | BLJB | **Boot Loader Jump Bit** <br> Unprogrammed ('1' value) to start the user's application on next reset at address 0000h. <br> Programmed ('0' value) to start the boot loader at address F800h on next reset (Default). |
| 5 | OSC | **Oscillator Bit** <br> Programmed to allow oscillator B at startup <br> Unprogrammed this bit to allow oscillator A at startup ( Default). |
| 4 | - | **Reserved** |
| 3 | XRAM | **XRAM config bit (only programmable by programmer tools)** <br> Programmed to inhibit XRAM <br> Unprogrammed, this bit to valid XRAM (Default) |
| 2-0 | LB2-0 | **User Memory Lock Bits (only programmable by programmer tools)** <br> See Table 85 |

Boot Loader Jump Bit (BLJB)

One bit of the HSB, the BLJB bit, is used to force the boot address:

- When this bit is programmed ('1' value) the boot address is 0000h.
- When this bit is unprogrammed ('1' value) the boot address is F800h. By default, this bit is unprogrammed and the ISP is enabled.

**Flash Memory Lock Bits**

The three lock bits provide different levels of protection for the on-chip code and data, when programmed as shown in Table 85.

**Table 85.** Program Lock Bits

| Program Lock Bits | | | | |
|---|---|---|---|---|
| Security Level | LB0 | LB1 | LB2 | Protection Description |
| 1 | U | U | U | No program lock features enabled. |
| 2 | P | U | U | MOVC instruction executed from external program memory is disabled from fetching code Bytes from internal memory, $\overline{EA}$ is sampled and latched on reset, and further parallel programming of the Flash is disabled. ISP and software programming with API are still allowed. |
| 3 | X | P | U | Same as 2, also verify through parallel programming interface is disabled. |
| 4 | X | X | P | Same as 3, also external execution is disabled. (Default) |

Note: U: unprogrammed or "one" level.
P: programmed or "zero" level.
X: don't care
WARNING: Security level '2' and '3' should only be programmed after Flash and code verification.

These security bits protect the code access through the parallel programming interface. They are set by default to level 4. The code access through the ISP is still possible and is controlled by the "software security bits" which are stored in the extra Flash memory accessed by the ISP firmware.

To load a new application with the parallel programmer, a chip erase must first be done. This will set the HSB in its inactive state and will erase the Flash memory. The part reference can always be read using Flash parallel programming modes.

**Default Values**

The default value of the HSB provides parts ready to be programmed with ISP:

• BLJB: Programmed force ISP operation.

• X2: Unprogrammed to force X1 mode (Standard Mode).

• XRAM: Unprogrammed to valid XRAM

• LB2-0: Security level four to protect the code from a parallel access with maximum security.

**Software Registers**

Several registers are used, in factory and by parallel programmers, to make copies of hardware registers contents. These values are used by Atmel ISP.

These registers are in the "Extra Flash Memory" part of the Flash memory. This block is also called "XAF" or eXtra Array Flash. They are accessed in the following ways:

• Commands issued by the parallel memory programmer.

• Commands issued by the ISP software.

• Calls of API issued by the application software.

Several software registers are described in Table 86.

**Table 86.** Default Values

| Mnemonic | Definition | Default value | Description |
|----------|-----------|---------------|-------------|
| SBV | Software Boot Vector | FCh | |
| HSB | Hardware security Byte | 101x 1011b | |
| BSB | Boot Status Byte | 0FFh | |
| SSB | Software Security Byte | FFh | |
| | Copy of the Manufacturer Code | 58h | ATMEL |
| | Copy of the Device ID #1: Family Code | D7h | C51 X2, Electrically Erasable |
| | Copy of the Device ID #2: memories | F7h | AT89C51IC2 32KB |
| | Copy of the Device ID #3: name and revision | EFh | AT89C51IC2 32KB, Revision 0 |

After programming the part by ISP, the BSB must be cleared (00h) in order to allow the application to boot at 0000h.

The content of the Software Security Byte (SSB) is described in Table 86 and Table 88.

To assure code protection from a parallel access, the HSB must also be at the required level.

**Table 87.** Software Security Byte

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | LB1 | LB0 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|-------------|
| 7 | - | **Reserved** <br> Do not clear this bit. |
| 6 | - | **Reserved** <br> Do not clear this bit. |
| 5 | - | **Reserved** <br> Do not clear this bit. |
| 4 | - | **Reserved** <br> Do not clear this bit. |
| 3 | - | **Reserved** <br> Do not clear this bit. |
| 2 | - | **Reserved** <br> Do not clear this bit. |
| 1-0 | LB1-0 | **User Memory Lock Bits** <br> see Table 88 |

The two lock bits provide different levels of protection for the on-chip code and data, when programmed as shown in Table 88.

**Table 88.** Program Lock Bits of the SSB

| Program Lock Bits | | | |
|---|---|---|---|
| Security level | LB0 | LB1 | Protection Description |
| 1 | U | U | No program lock features enabled. |
| 2 | P | U | ISP programming of the Flash is disabled. |
| 3 | X | P | Same as 2, also verify through ISP programming interface is disabled. |

Note:    U: unprogrammed or "one" level.
         P: programmed or "zero" level.
         X: don't care
         WARNING:  Security level 2 and 3 should only be programmed after Flash and code verification.

**Flash Memory Status**    AT89C51IC2 parts are delivered in standard with the ISP boot in the Flash memory. After ISP or parallel programming, the possible contents of the Flash memory are summarized on Figure 42.

**Figure 42.** Flash Memory Possible Contents
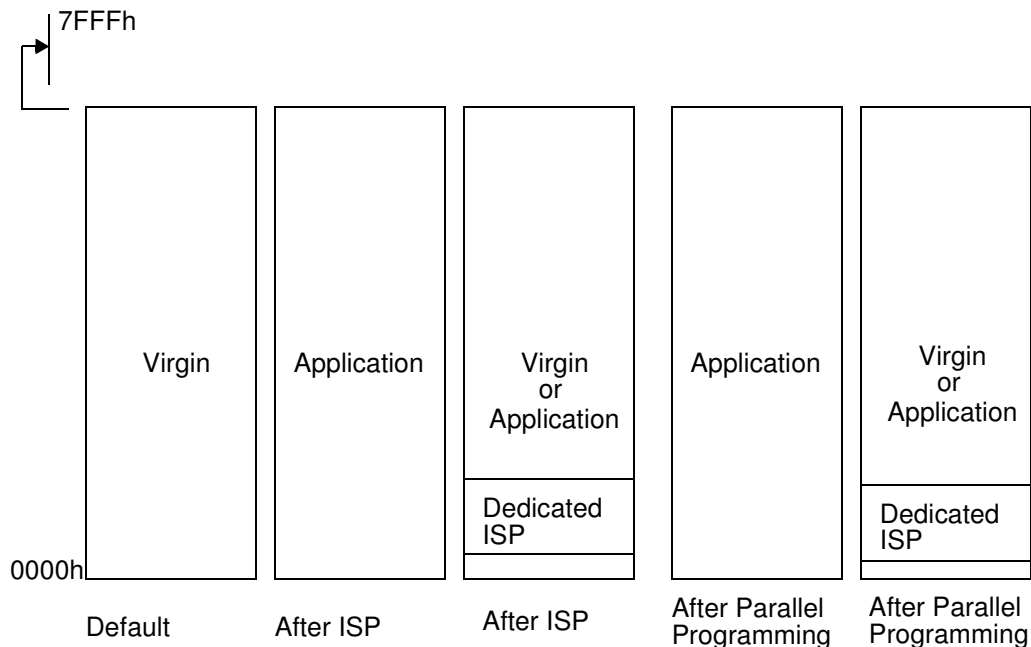


**Memory Organization**    In the AT89C51IC2, the lowest 32K of the 64 KB program memory address space is filled by internal Flash.

When the $\overline{EA}$ pin is high, the processor fetches instructions from internal program Flash. Bus expansion for accessing program memory from 32K upward automatic since external instruction fetches occur automatically when the program counter exceeds 7FFFh (32K). If the $\overline{EA}$ pin is tied low, all program memory fetches are from external memory.

## Bootloader Architecture

**Introduction**

The bootloader manages a communication according to a specific defined protocol to provide the whole access and service on Flash memory. Furthermore, all accesses and routines can be called from the user application.

**Figure 43.** Diagram Context Description
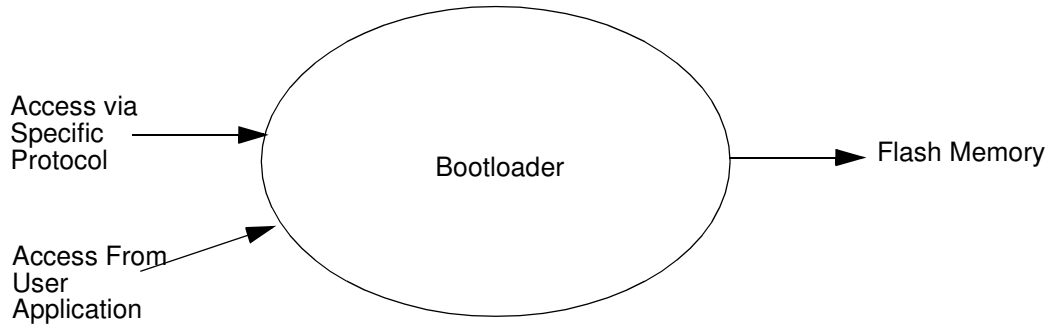


**Acronyms**

ISP: In-system Programming

SBV: Software Boot Vector

BSB: Boot Status Byte

SSB: Software Security Bit

HW : Hardware Byte

**Functional Description**

**Figure 44.** Bootloader Functional Description
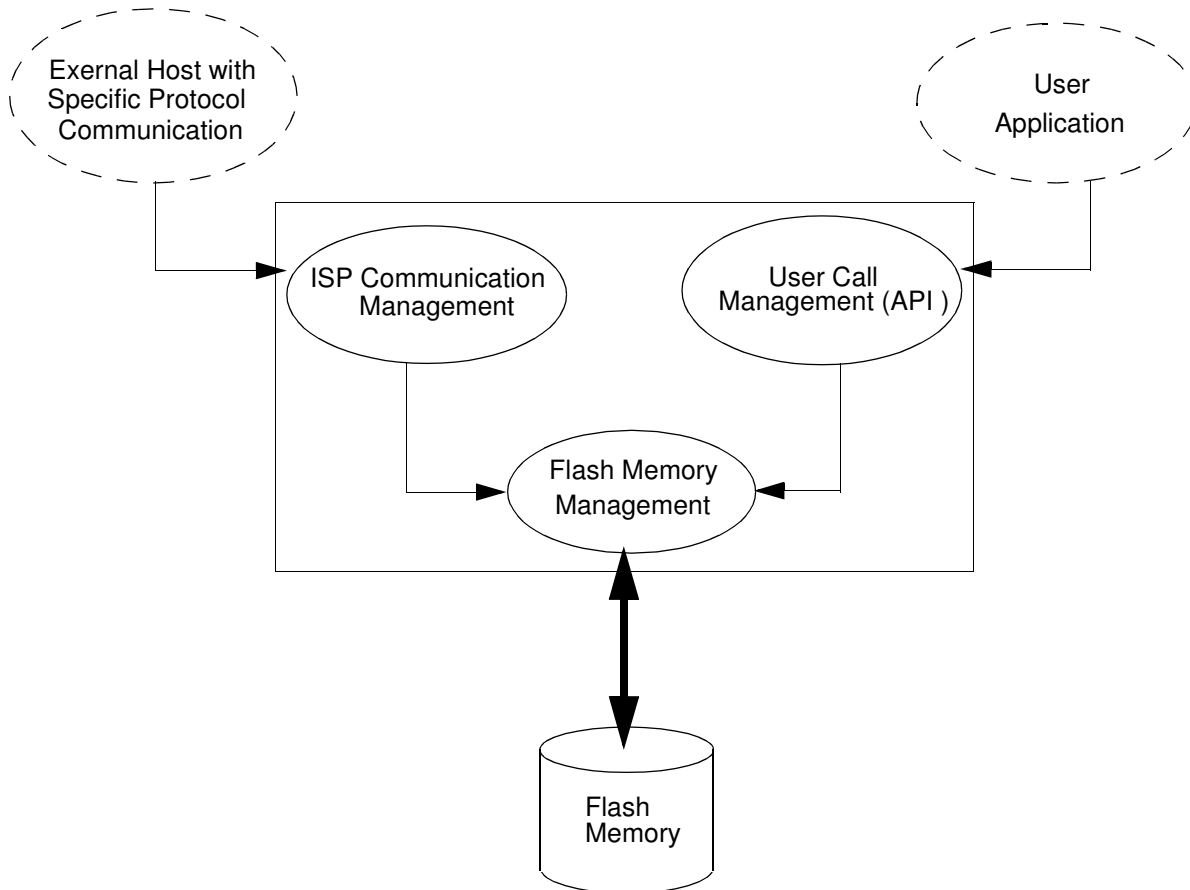


On the above diagram, the on-chip bootloader processes are:

• ISP Communication Management

The purpose of this process is to manage the communication and its protocol between the on-chip bootloader and a external device. The on-chip ROM implement a serial protocol (see section Bootloader Protocol). This process translate serial communication frame (UART) into Flash memory acess (read, write, erase ...).

• User Call Management

Several Application Program Interface (API) calls are available for use by an application program to permit selective erasing and programming of Flash pages. All calls are made through a common interface (API calls), included in the ROM bootloader. The programming functions are selected by setting up the microcontroller's registers before making a call to a common entry point (0xFFF0). Results are returned in the registers. The purpose on this process is to translate the registers values into internal Flash Memory Management.

• Flash Memory Management

This process manages low level access to Flash memory (performs read and write access).

**Bootloader Functionality**

*Introduction*

The bootloader can be activated by two means: Hardware conditions or regular boot process.

The Hardware conditions (EA = 1, PSEN = 0) during the Reset# falling edge force the on-chip bootloader execution. This allows an application to be built that will normally execute the end user's code but can be manually forced into default ISP operation.

As PSEN is an output port in normal operating mode (running user application or boorloader code) after reset, it is recommended to release PSEN after falling edge of reset signal. The hardware conditions are sampled at reset signal falling edge, thus they can be released at any time when reset input is low.

To ensure correct microcontroller startup, the PSEN pin should not be tied to ground during power-on (See Figure 45).

**Figure 45.** Hardware conditions typical sequence during power-on.
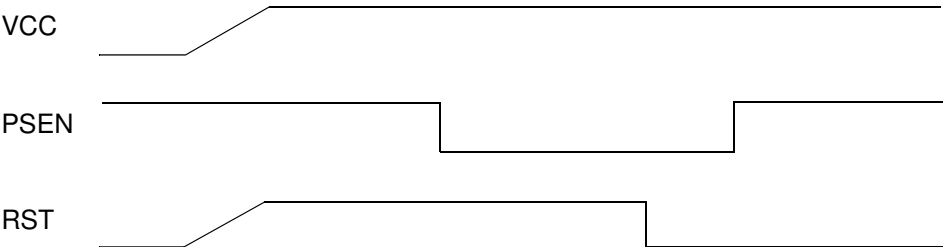


The on-chip bootloader boot process is shown in Figure 46.

|  | **Purpose** |
|---|---|
| Hardware Conditions | The Hardware Conditions force the bootloader execution whatever BLJB, BSB and SBV values. |
| BLJB | The Boot Loader Jump Bit forces the application execution.<br>BLJB = 0 => Boot loader execution.<br>BLJB = 1 => Application execution.<br><br>The BLJB is a fuse bit in the Hardware Byte.<br>That can be modified by hardware (programmer) or by software (API).<br><br>Note:<br>The BLJB test is perform by hardware to prevent any program execution. |
| SBV | The Software Boot Vector contains the high address of custumer bootloader stored in the application.<br>SBV = FCh (default value) if no custumer bootloader in user Flash.<br><br>Note:<br>The costumer bootloader is called by JMP [SBV]00h instruction. |

**Boot Process**

**Figure 46.** Bootloader process



RESET

If BLJB = 0 then ENBOOT bit (AUXR1) is set
    else ENBOOT bit (AUXR1) is cleared

Yes (PSEN = 0, EA = 1, and ALE = 1 or not connected)

**Hardware**

Hardware
Condition?

**FCON = 00h**

**FCON = F0h**

BLJB = 1
ENBOOT = 0

BLJB != 0
?

BLJB = 0
ENBOOT = 1

F800h

**Software**

FCON = 00h
?

yes = hardware boot conditions

BSB = 00h
?

PC = 0000h

USER APPLICATION

SBV = FCh
?

USER BOOT LOADER

PC= [SBV]00h

Atmel BOOT LOADER
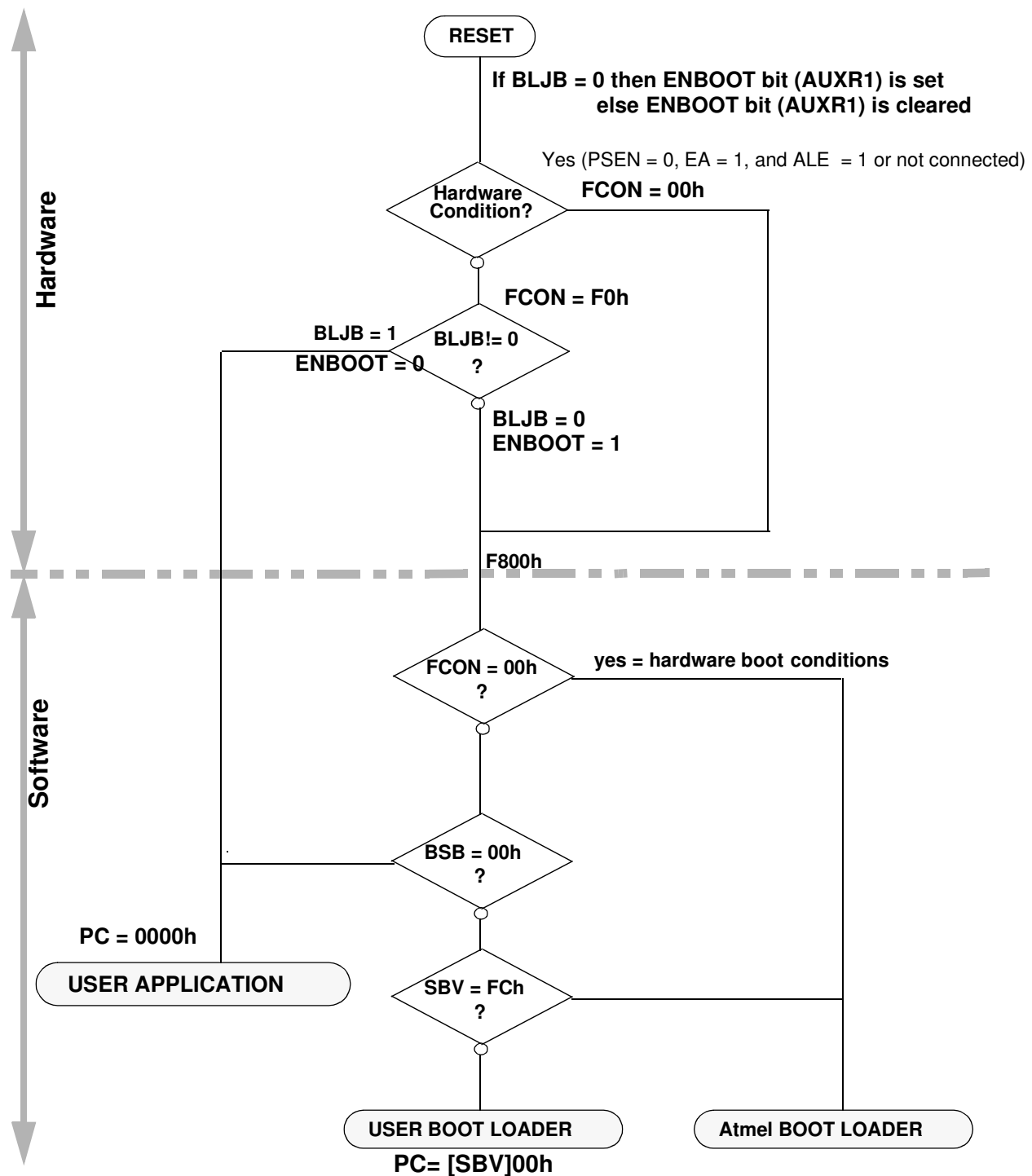
# ISP Protocol Description

**Physical Layer**     The UART used to transmit information has the following configuration:

- Character: 8-bit data
- Parity: none
- Stop: 1 bit
- Flow control: none
- Baud rate: autobaud is performed by the bootloader to compute the baud rate choosen by the host.

**Frame Description**    The Serial Protocol is based on the Intel Hex-type records.

Intel Hex records consist of ASCII characters used to represent hexadecimal values and are summarized below.

**Table 89.** Intel Hex Type Frame

| Record Mark ':' | Reclen | Load Offset | Record Type | Data or Info | Checksum |
|---|---|---|---|---|---|
| 1 byte | 1 byte | 2 bytes | 1 bytes | n byte | 1 byte |

- Record Mark:
    - Record Mark is the start of frame. This field must contain ':'.
- Reclen:
    - Reclen specifies the number of Bytes of information or data which follows the Record Type field of the record.
- Load Offset:
    - Load Offset specifies the 16-bit starting load offset of the data Bytes, therefore this field is used only for
    - Data Program Record (see Section "ISP Commands Summary").
- Record Type:
    - Record Type specifies the command type. This field is used to interpret the remaining information within the frame. The encoding for all the current record types is described in Section "ISP Commands Summary".
- Data/Info:
    - Data/Info is a variable length field. It consists of zero or more Bytes encoded as pairs of hexadecimal digits. The meaning of data depends on the Record Type.
- Checksum:
    - The two's complement of the 8-bit Bytes that result from converting each pair of ASCII hexadecimal digits to one Byte of binary, and including the Reclen field to and including the last Byte of the Data/Info field. Therefore, the sum of all the ASCII pairs in a record after converting to binary, from the Reclen field to and including the Checksum field, is zero.

## Functional Description

**Software Security Bits (SSB)**
The SSB protects any Flash access from ISP command.
The command "Program Software Security bit" can only write a higher priority level.

There are three levels of security:

*   level 0: **NO_SECURITY** (FFh)

This is the default level.
From level 0, one can write level 1 or level 2.

*   level 1: **WRITE_SECURITY** (FEh )

For this level it is impossible to write in the Flash memory, BSB and SBV.
The Bootloader returns 'P' on write access.
From level 1, one can write only level 2.

*   level 2: **RD_WR_SECURITY** (FCh

The level 2 forbids all read and write accesses to/from the Flash/EEPROM memory.
The Bootloader returns 'L' on read or write access.
Only a full chip erase in parallel mode (using a programmer) or ISP command can reset
the software security bits.
From level 2, one cannot read and write anything.

**Table 90.** Software Security Byte Behavior

|  | **Level 0** | **Level 1** | **Level 2** |
|---|---|---|---|
| Flash/EEPROM | Any access allowed | Read only access allowed | Any access not allowed |
| Fuse Bit | Any access allowed | Read only access allowed | Any access not allowed |
| BSB & SBV | Any access allowed | Read only access allowed | Any access not allowed |
| SSB | Any access allowed | Write level 2 allowed | Read only access allowed |
| Manufacturer Info | Read only access allowed | Read only access allowed | Read only access allowed |
| Bootloader Info | Read only access allowed | Read only access allowed | Read only access allowed |
| Erase Block | Allowed | Not allowed | Not allowed |
| Full-chip Erase | Allowed | Allowed | Allowed |
| Blank Check | Allowed | Allowed | Allowed |

**Full Chip Erase**    The ISP command "Full Chip Erase" erases all User Flash memory (fills with FFh) and sets some Bytes used by the bootloader at their default values:

- BSB = FFh
- SBV = FCh
- SSB = FFh and finally erase the Software Security Bits

The Full Chip Erase does not affect the bootloader.

**Checksum Error**    When a checksum error is detected send 'X' followed with CR&LF.

## Flow Description

**Overview**    An initialization step must be performed after each Reset. After microcontroller reset, the bootloader waits for an autobaud sequence ( see section 'autobaud performance').

When the communication is initialized the protocol depends on the record type requested by the host.

FLIP, a software utility to implement ISP programming with a PC, is available from the Atmel the web site.

**Communication Initialization**    The host initializes the communication by sending a 'U' character to help the bootloader to compute the baudrate (autobaud).

**Figure 47.** Initialization
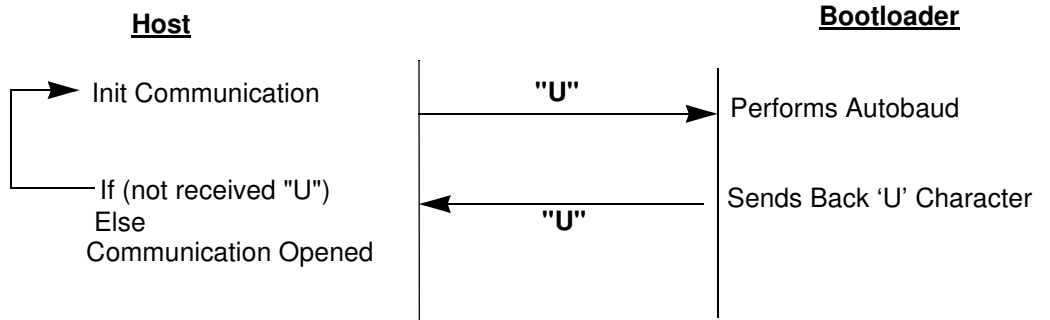
**Autobaud Performances**

The ISP feature allows a wide range of baud rates in the user application. It is also adaptable to a wide range of oscillator frequencies. This is accomplished by measuring the bit-time of a single bit in a received character. This information is then used to program the baud rate in terms of timer counts based on the oscillator frequency. The ISP feature requires that an initial character (an uppercase U) be sent to the AT89C51IC2 to establish the baud rate. Table 91 shows the autobaud capability.

**Table 91.** Autobaud Performances

| Frequency (MHz)<br>Baudrate (bit/s) | 1.8432 | 2 | 2.4576 | 3 | 3.6864 | 4 | 5 | 6 | 7.3728 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|
| 2400 | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 4800 | OK | - | OK | OK | OK | OK | OK | OK | OK | OK |
| 9600 | OK | - | OK | OK | OK | OK | OK | OK | OK | OK |
| 19200 | OK | - | OK | OK | OK | - | - | OK | OK | OK |
| 38400 | - | - | OK | | OK | - | OK | OK | OK | |
| 57600 | - | - | - | - | OK | - | - | - | OK | |
| 115200 | - | - | - | - | - | - | - | - | OK | |

| Frequency (MHz)<br>Baudrate (bit/s) | 10 | 11.0592 | 12 | 14.318 | 14.746 | 16 | 20 | 24 | 26.6 |
|---|---|---|---|---|---|---|---|---|---|
| 2400 | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 4800 | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 9600 | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 19200 | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 38400 | - | OK | OK | OK | OK | OK | OK | OK | OK |
| 57600 | - | OK | - | OK | OK | OK | OK | OK | OK |
| 115200 | - | OK | - | OK | OK | - | - | - | - |

**Command Data Stream Protocol**

All commands are sent using the same flow. Each frame sent by the host is echoed by the bootloader.

**Figure 48.** Command Flow
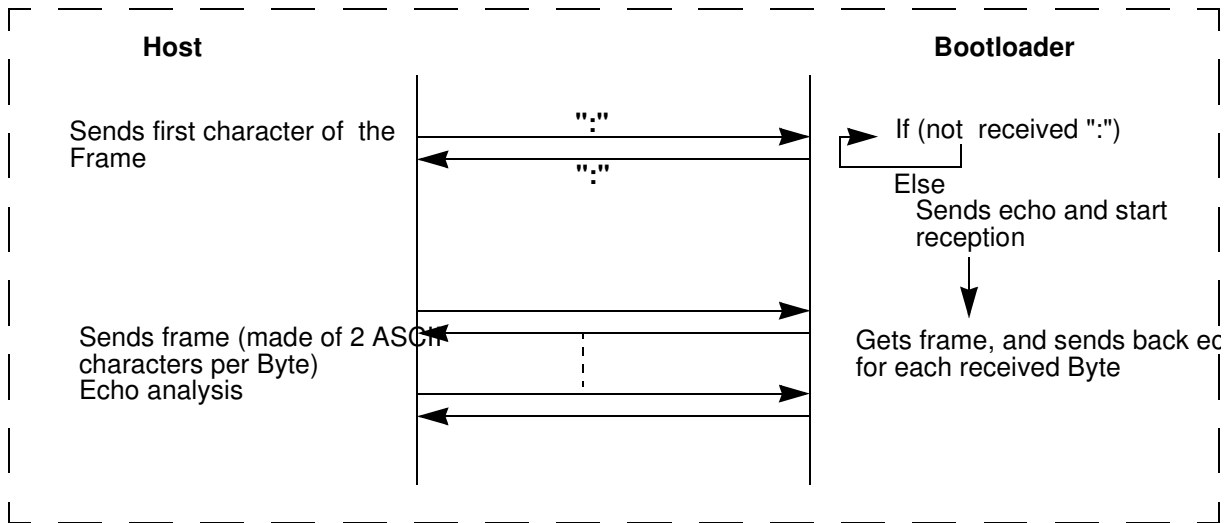


**Write/Program Commands**     This flow is common to the following frames:

- Flash/EEPROM Programming Data Frame
- EOF or Atmel Frame (only Programming Atmel Frame)
- Config Byte Programming Data Frame
- Baud Rate Frame

*Description*

**Figure 49.** Write/Program Flow
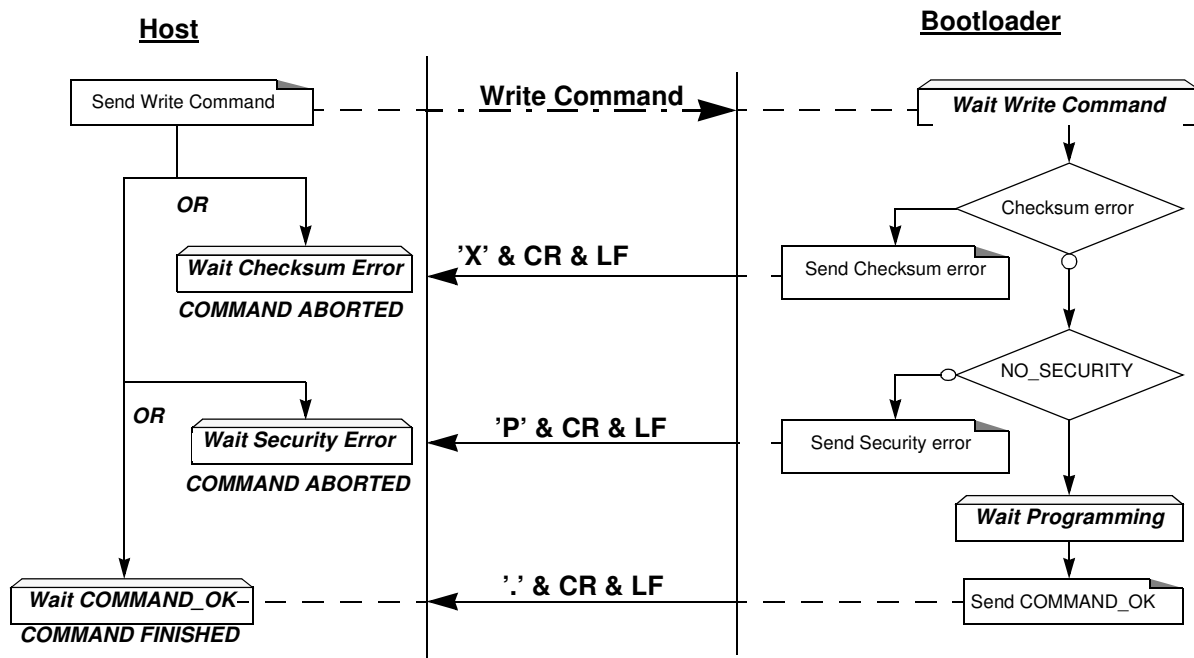
*Example*

Programming Data (write 55h at address 0010h in the Flash)

| | |
|---|---|
| **HOST** | : 01 0010 00 55 9A |
| **BOOTLOADER** | : 01 0010 00 55 9A . CR LF |

Programming Atmel function (write SSB to level 2)

| | |
|---|---|
| **HOST** | : 02 0000 03 05 01 F5 |
| **BOOTLOADER** | : 02 0000 03 05 01 F5. CR LF |

Writing Frame (write BSB to 55h)

| | |
|---|---|
| **HOST** | : 03 0000 03 06 00 55 9F |
| **BOOTLOADER** | : 03 0000 03 06 00 55 9F . CR LF |

**Blank Check Command**

*Description*

**Figure 50.** Blank Check Flow
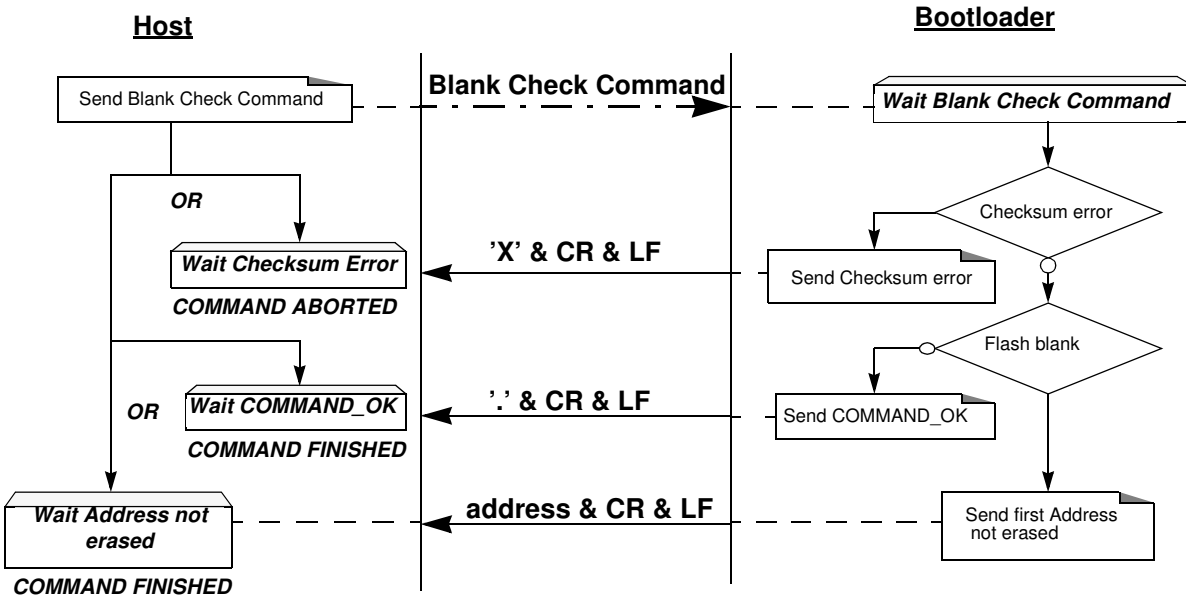


*Example*

```
Blank Check ok
HOST        : 05 0000 04 0000 7FFF 01 78
BOOTLOADER  : 05 0000 04 0000 7FFF 01 78 . CR LF

Blank Check ko at address xxxx
HOST        : 05 0000 04 0000 7FFF 01 78
BOOTLOADER  : 05 0000 04 0000 7FFF 01 78 xxxx CR LF

Blank Check with checksum error
HOST        : 05 0000 04 0000 7FFF 01 70
BOOTLOADER  : 05 0000 04 0000 7FFF 01 70 X CR LF CR LF
```

**Display Data**

*Description*

**Figure 51.** Display Flow
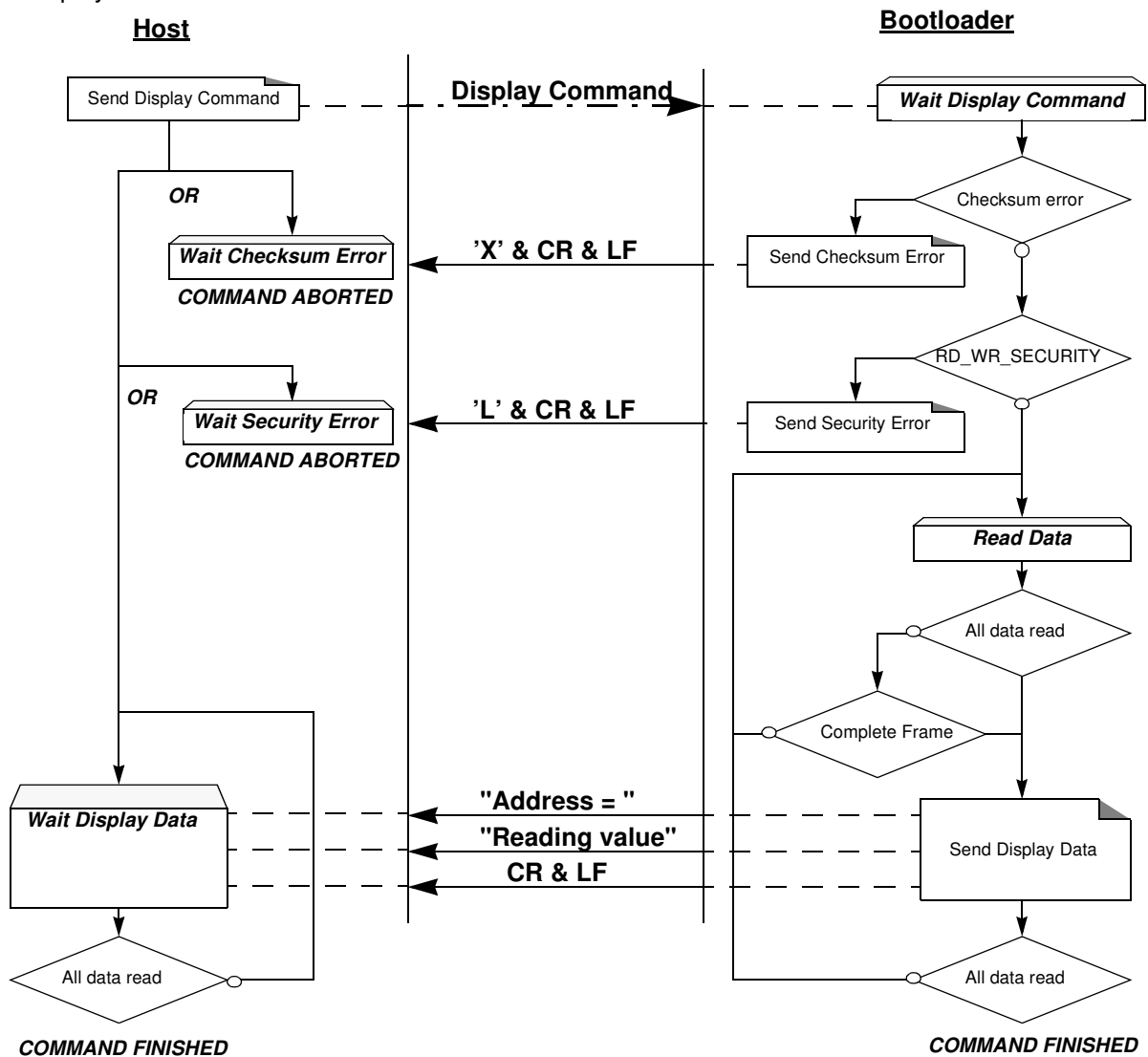


Note: The maximum size of block is 400h. To read more than 400h Bytes, the Host must send a new command.

*Example*

```
Display data from address 0000h to 0020h
```

| | |
|---|---|
| **HOST** | : 05 0000 04 0000 0020 00 D7 |
| **BOOTLOADER** | : 05 0000 04 0000 0020 00 D7 |
| **BOOTLOADER** | 0000=-----data------ CR LF  (16 data) |
| **BOOTLOADER** | 0010=-----data------ CR LF  (16 data) |
| **BOOTLOADER** | 0020=data CR LF          ( 1 data) |

**Read Function**

This flow is similar for the following frames:
- Reading Frame
- EOF Frame/Atmel Frame (only reading Atmel Frame)

*Description*

**Figure 52.** Read Flow
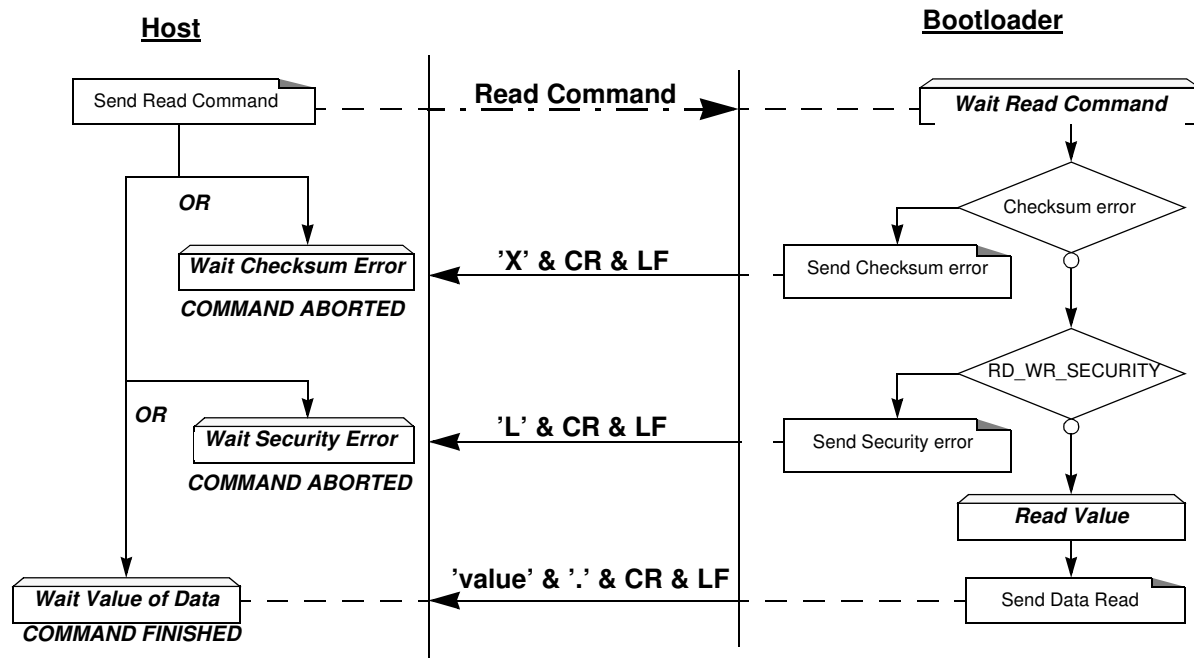


*Example*

```
Read function (read SBV)
```

| | |
|---|---|
| **HOST** | : 02 0000 05 07 02 F0 |
| **BOOTLOADER** | : 02 0000 05 07 02 F0 Value . CR LF |

```
Atmel Read function (read Bootloader version)
```

| | |
|---|---|
| **HOST** | : 02 0000 01 02 00 FB |
| **BOOTLOADER** | : 02 0000 01 02 00 FB Value . CR LF |

**ISP Commands Summary**

**Table 92.** ISP Commands Summary

| Command | Command Name | Data[0] | Data[1] | Command Effect |
|---|---|---|---|---|
| 00h | Program Data | | | Program Nb Data Byte.<br>Bootloader will accept up to 128 (80h) data Bytes. The data Bytes should be 128 Byte page Flash boundary. |
| 03h | Write Function | 01h | 00h | Erase block0 (0000h-1FFFh) |
| | | | 20h | Erase block1 (2000h-3FFFh) |
| | | | 40h | Erase block2 (4000h-7FFFh) |
| | | | 80h | Erase block3 (8000h- BFFFh) |
| | | | C0h | Erase block4 (C000h- FFFFh) |
| | | 03h | 00h | Hardware Reset |
| | | 04h | 00h | Erase SBV & BSB |
| | | 05h | 00h | Program SSB level 1 |
| | | | 01h | Program SSB level 2 |
| | | 06h | 00h | Program BSB (value to write in data[2]) |
| | | | 01h | Program SBV (value to write in data[2]) |
| | | 07h | - | Full Chip Erase (This command needs about 6 sec to be executed) |
| | | 0Ah | 02h | Program Osc fuse (value to write in data[2]) |
| | | | 04h | Program BLJB fuse (value to write in data[2]) |
| | | | 08h | Program X2 fuse (value to write in data[2]) |
| 04h | Display Function | Data[0:1] = start address<br>Data [2:3] = end address<br>Data[4] = 00h -> Display data<br>Data[4] = 01h -> Blank check | | Display Data<br>Note: The maximum number of data that can be read with a single command frame (difference between start and end address) is 1kbyte.<br>Blank Check |
| 05h | Read Function | 00h | 00h | Manufacturer ID |
| | | | 01h | Device ID #1 |
| | | | 02h | Device ID #2 |
| | | | 03h | Device ID #3 |
| | | 07h | 00h | Read SSB |
| | | | 01h | Read BSB |
| | | | 02h | Read SBV |
| | | | 06h | Read Extra Byte |
| | | 0Bh | 00h | Read Hardware Byte |
| | | 0Eh | 00h | Read Device Boot ID1 |
| | | | 01h | Read Device Boot ID2 |
| | | 0Fh | 00h | Read Bootloader Version |

**AIMEL**

**125**

## API Call Description

Several Application Program Interface (API) calls are available for use by an application program to permit selective erasing and programming of Flash pages. All calls are made through a common interface, PGM_MTP. The programming functions are selected by setting up the microcontroller's registers before making a call to PGM_MTP at FFF0h. Results are returned in the registers.

When several Bytes have to be programmed, it is highly recommended to use the Atmel API "PROGRAM DATA PAGE" call. Indeed, this API call writes up to 128 Bytes in a single command.

All routines for software access are provided in the C Flash driver available at Atmel's web site.

The API calls description and arguments are shown in Table 93.

**Table 93.** API Call Summary

| Command | R1 | A | DPTR0 | DPTR1 | Returned Value | Command Effect |
|---|---|---|---|---|---|---|
| READ MANUF ID | 00h | XXh | 0000h | XXh | ACC = Manufacturer Id | Read Manufacturer identifier |
| READ DEVICE ID1 | 00h | XXh | 0001h | XXh | ACC = Device Id 1 | Read Device identifier 1 |
| READ DEVICE ID2 | 00h | XXh | 0002h | XXh | ACC = Device Id 2 | Read Device identifier 2 |
| READ DEVICE ID3 | 00h | XXh | 0003h | XXh | ACC = Device Id 3 | Read Device identifier 3 |
| ERASE BLOCK | 01h | XXh | DPH = 00h | 00h | ACC = DPH | Erase block 0 |
| | | | DPH = 20h | | | Erase block 1 |
| | | | DPH = 40h | | | Erase block 2 |
| | | | Address of byte to program | | | Program one Data Byte in user Flash |
| | | | XXh | | | Erase Software boot vector and boot status byte. (SBV = FCh and BSB = FFh) |
| PROGRAM SSB | 05h | XXh | DPH = 00h DPL = 00h | 00h | ACC = SSB value | Set SSB level 1 |
| | | | DPH = 00h DPL = 01h | | | Set SSB level 2 |
| | | | DPH = 00h DPL = 10h | | | Set SSB level 0 |
| | | | DPH = 00h DPL = 11h | | | Set SSB level 1 |
| PROGRAM BSB | 06h | New BSB value | 0000h | XXh | none | Program boot status byte |
| PROGRAM SBV | 06h | New SBV value | 0001h | XXh | none | Program software boot vector |
| READ SSB | 07h | XXh | 0000h | XXh | ACC = SSB | Read Software Security Byte |
| READ BSB | 07h | XXh | 0001h | XXh | ACC = BSB | Read Boot Status Byte |
| READ SBV | 07h | XXh | 0002h | XXh | ACC = SBV | Read Software Boot Vector |

**Table 93.** API Call Summary  (Continued)

| Command | R1 | A | DPTR0 | DPTR1 | Returned Value | Command Effect |
|---|---|---|---|---|---|---|
| PROGRAM DATA PAGE | 09h | Number of byte to program | Address of the first byte to program in the Flash memory | Address in XRAM of the first data to program | ACC = 0: DONE | Program up to 128 bytes in user Flash. Remark:  number of bytes to program is limited such as the Flash write remains in a single 128 bytes page. Hence, when ACC is 128, valid values of DPL are 00h, or, 80h. |
| PROGRAM X2 FUSE | 0Ah | Fuse value 00h or 01h | 0008h | XXh | none | Program X2 fuse bit with ACC |
| PROGRAM BLJB FUSE | 0Ah | Fuse value 00h or 01h | 0004h | XXh | none | Program BLJB fuse bit with ACC |
| READ HSB | 0Bh | XXh | XXXXh | XXh | ACC = HSB | Read Hardware Byte |
| READ BOOT ID1 | 0Eh | XXh | DPL = 00h | XXh | ACC = ID1 | Read boot ID1 |
| READ BOOT ID2 | 0Eh | XXh | DPL = 01h | XXh | ACC = ID2 | Read boot ID2 |
| READ BOOT VERSION | 0Fh | XXh | XXXXh | XXh | ACC = Boot_Version | Read bootloader version |

# Electrical Characteristics

## Absolute Maximum Ratings

| | |
|---|---|
| C = commercial......................................................0°C to 70°C<br>I = industrial .......................................................-40°C to 85°C<br>Storage Temperature .................................... -65°C to + 150°C<br>Voltage on $V_{CC}$ to $V_{SS}$ ........................................-0.5V to + 6.5V<br>Voltage on Any Pin to $V_{SS}$..........................-0.5V to $V_{CC}$ + 0.5V<br>Power Dissipation ............................................................. 1 W | Note: Stresses at or above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions may affect device reliability.<br>Power dissipation value is based on the maximum allowable die temperature and the thermal resistance of the package. |

## DC Parameters for Standard Voltage

$T_A$ = -40°C to +85°C; $V_{SS}$ = 0V;

$V_{CC}$ =2.7V to 5.5V and F = 0 to 40 MHz (both internal and external code execution)

$V_{CC}$ =4.5V to 5.5V and F = 0 to 60 MHz (internal code execution only)

| Symbol | Parameter | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $V_{IL}$ | Input Low Voltage | -0.5 | | 0.2 $V_{CC}$ - 0.1 | V | |
| $V_{IH}$ | Input High Voltage except RST, XTAL1 | 0.2 $V_{CC}$ + 0.9 | | $V_{CC}$ + 0.5 | V | |
| $V_{IH1}$ [9] | Input High Voltage RST, XTAL1 | 0.7 $V_{CC}$ | | $V_{CC}$ + 0.5 | V | |
| $V_{OL}$ | Output Low Voltage, ports 1, 2, 3, 4 [6] | | | 0.3<br>0.45<br>1.0 | V<br>V<br>V | VCC = 4.5V to 5.5V<br>$I_{OL}$ = 100 μA[4]<br>$I_{OL}$ = 1.6 mA[4]<br>$I_{OL}$ = 3.5 mA[4] |
| | | | | 0.45 | V | VCC = 2.7V to 5.5V<br>$I_{OL}$ = 0.8 mA[4] |
| $V_{OL1}$ | Output Low Voltage, port 0, ALE, $\overline{PSEN}$ [6] | | | 0.3<br>0.45<br>1.0 | V<br>V<br>V | VCC = 4.5V to 5.5V<br>$I_{OL}$ = 200 μA[4]<br>$I_{OL}$ = 3.2 mA[4]<br>$I_{OL}$ = 7.0 mA[4] |
| | | | | 0.45 | V | VCC = 2.7V to 5.5V<br>$I_{OL}$ = 1.6 mA[4] |
| $V_{OH}$ | Output High Voltage, ports 1, 2, 3, 4 | $V_{CC}$ - 0.3<br>$V_{CC}$ - 0.7<br>$V_{CC}$ - 1.5 | | | V<br>V<br>V | $V_{CC}$ = 5V ± 10%<br>$I_{OH}$ = -10 μA<br>$I_{OH}$ = -30 μA<br>$I_{OH}$ = -60 μA |
| | | 0.9 $V_{CC}$ | | | V | VCC = 2.7V to 5.5V<br>$I_{OH}$ = -10 μA |

$T_A$ = -40°C to +85°C; $V_{SS}$ = 0V;

$V_{CC}$ =2.7V to 5.5V and F = 0 to 40 MHz (both internal and external code execution)

$V_{CC}$ =4.5V to 5.5V and F = 0 to 60 MHz (internal code execution only) (Continued)

| Symbol | Parameter | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $V_{OH1}$ | Output High Voltage, port 0, ALE, $\overline{PSEN}$ | $V_{CC}$ - 0.3<br>$V_{CC}$ - 0.7<br>$V_{CC}$ - 1.5 | | | V<br>V<br>V | $V_{CC}$ = 5V ± 10%<br>$I_{OH}$ = -200 µA<br>$I_{OH}$ = -3.2 mA<br>$I_{OH}$ = -7.0 mA |
| | | 0.9 $V_{CC}$ | | | V | VCC = 2.7V to 5.5V<br>$I_{OH}$ = -10 µA |
| $R_{RST}$ | RST Pulldown Resistor | 50 | 200[5] | 250 | kΩ | |
| $I_{IL}$ | Logical 0 Input Current ports 1, 2, 3, 4 and 5 | | | -50 | µA | $V_{IN}$ = 0.45V |
| $I_{LI}$ | Input Leakage Current for P0 only | | | ±10 | µA | 0.45V < $V_{IN}$ < $V_{CC}$ |
| $I_{TL}$ | Logical 1 to 0 Transition Current, ports 1, 2, 3, 4 | | | -650 | µA | $V_{IN}$ = 2.0V |
| $C_{IO}$ | Capacitance of I/O Buffer | | | 10 | pF | Fc = 3 MHz<br>$T_A$ = 25°C |
| $I_{PD}$ | Power Down Current | | 100 | 150 | µA | 4.5V < $V_{CC}$ < 5.5V[3] |
| $I_{CCOP}$ | Power Supply Current on normal mode | | | 0.4 x Frequency (MHz) + 5 | mA | $V_{CC}$ = 5.5V[1] |
| $I_{CCIDLE}$ | Power Supply Current on idle mode | | | 0.3 x Frequency (MHz) + 5 | mA | $V_{CC}$ = 5.5V[1] |
| $I_{CCProg}$ | Power Supply Current during flash Write / Erase | | 0.4 x Frequency (MHz) + 20 | | mA | $V_{CC}$ = 5.5V[8] |

Notes: 1. Operating $I_{CC}$ is measured with all output pins disconnected; XTAL1 driven with $T_{CLCH}$, $T_{CHCL}$ = 5 ns (see Figure 56.), $V_{IL}$ = $V_{SS}$ + 0.5V,
$V_{IH}$ = $V_{CC}$ - 0.5V; XTAL2 N.C.; $\overline{EA}$ = RST = Port 0 = $V_{CC}$. $I_{CC}$ would be slightly higher if a crystal oscillator used (see Figure 53).

2. Idle $I_{CC}$ is measured with all output pins disconnected; XTAL1 driven with $T_{CLCH}$, $T_{CHCL}$ = 5 ns, $V_{IL}$ = $V_{SS}$ + 0.5V, $V_{IH}$ = $V_{CC}$ - 0.5V; XTAL2 N.C; Port 0 = $V_{CC}$; $\overline{EA}$ = RST = $V_{SS}$ (see Figure 54).

3. Power Down $I_{CC}$ is measured with all output pins disconnected; $\overline{EA}$ = $V_{SS}$, PORT 0 = $V_{CC}$; XTAL2 NC.; RST = $V_{SS}$ (see Figure 55).

4. Capacitance loading on Ports 0 and 2 may cause spurious noise pulses to be superimposed on the $V_{OL}$s of ALE and Ports 1 and 3. The noise is due to external bus capacitance discharging into the Port 0 and Port 2 pins when these pins make 1 to 0 transitions during bus operation. In the worst cases (capacitive loading 100pF), the noise pulse on the ALE line may exceed 0.45V with maxi $V_{OL}$ peak 0.6V. A Schmitt Trigger use is not necessary.

5. Typical are based on a limited number of samples and are not guaranteed. The values listed are at room temperature and 5V.

6. Under steady state (non-transient) conditions, $I_{OL}$ must be externally limited as follows:
Maximum $I_{OL}$ per port pin: 10 mA
Maximum $I_{OL}$ per 8-bit port:
Port 0: 26 mA
Ports 1, 2 and 3: 15 mA
Maximum total $I_{OL}$ for all output pins: 71 mA
If $I_{OL}$ exceeds the test condition, $V_{OL}$ may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.

7. For other values, please contact your sales office.

8. Icc Flash Write operation current while an on-chip flash page write is on going.

9. Flash Retention is guaranteed with the same formula for $V_{CC}$ Min down to 0.

**AT MEL**®

## DC Parameters for Low Voltage

$T_A$ = 0°C to +70°C; $V_{SS}$ = 0V; $V_{CC}$ = 2.7V to 3.6V; F = 0to 48 MHz
$T_A$ = -40°C to +85°C; $V_{SS}$ = 0V; $V_{CC}$ = 2.7V to 3.6V; F = 0 to 48 MHz

| Symbol | Parameter | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $V_{IL}$ | Input Low Voltage | -0.5 | | 0.2 $V_{CC}$ - 0.1 | V | |
| $V_{IH}$ | Input High Voltage except RST, XTAL1 | 0.2 $V_{CC}$ + 0.9 | | $V_{CC}$ + 0.5 | V | |
| $V_{IH1}$ | Input High Voltage, RST, XTAL1 | 0.7 $V_{CC}$ | | $V_{CC}$ + 0.5 | V | |
| $V_{OL}$ | Output Low Voltage, ports 1, 2, 3, 4[6] | | | 0.45 | V | $I_{OL}$ = 0.8 mA[4] |
| $V_{OL1}$ | Output Low Voltage, port 0, ALE, $\overline{PSEN}$ [6] | | | 0.45 | V | $I_{OL}$ = 1.6 mA[4] |
| $V_{OH}$ | Output High Voltage, ports 1, 2, 3, 4 | 0.9 $V_{CC}$ | | | V | $I_{OH}$ = -10 µA |
| $V_{OH1}$ | Output High Voltage, port 0, ALE, $\overline{PSEN}$ | 0.9 $V_{CC}$ | | | V | $I_{OH}$ = -40 µA |
| $I_{IL}$ | Logical 0 Input Current ports 1, 2, 3, 4 | | | -50 | µA | $V_{IN}$ = 0.45 V |
| $I_{LI}$ | Input Leakage Current for P0 only | | | ±10 | µA | 0.45V < $V_{IN}$ < $V_{CC}$ |
| $I_{TL}$ | Logical 1 to 0 Transition Current, ports 1, 2, 3, | | | -650 | µA | $V_{IN}$ = 2.0V |
| $R_{RST}$ | RST Pulldown Resistor | 50 | 200 [5] | 250 | kΩ | |
| $C_{IO}$ | Capacitance of I/O Buffer | | | 10 | pF | Fc = 3 MHz $T_A$ = 25°C |
| $I_{PD}$ | Power Down Current | | 10 [5] | 50 | µA | $V_{CC}$ = 2.7V to 3.6V[3] |
| $I_{CCOP}$ | Power Supply Current on normal mode | | | 0.4 x Frequency (MHz) + 5 | mA | $V_{CC}$ = 3.6 V[1] |
| $I_{CCIDLE}$ | Power Supply Current on idle mode | | | 0.3 x Frequency (MHz) + 5 | mA | $V_{CC}$ = 3.6 V[2] |
| $I_{CCProg}$ | Power Supply Current during flash Write / Erase | | 0.4 x Frequency (MHz) + 20 | | mA | $V_{CC}$ = 5.5V[8] |

Notes: 1. Operating $I_{CC}$ is measured with all output pins disconnected; XTAL1 driven with $T_{CLCH}$, $T_{CHCL}$ = 5 ns (see Figure 56.), $V_{IL}$ = $V_{SS}$ + 0.5V,
$V_{IH}$ = $V_{CC}$ - 0.5V; XTAL2 N.C.; $\overline{EA}$ = RST = Port 0 = $V_{CC}$. $I_{CC}$ would be slightly higher if a crystal oscillator used (see Figure 53).

2. Idle $I_{CC}$ is measured with all output pins disconnected; XTAL1 driven with $T_{CLCH}$, $T_{CHCL}$ = 5 ns, $V_{IL}$ = $V_{SS}$ + 0.5V, $V_{IH}$ = $V_{CC}$ - 0.5V; XTAL2 N.C; Port 0 = $V_{CC}$; $\overline{EA}$ = RST = $V_{SS}$ (see Figure 54).

3. Power Down $I_{CC}$ is measured with all output pins disconnected; $\overline{EA}$ = $V_{SS}$, PORT 0 = $V_{CC}$; XTAL2 NC.; RST = $V_{SS}$ (see Figure 55).

4. Capacitance loading on Ports 0 and 2 may cause spurious noise pulses to be superimposed on the $V_{OL}$s of ALE and Ports 1 and 3. The noise is due to external bus capacitance discharging into the Port 0 and Port 2 pins when these pins make 1 to 0 transitions during bus operation. In the worst cases (capacitive loading 100pF), the noise pulse on the ALE line may exceed 0.45V with maxi $V_{OL}$ peak 0.6V. A Schmitt Trigger use is not necessary.

5. Typical are based on a limited number of samples and are not guaranteed. The values listed are at room temperature and 5V.

6. Under steady state (non-transient) conditions, $I_{OL}$ must be externally limited as follows:
Maximum $I_{OL}$ per port pin: 10 mA
Maximum $I_{OL}$ per 8-bit port:
Port 0: 26 mA
Ports 1, 2 and 3: 15 mA
Maximum total $I_{OL}$ for all output pins: 71 mA

If $I_{OL}$ exceeds the test condition, $V_{OL}$ may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.

7.  For other values, please contact your sales office.
8.  Icc Flash Write operation current while an on-chip flash page write is on going.

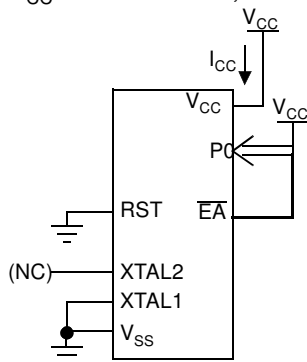**Figure 53.** $I_{CC}$ Test Condition, Active Mode
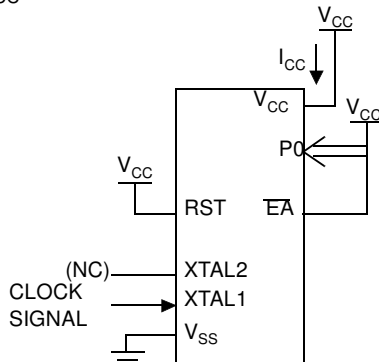
All other pins are disconnected.
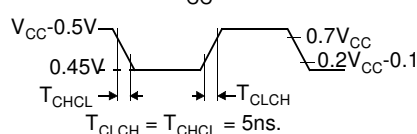
**Figure 54.** $I_{CC}$ Test Condition, Idle Mode
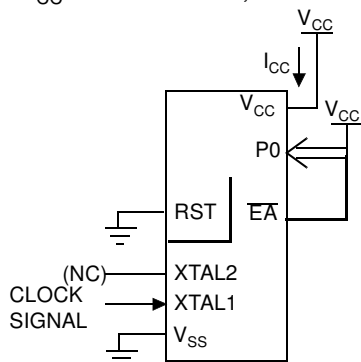
All other pins are disconnected.

**Figure 55.** $I_{CC}$ Test Condition, Power-down Mode

All other pins are disconnected.

**Figure 56.** Clock Signal Waveform for $I_{CC}$ Tests in Active and Idle Modes

$$T_{CLCH} = T_{CHCL} = 5ns.$$

# AC Parameters

**Explanation of the AC Symbols**

Each timing symbol has 5 characters. The first character is always a "T" (stands for time). The other characters, depending on their positions, stand for the name of a signal or the logical status of that signal. The following is a list of all the characters and what they stand for.

Example: $T_{AVLL}$ = Time for Address Valid to ALE Low.
$T_{LLPL}$ = Time for ALE Low to $\overline{PSEN}$ Low.

(Load Capacitance for port 0, ALE and PSEN = 100 pF; Load Capacitance for all other outputs = 80 pF.)

Table 94 Table 97, and Table 99 give the description of each AC symbols.

Table 96, Table 98 and Table 100 give the AC parameterfor each range.

Table 95, Table 96 and Table 101 gives the frequency derating formula of the AC parameter for each speed range description. To calculate each AC symbols, take the x value in the correponding column (-M or -L) and use this value in the formula.

Example: $T_{LLIU}$ for -M and 20 MHz, Standard clock.
x = 35 ns
T 50 ns
$T_{CCIV}$ = 4T - x = 165 ns

**External Program Memory Characteristics**

**Table 94.** Symbol Description

| Symbol | Parameter |
|---|---|
| T | Oscillator clock period |
| $T_{LHLL}$ | ALE pulse width |
| $T_{AVLL}$ | Address Valid to ALE |
| $T_{LLAX}$ | Address Hold after ALE |
| $T_{LLIV}$ | ALE to Valid Instruction In |
| $T_{LLPL}$ | ALE to $\overline{PSEN}$ |
| $T_{PLPH}$ | $\overline{PSEN}$ Pulse Width |
| $T_{PLIV}$ | $\overline{PSEN}$ to Valid Instruction In |
| $T_{PXIX}$ | Input Instruction Hold after $\overline{PSEN}$ |
| $T_{PXIZ}$ | Input Instruction Float after $\overline{PSEN}$ |
| $T_{AVIV}$ | Address to Valid Instruction In |
| $T_{PLAZ}$ | $\overline{PSEN}$ Low to Address Float |

**Table 95.** AC Parameters for a Fix Clock

| Symbol | -M | | | -L | | Units |
|---|---|---|---|---|---|---|
| | Min | | Max | Min | Max | |
| | Min | Max | | Min | Max | |
| T | 25 | | | 25 | | ns |
| $T_{LHLL}$ | 35 | | | 35 | | ns |
| $T_{AVLL}$ | 5 | | | 5 | | ns |
| $T_{LLAX}$ | 5 | | | 5 | | ns |
| $T_{LLIV}$ | | | n 65 | | 65 | ns |
| $T_{LLPL}$ | 5 | | | 5 | | ns |
| $T_{PLPH}$ | 50 | | | 50 | | ns |
| $T_{PLIV}$ | | | 30 | | 30 | ns |
| $T_{PXIX}$ | 0 | | | 0 | | ns |
| $T_{PXIZ}$ | | | 10 | | 10 | ns |
| $T_{AVIV}$ | | | 80 | | 80 | ns |
| $T_{PLAZ}$ | | | 10 | | 10 | ns |

**Table 96.** AC Parameters for a Variable Clock

| Symbol | Type | Standard Clock | X2 Clock | X Parameter for -M Range | X Parameter for -L Range | Units |
|---|---|---|---|---|---|---|
| $T_{LHLL}$ | Min | 2 T - x | T - x | 15 | 15 | ns |
| $T_{AVLL}$ | Min | T - x | 0.5 T - x | 20 | 20 | ns |
| $T_{LLAX}$ | Min | T - x | 0.5 T - x | 20 | 20 | ns |
| $T_{LLIV}$ | Max | 4 T - x | 2 T - x | 35 | 35 | ns |
| $T_{LLPL}$ | Min | T - x | 0.5 T - x | 15 | 15 | ns |
| $T_{PLPH}$ | Min | 3 T - x | 1.5 T - x | 25 | 25 | ns |
| $T_{PLIV}$ | Max | 3 T - x | 1.5 T - x | 45 | 45 | ns |
| $T_{PXIX}$ | Min | x | x | 0 | 0 | ns |
| $T_{PXIZ}$ | Max | T - x | 0.5 T - x | 15 | 15 | ns |
| $T_{AVIV}$ | Max | 5 T - x | 2.5 T - x | 45 | 45 | ns |
| $T_{PLAZ}$ | Max | x | x | 10 | 10 | ns |

**External Program Memory
Read Cycle**
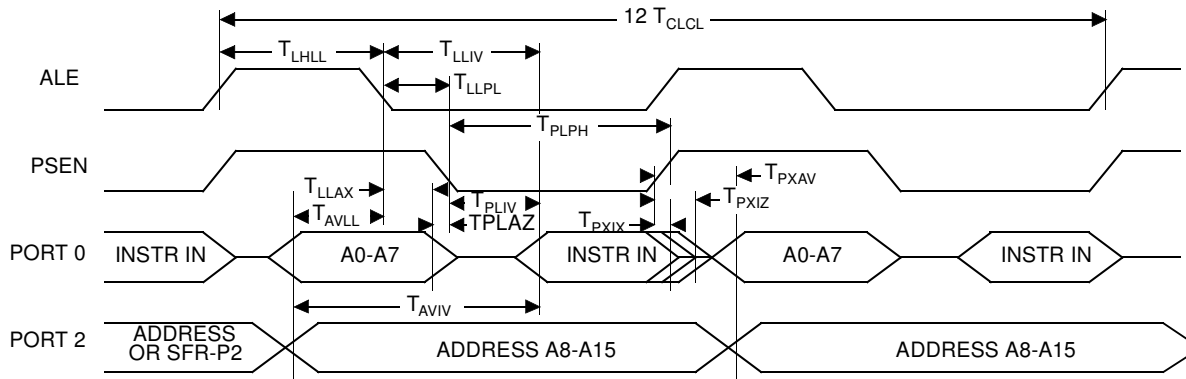


**External Data Memory
Characteristics**

**Table 97.** Symbol Description

| Symbol | Parameter |
|--------|-----------|
| $T_{RLRH}$ | $\overline{RD}$ Pulse Width |
| $T_{WLWH}$ | $\overline{WR}$ Pulse Width |
| $T_{RLDV}$ | $\overline{RD}$ to Valid Data In |
| $T_{RHDX}$ | Data Hold After $\overline{RD}$ |
| $T_{RHDZ}$ | Data Float After $\overline{RD}$ |
| $T_{LLDV}$ | ALE to Valid Data In |
| $T_{AVDV}$ | Address to Valid Data In |
| $T_{LLWL}$ | ALE to $\overline{WR}$ or $\overline{RD}$ |
| $T_{AVWL}$ | Address to $\overline{WR}$ or $\overline{RD}$ |
| $T_{QVWX}$ | Data Valid to $\overline{WR}$ Transition |
| $T_{QVWH}$ | Data set-up to $\overline{WR}$ High |
| $T_{WHQX}$ | Data Hold After $\overline{WR}$ |
| $T_{RLAZ}$ | $\overline{RD}$ Low to Address Float |
| $T_{WHLH}$ | $\overline{RD}$ or $\overline{WR}$ High to ALE high |

**Table 98.** AC Parameters for a Fix Clock

| Symbol | -M | | -L | | Units |
|---|---|---|---|---|---|
| | Min | Max | Min | Max | |
| $T_{RLRH}$ | 125 | | 125 | | ns |
| $T_{WLWH}$ | 125 | | 125 | | ns |
| $T_{RLDV}$ | | 95 | | 95 | ns |
| $T_{RHDX}$ | 0 | | 0 | | ns |
| $T_{RHDZ}$ | | 25 | | 25 | ns |
| $T_{LLDV}$ | | 155 | | 155 | ns |
| $T_{AVDV}$ | | 160 | | 160 | ns |
| $T_{LLWL}$ | 45 | 105 | 45 | 105 | ns |
| $T_{AVWL}$ | 70 | | 70 | | ns |
| $T_{QVWX}$ | 5 | | 5 | | ns |
| $T_{QVWH}$ | 155 | | 155 | | ns |
| $T_{WHQX}$ | 10 | | 10 | | ns |
| $T_{RLAZ}$ | 0 | | 0 | | ns |
| $T_{WHLH}$ | 5 | 45 | 5 | 45 | ns |

| Symbol | Type | Standard Clock | X2 Clock | X Parameter for - M Range | X Parameter for - L Range | Units |
|--------|------|----------------|----------|---------------------------|---------------------------|-------|
| $T_{RLRH}$ | Min | 6 T - x | 3 T - x | 25 | 25 | ns |
| $T_{WLWH}$ | Min | 6 T - x | 3 T - x | 25 | 25 | ns |
| $T_{RLDV}$ | Max | 5 T - x | 2.5 T - x | 30 | 30 | ns |
| $T_{RHDX}$ | Min | x | x | 0 | 0 | ns |
| $T_{RHDZ}$ | Max | 2 T - x | T - x | 25 | 25 | ns |
| $T_{LLDV}$ | Max | 8 T - x | 4T -x | 45 | 45 | ns |
| $T_{AVDV}$ | Max | 9 T - x | 4.5 T - x | 65 | 65 | ns |
| $T_{LLWL}$ | Min | 3 T - x | 1.5 T - x | 30 | 30 | ns |
| $T_{LLWL}$ | Max | 3 T + x | 1.5 T + x | 30 | 30 | ns |
| $T_{AVWL}$ | Min | 4 T - x | 2 T - x | 30 | 30 | ns |
| $T_{QVWX}$ | Min | T - x | 0.5 T - x | 20 | 20 | ns |
| $T_{QVWH}$ | Min | 7 T - x | 3.5 T - x | 20 | 20 | ns |
| $T_{WHQX}$ | Min | T - x | 0.5 T - x | 15 | 15 | ns |
| $T_{RLAZ}$ | Max | x | x | 0 | 0 | ns |
| $T_{WHLH}$ | Min | T - x | 0.5 T - x | 20 | 20 | ns |
| $T_{WHLH}$ | Max | T + x | 0.5 T + x | 20 | 20 | ns |

**External Data Memory Write Cycle**
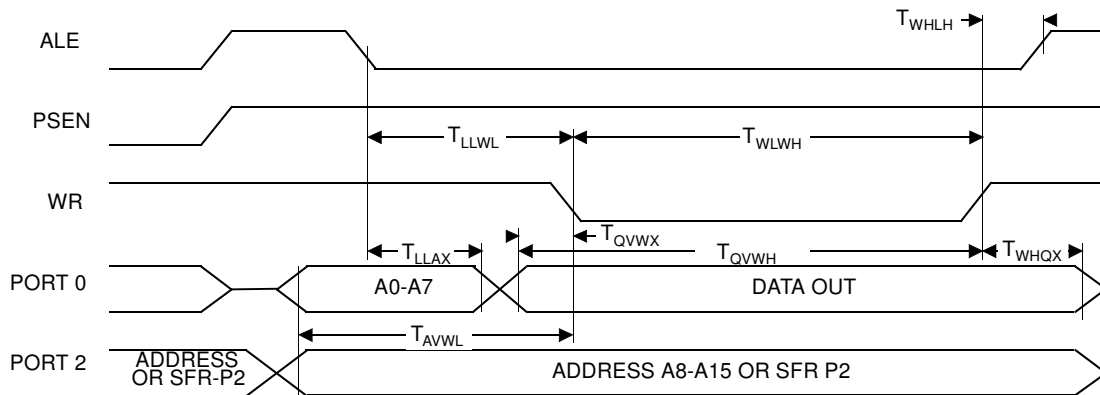
**External Data Memory Read Cycle**
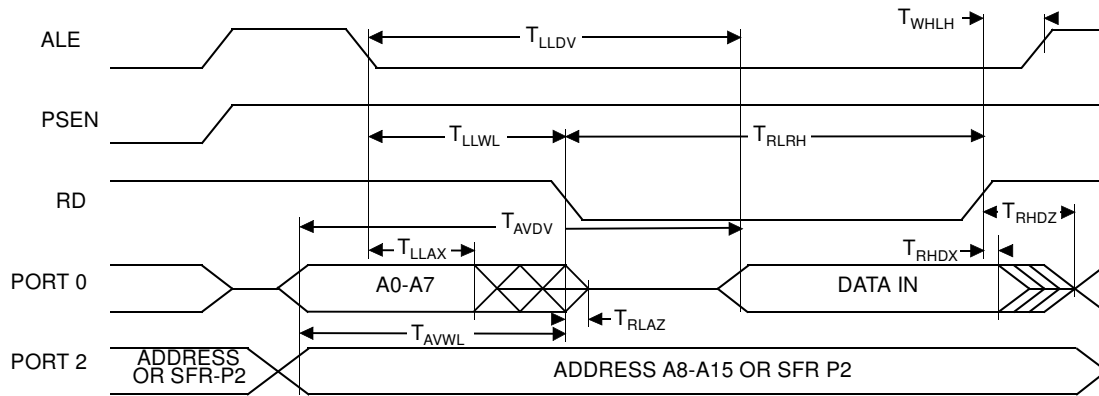


**Serial Port Timing - Shift Register Mode**

**Table 99.** Symbol Description

| Symbol | Parameter |
|--------|-----------|
| $T_{XLXL}$ | Serial port clock cycle time |
| $T_{QVHX}$ | Output data set-up to clock rising edge |
| $T_{XHQX}$ | Output data hold after clock rising edge |
| $T_{XHDX}$ | Input data hold after clock rising edge |
| $T_{XHDV}$ | Clock rising edge to input data valid |

**Table 100.** AC Parameters for a Fix Clock

| Symbol | -M | | -L | | |
|--------|-----|-----|-----|-----|-------|
| | **Min** | **Max** | **Min** | **Max** | **Units** |
| $T_{XLXL}$ | 300 | | 300 | | ns |
| $T_{QVHX}$ | 200 | | 200 | | ns |
| $T_{XHQX}$ | 30 | | 30 | | ns |
| $T_{XHDX}$ | 0 | | 0 | | ns |
| $T_{XHDV}$ | | 117 | | 117 | ns |

**Table 101.** AC Parameters for a Variable Clock

| Symbol | Type | Standard Clock | X2 Clock | X Parameter for -M Range | X Parameter for -L Range | Units |
|--------|------|----------------|----------|--------------------------|--------------------------|-------|
| $T_{XLXL}$ | Min | 12 T | 6 T | | | ns |
| $T_{QVHX}$ | Min | 10 T - x | 5 T - x | 50 | 50 | ns |
| $T_{XHQX}$ | Min | 2 T - x | T - x | 20 | 20 | ns |
| $T_{XHDX}$ | Min | x | x | 0 | 0 | ns |
| $T_{XHDV}$ | Max | 10 T - x | 5 T- x | 133 | 133 | ns |

**Shift Register Timing Waveforms**
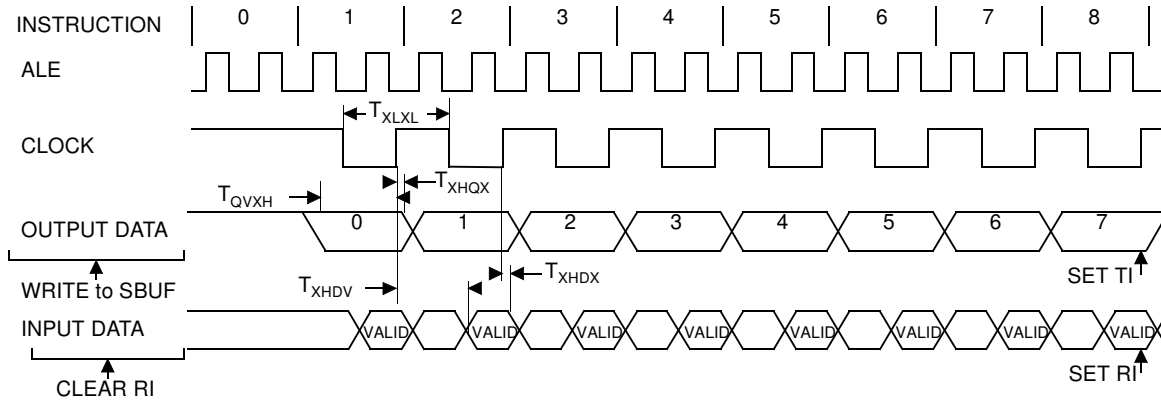


**External Clock Drive Waveforms**
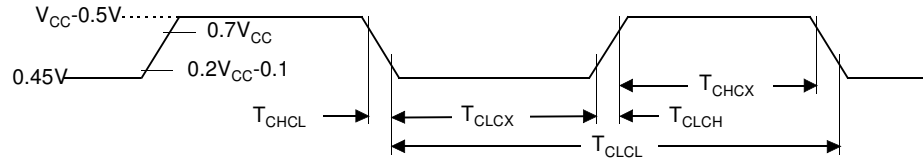


**AC Testing Input/Output Waveforms**
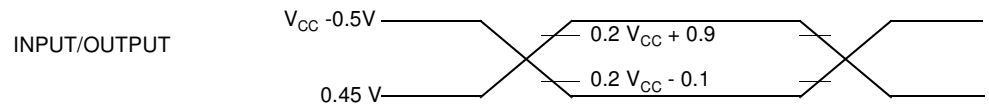


AC inputs during testing are driven at $V_{CC}$ - 0.5 for a logic "1" and 0.45V for a logic "0". Timing measurement are made at $V_{IH}$ min for a logic "1" and $V_{IL}$ max for a logic "0".

**Float Waveforms**
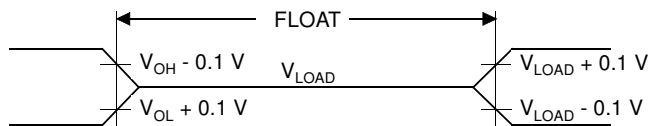


For timing purposes as port pin is no longer floating when a 100 mV change from load voltage occurs and begins to float when a 100 mV change from the loaded $V_{OH}/V_{OL}$ level occurs. $I_{OL}/I_{OH} \geq \pm 20$mA.

**Clock Waveforms**        Valid in normal clock mode. In X2 mode XTAL2 must be changed to XTAL2/2.

**Figure 57.** Internal Clock Signals
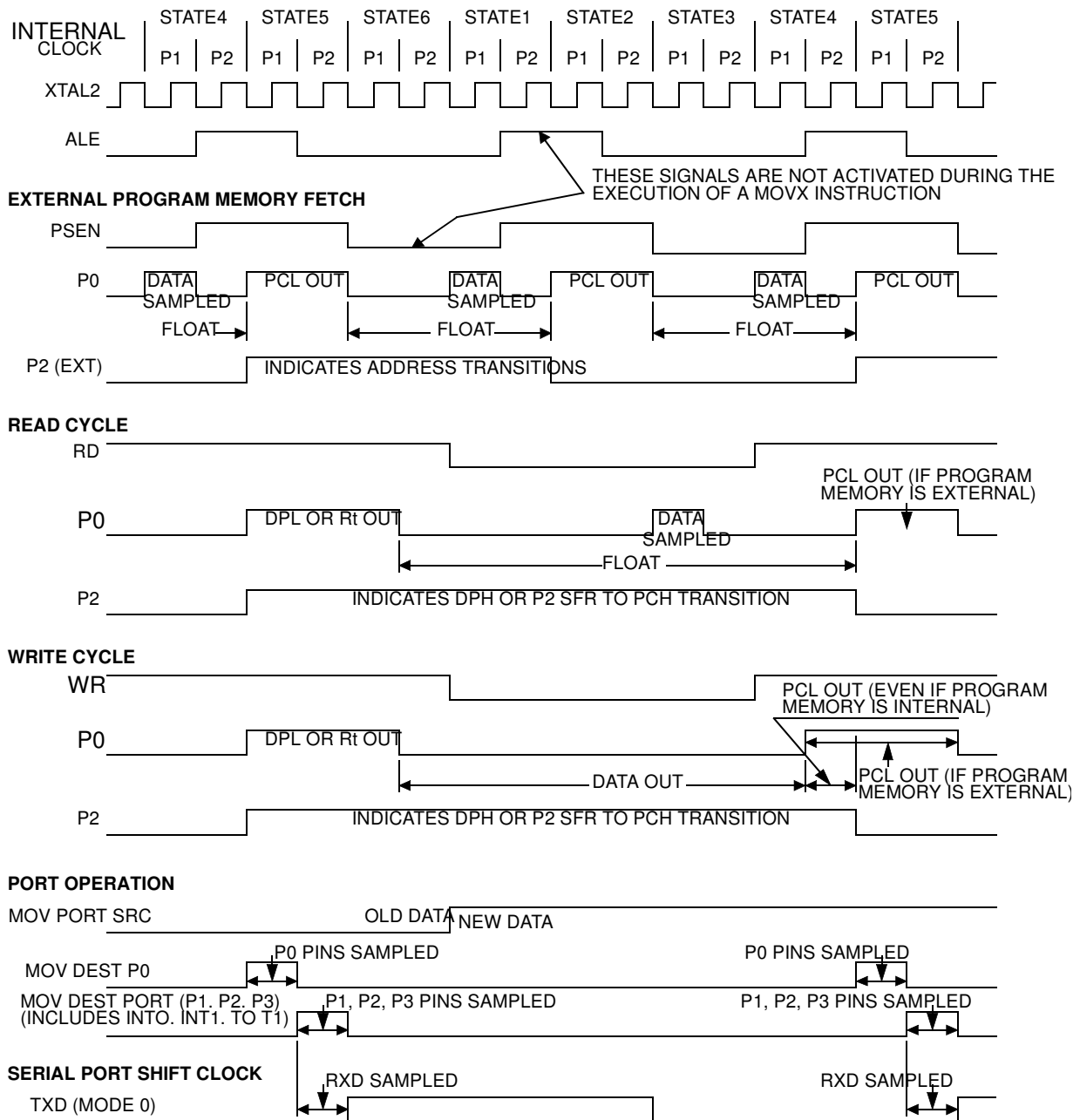


This diagram indicates when signals are clocked internally. The time it takes the signals to propagate to the pins, however, ranges from 25 to 125 ns. This propagation delay is dependent on variables such as temperature and pin loading. Propagation also varies from output to output and component. Typically though ($T_A$ = 25°C fully loaded) RD and WR propagation delays are approximately 50 ns. The other signals are typically 85 ns. Propagation delays are incorporated in the AC specifications.

# Ordering Information

**Table 102.** Possible Order Entries

| Part Number | Flash Memory Size | Supply Voltage | Temperature Range | Package | Packing | Product Marking |
|---|---|---|---|---|---|---|
| AT89C51IC2-SLSCM | | | | | | |
| AT89C51IC2-SLSIM | | | | | | |
| AT89C51IC2-RLTIM | | | OBSOLETE | | | |
| AT89C51IC2-RLTIL | | | | | | |
| AT89C51IC2-SLSIL | | | | | | |
| | | | | | | |
| AT89C51IC2-SLSUM | 32K bytes | 5V | Industrial & Green | PLCC44 | Stick | 89C51IC2-UM |
| AT89C51IC2-RLTUM | 32K bytes | | Industrial & Green | VQFP44 | Tray | 89C51IC2-UM |
| AT89C51IC2-RLTUL | 32K bytes | 3V | Industrial & Green | VQFP44 | Tray | 89C51IC2-UL |
| AT89C51IC2-SLSUL | 32K bytes | | Industrial & Green | PLCC44 | Stick | 89C51IC2-UL |

# Package Drawing

## PLCC44
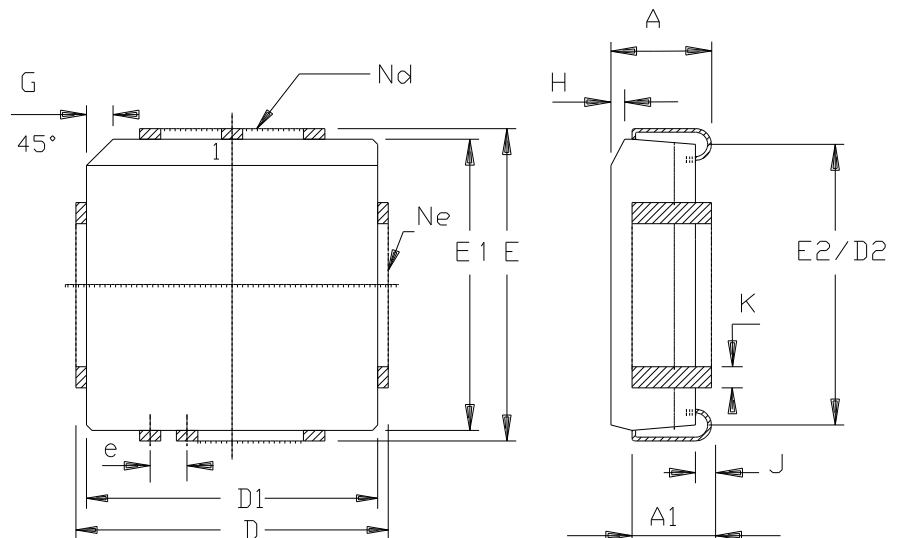


| | MM | | INCH | |
|---|---|---|---|---|
| A | 4.20 | 4.57 | .165 | .180 |
| A1 | 2.29 | 3.04 | .090 | .120 |
| D | 17.40 | 17.65 | .685 | .695 |
| D1 | 16.44 | 16.66 | .647 | .656 |
| D2 | 14.99 | 16.00 | .590 | .630 |
| E | 17.40 | 17.65 | .685 | .695 |
| E1 | 16.44 | 16.66 | .647 | .656 |
| E2 | 14.99 | 16.00 | .590 | .630 |
| e | 1.27 | BSC | .050 | BSC |
| G | 1.07 | 1.22 | .042 | .048 |
| H | 1.07 | 1.42 | .042 | .056 |
| J | 0.51 | – | .020 | – |
| K | 0.33 | 0.53 | .013 | .021 |
| Nd | 11 | | 11 | |
| Ne | 11 | | 11 | |
| PKG STD | | 00 | | |

## Package Drawing

**VQFP44**
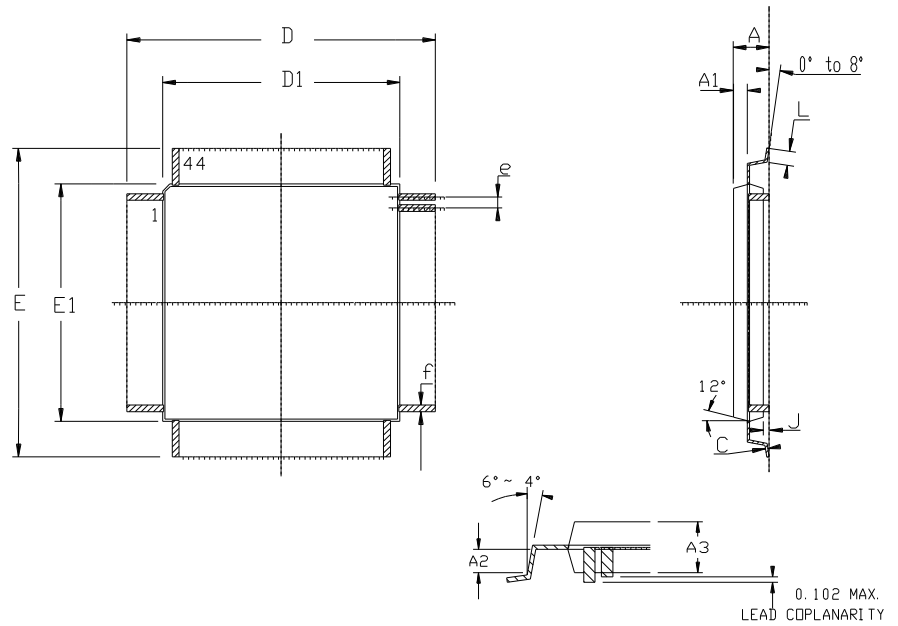


| | MM | | INCH | |
|---|---|---|---|---|
| | Min | Max | Min | Max |
| A | – | 1.60 | – | .063 |
| A1 | 0.64 REF | | .025 REF | |
| A2 | 0.64 REF | | .025 REF | |
| A3 | 1.35 | 1.45 | .053 | .057 |
| D | 11.90 | 12.10 | .468 | .476 |
| D1 | 9.90 | 10.10 | .390 | .398 |
| E | 11.90 | 12.10 | .468 | .476 |
| E1 | 9.90 | 10.10 | .390 | .398 |
| J | 0.05 | – | .002 | – |
| L | 0.45 | 0.75 | .018 | .030 |
| e | 0.80 BSC | | .0315 BSC | |
| f | 0.35 BSC | | .014 BSC | |

## Datasheet Revision History

**Changes from Rev. A
01/04 - Rev. B 01/06**

1. Added green product ordering information.

**Changes from Rev. B
01/06 - Rev. C 06/06**

1. Correction to ordering information concerning product marking on green products.

**Changes from Rev. C
06/06 - Rev. D 02/08**

1. Removed non green part numbers from ordering information.

# Table of Contents

## Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

## Regional Headquarters

*Europe*
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

*Asia*
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

*Japan*
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

## Atmel Operations

*Memory*
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

*Microcontrollers*
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

*ASIC/ASSP/Smart Cards*
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

*RF/Automotive*
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

*Biometrics/Imaging/Hi-Rel MPU/*
*High Speed Converters/RF Datacom*
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

*Literature Requests*
www.atmel.com/literature

 Printed on recycled paper.

# Mouser Electronics

Authorized Distributor


Click to View Pricing, Inventory, Delivery & Lifecycle Information:


[Microchip](#):
  [AT89C51IC2-RLRUL](#)  [AT89C51IC2-RLRUM](#)  [AT89C51IC2-SLRUL](#)  [AT89C51IC2-SLRUM](#)  [AT89C51IC2-SLSUM](#)
[AT89C51IC2-RLTUM](#)  [AT89C51IC2-RLTUL](#)  [AT89C51IC2-SLSUL](#)

**LifeElectronics**
Живое партнерство

**ООО "ЛайфЭлектроникс"**                                                                                          **"LifeElectronics" LLC**

*ИНН 7805602321 КПП 780501001 Р/С 40702810122510004610 ФАКБ "АБСОЛЮТ БАНК" (ЗАО) в г.Санкт-Петербурге К/С 30101810900000000703 БИК 044030703*

Компания «Life Electronics» занимается поставками электронных компонентов импортного и отечественного производства от производителей и со складов крупных дистрибьюторов Европы, Америки и Азии.

С конца 2013 года компания активно расширяет линейку поставок компонентов по направлению коаксиальный кабель, кварцевые генераторы и конденсаторы (керамические, пленочные, электролитические),  за  счёт заключения дистрибьюторских договоров

Мы предлагаем:

- Конкурентоспособные цены и скидки постоянным клиентам.
- Специальные условия для постоянных клиентов.
- Подбор аналогов.
- Поставку компонентов в любых объемах, удовлетворяющих вашим потребностям.

- Приемлемые сроки поставки, возможна ускоренная поставка.
- Доставку товара в любую точку России и стран СНГ.
- Комплексную поставку.
- Работу по проектам и поставку образцов.
- Формирование склада под заказчика.

- Сертификаты соответствия на поставляемую продукцию (по желанию клиента).
- Тестирование поставляемой продукции.
- Поставку компонентов, требующих военную и космическую приемку.
- Входной контроль качества.
- Наличие сертификата ISO.

В составе нашей компании организован Конструкторский отдел, призванный помогать разработчикам, и инженерам.

Конструкторский отдел помогает осуществить:

- Регистрацию проекта у производителя компонентов.
- Техническую поддержку проекта.
- Защиту от снятия компонента с производства.
- Оценку стоимости проекта по компонентам.
- Изготовление тестовой платы монтаж и пусконаладочные работы.



*Тел:  +7 (812) 336 43 04 (многоканальный)*
*Email:  org@lifeelectronics.ru*

**www.lifeelectronics.ru**