**8051 RISC Microcontroller**

**High-Speed, 8051-Compatible,
With SRAM and Extended Functions**

## 1. Description

The TSCR8051 8-bit microcontroller is software compatible with the millions of devices that have been produced since Intel® introduced the 8051 line in 1980. It executes all ASM51 instructions and uses the same instruction set as the 8031. Its Reduced Instruction Set Computer (RISC) core executes many of its instructions in a single clock cycle, providing a significant speed advantage over traditional 8051 devices that execute an instruction every twelve clock cycles. With clock speeds of up to 200 MHz, this device is an 8051 performance leader.

The TSCR8051 features 128KBKBytes of partitionable Data and Program memory and extended 32-bit capabilities including an IEEE 754-compliant floating-point coprocessor with comparator, a multiply/divide unit, a population counter, and a leading-zero counter.

## 2. Features

- Industry standard 8051 / 8031 software compatible
- RISC architecture with up to x12 speed advantage / MHz over traditional 8051 family devices
- Four speed grades: 100, 150, 180, and 200 MHz
- 128KB of additional high-speed SRAM memory
- IEEE 754-compliant floating point coprocessor for full arithmetic capabilities – up to 100 MFlops
- Extended 32-bit computing functions including population counter, leading zero counter, and floating-point comparator
- Dual data pointers for fast data block moves
- Full 8051-compatible architecture including:
  - Four 8-bit bi-directional ports
  - 256 Bytes of “Scratch Pad” memory
  - Three 16-bit timer/counters
  - Interrupt controller with 12 interrupt sources and 4 priority levels
  - 15-bit programmable watchdog timer
  - Core 8-bit arithmetic logic unit and 16-bit multiplication division unit
  - Two full-duplex serial ports
  - Four capture/compare units to generate pulse width modulated signals
  - Special Function Register (SFR) interface, serving up to 50 SFR devices

## 3. Part Numbering

| Options | | Marking |
|---|---|---|
| ▪ Packages: | | |
| | 44 PLCC | xxxxx |
| | 68 PLCC | xxxxx |
| | 100 TQFP | xxxxx |
| ▪ Operating Temperature: | | |
| | Standard, 0º to 70º C | S |
| | Extended (planned) | E |
| ▪ Speed Grade: | | |
| | 66 MHz | -06 |
| | 100 MHz | -10 |
| | 150 MHz | -15 |
| | 180 MHz | -18 |
| | 200 MHz | -20 |

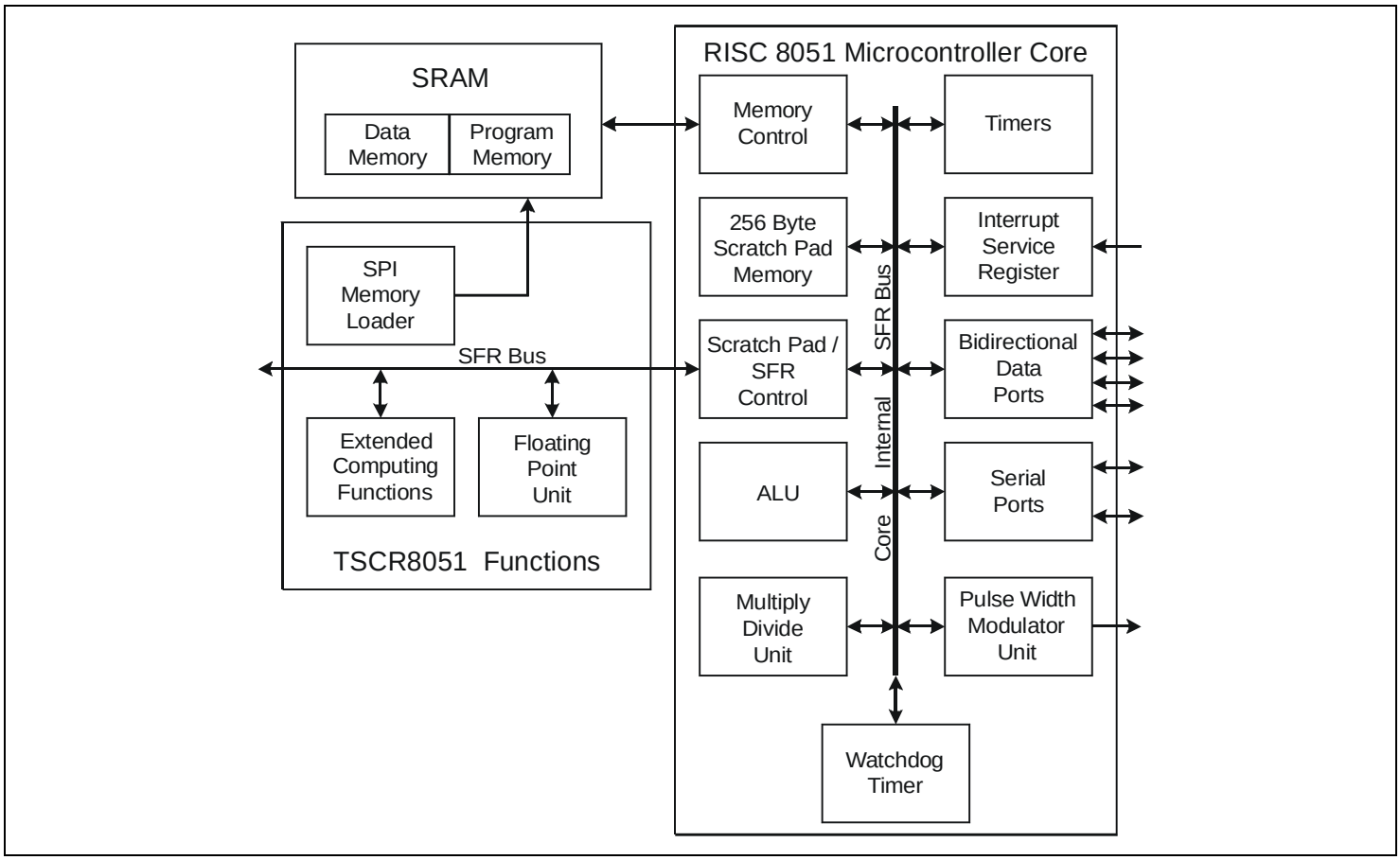
Part number example: **TSCxxxxxPS-06**

## 4. Operating Voltages

$V_{DDQ}$, $V_{DDQF}$ = 3.3 ± .3 VDC
$V_{DD}$ = 1.8 ± .2 VDC

**Figure 1: Block Diagram**




## 5. Table of Contents



## 6. Table of Figures



## Special Function Register Descriptions (Continued)



## 7. Table of Tables



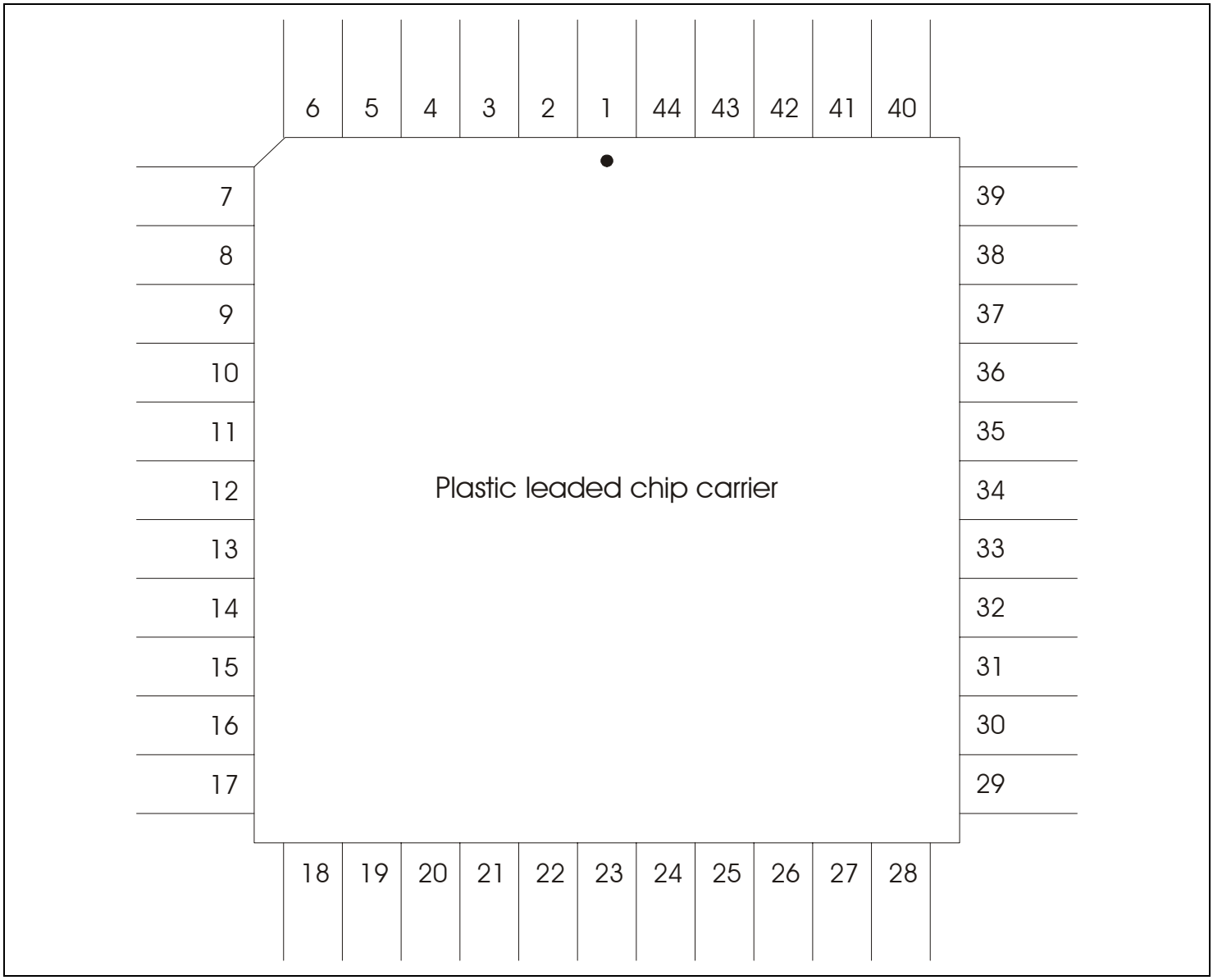
## Special Function Register Descriptions (Continued)

## 8. Block Diagram

## 9. Pin-Out

### 9.1. 44 PLCC

**Figure 2: 44 PPLC, Top View**



## Special Function Register Descriptions (Continued)

**Table 1: 44 PLCC Pinout**

| No. | Name | No. | Name | No. | Name | No. | Name |
|---|---|---|---|---|---|---|---|
| 1 | Vdd Core | 12 | SD_IN | 23 | SD/CLK | 34 | Vss |
| 2 | P1.0 | 13 | P3.1/TxD | 24 | P2.0/A8 | 35 | #EA |
| 3 | P1.1 | 14 | P3.2/#INT0 | 25 | P2.1/A9 | 36 | P0.7/AD7 |
| 4 | P1.2 | 15 | P3.3/#INT1 | 26 | P2.2/A10 | 37 | P0.6/AD6 |
| 5 | P1.3 | 16 | P3.4/T0 | 27 | P2.3/A11 | 38 | P0.5/AD5 |
| 6 | P1.4 | 17 | P3.5/T1 | 28 | P2.4/A12 | 39 | P0.4.AD4 |
| 7 | P1.5 | 18 | P3.6/#WR | 29 | P2.5/A13 | 40 | P0.3/AD3 |
| 8 | P1.6/SCL | 19 | P3.7/#RD | 30 | P2.6/A14 | 41 | P0.2/AD2 |
| 9 | P1.7/SDA | 20 | CLK1 | 31 | P2.7/A15 | 42 | P0.1/AD1 |
| 10 | RST | 21 | CLK0 | 32 | #PSEN | 43 | P0.0/AD0 |
| 11 | P3.0/RxD | 22 | Vss | 33 | ALE | 44 | Vdd |

### 9.2. 68 PLCC

*Not available at time of revision.*

### 9.3. 100 TQFP

*Not available at time of revision.*

## Special Function Register Descriptions (Continued)

### 9.4. Pin Descriptions

| PORT PINS | | | | | |
|---|---|---|---|---|---|
| These are 8-bit bi-directional I/O ports. Each consists of a latch, output driver, and input buffer. | | | | | |
| **Symbol** | **Type** | **Description** | | | |
| PORT0 [7:0] | I/O | An open-drain port. Pins set to 1 will float and can be used as high impedance inputs. PORT0 pins must be polarized to $V_{DDQ}$ or $V_{SSQ}$ in order to prevent any parasitic current consumption. | | | |
| PORT1 [7:0] | I/O | A port with internal pull-ups. Pins set to 1 are pulled high and can be used as inputs. Input pins that are externally pulled low will source current because of the internal pull-ups.<br>In addition to general purpose I/O, PORT1 has alternate functions that may be accessed through the Special Function Registers. Data can be read or written through any pin that is not being used for alternate functions. | | | |
| | | **Pin** | **Name** | **Type** | **Description** |
| | | PORT1[0] | INT3 | Input | External Interrupt 3 |
| | | | CC0 | Input | Capture/Compare 0 |
| | | | RXD1 | Input | Serial Port 1 Receive Pin (see page 51) |
| | | PORT1[1] | INT4 | Input | External Interrupt 4 |
| | | | CC1 | Input | Capture/Compare 1 |
| | | | TXD1 | Output | Serial Port 1 Transmit Pin (see page 51) |
| | | PORT1[2] | INT5 | Input | External Interrupt 5 |
| | | | CC2 | Input | Capture/Compare 2 |
| | | PORT1[3] | INT6 | Input | External Interrupt 6 |
| | | | CC3 | Input | Capture/Compare 3 |
| | | PORT1[4] | INT2 | Input | External Interrupt 2 |
| | | PORT1[5] | T2EX | Input | Timer 2 External Reload Trigger |
| | | PORT1[6] | No Alternate Function | | |
| | | PORT1[7] | T2 | Input | Timer 2 Counter Trigger or Timer Gate |
| PORT2 [7:0] | I/O | A port with internal pull-ups. Pins set to 1 are pulled high and can be used as inputs. Input pins that are externally pulled low will source current because of the internal pull-ups. | | | |
| PORT3 [7:0] | I/O | A port with internal pull-ups. Pins set to 1 are pulled high and can be used as inputs. Input pins that are externally pulled low will source current because of the internal pull-ups.<br>In addition to general purpose I/O, PORT3 has alternate functions that may be accessed through the Special Function Registers. Data can be read or written through any pin that is not being used for alternate functions. | | | |
| | | **Pin** | **Name** | **Type** | **Description** |
| | | PORT3[0] | RXD0 | Input | Serial Port 0 Receive Pin (see page 51) |
| | | PORT3[1] | TXD0 | Output | Serial Port 0 Transmit Pin (see page 51) |
| | | PORT3[2] | INT0 | Input | External Interrupt 0 |
| | | PORT3[3] | INT1 | Input | External Interrupt 1 |
| | | PORT3[4] | T0 | Input | Timer 0 Counter Trigger |
| | | PORT3[5] | T1 | Input | Timer 1 Counter Trigger |
| | | PORT3[6] | No Alternate Function | | |
| | | PORT3[7] | No Alternate Function | | |

## Special Function Register Descriptions (Continued)

## Pin Descriptions (continued)

| Clock and Reset Pins | | |
|---|---|---|
| **Symbol** | **Type** | **Description** |
| CLK_0, CLK_1 | Input | **Clock Differential Inputs**: These signals (combined) form the internal system clock. They are designed to work with a differential clock oscillator and will not work with a crystal. These signals are ignored if CLK_OVR is asserted. |
| LCLK | Input | **Program Memory Loading Clock**: Drives program memory loading circuitry. After program memory is loaded, this signal is ignored unless CLK_OVR is asserted. This pin is designed to work with a clock oscillator and will not work with a crystal. |
| CLK_OUT | Output | **System Clock Output**: Provides access to the internal clock for external SFR circuitry. |
| RESET | Input | **Global Reset**: A high level for 2 clock cycles (if oscillator is running) resets the hardware. |
| CLK_ OVR | Input | **Clock Override**: When asserted, LCLK is used in place of CLK as the system clock. |

| Serial Peripheral Interface (SPI) Pins | | |
|---|---|---|
| **Symbol** | **Type** | **Description** |
| SD_IN | Input | **Serial Data Input for** SPI: program memory input. Immediately upon power-up or after reset, program memory is loaded from an SPI-compatible device. |
| SD_OUT | Output | **Serial Data Output for SPI** |
| SD_CLK | Output | **Serial Data Clock**: During program memory loading, SD_CLK is equivalent to either LCLK or the system clock, depending on the status of the LCLK_ EN input (see Clock Pins above). |
| SD_CS_ | Output | **Serial Data Chip Select (active low) for SPI** |
| SD_BSY | Output | **Serial Data Busy**: Asserted during SPI program memory loading. |

| Power and Ground Pins | | |
|---|---|---|
| **Symbol** | **Type** | **Description** |
| $V_{DD}$ | Power | **1.8 V Power Supply** |
| $V_{DDQ}$ | Power | **3.3 V Power Supply** |
| $V_{DDQF}$ | Power | **3.3 V Filtered Power Supply** for differential clock buffer |
| $V_{SS}$, $V_{SSQ}$, $V_{SSQF}$ | Ground | **Ground** for $V_{DD}$, $V_{DDQ}$, $V_{DDQF}$ |

| Miscellaneous Pins | | |
|---|---|---|
| **Symbol** | **Type** | **Description** |
| PMODE | Input | **Memory Page Mode** affects the meaning of registers PPG, DRPG, and DWPG, and the mapping of logical memory addresses. See section 11.2 on page 32 for details. |
| SWD | Input | **Start Watchdog Timer**: If held high during reset, the watchdog timer starts immediately after the reset. |

| Debug Pins | | |
|---|---|---|
| **Symbol** | **Type** | **Description** |
| DC2, DC1, DC0 | Input | **Debug Control**: These signals select which internal signals are available on the DBG bus. They control the debug multiplexer, which chooses among various signals that are accessed during factory testing. During normal operation DC[2:0] should be tied to ground. |
| DBG [7:0] | Output | **Debug Port**: This bus is driven by the debug multiplexer, which chooses among various signals based on the state of the debug control (DC[2:0]) signals. During normal operation these pins will be driven to ground and should not be connected to external circuitry. |

## Special Function Register Descriptions (Continued)

## Pin Descriptions (continued)

| SPECIAL FUNCTION REGISTER (SFR) PINS | | |
|---|---|---|
| **Symbol** | **Type** | **Description** |
| SFR_I [7:0] | Input | **SFR Input Bus**: Allows external SFR circuitry to transmit data to the 8051 core. If external SFR circuitry is not used, these signals should be tied to ground. |
| SFR_O [7:0] | Output | **SFR Output Bus**: Allows the 8051 core to send data to external SFR circuitry. |
| SFR_A [6:0] | Output | **SFR Address Bus**: The 8051 core selects external SFR circuitry via these pins. |
| SFR_WE | Output | **SFR Write Enable**: Asserted when the 8051 core is writing to external SFR circuitry. |
| SFR_OE | Output | **SFR Read Enable**: Asserted while the core is reading from external SFR circuitry. |
| SFR_DE | Input | **SFR Data Enable**: Asserted by external circuitry when writing on the SFR_I bus. If no external SFR circuitry is used, SFR_DE should be tied to ground. |
| SFR_BE | Input | **SFR Bus Enable**: Asserted by external SFR circuitry to enable communication from the 8051 core. If no external SFR circuitry is used, SFR_BE should be tied to ground. |

## Special Function Register Descriptions (Continued)

## 10. Special Function Registers

The TSCR8051 has 128 special function registers. Of these, 78 are predefined as shown in the table below. The remaining 50 undefined locations may be implemented via the external SFR bus. Read accesses to undefined locations will return unidentified data (high Z state).

**Table 2: Special Function Register Mapping**

| | 0/8 | 1/9 | 2/A | 3/B | 4/C | 5/D | 6/E | 7/F | |
|---|---|---|---|---|---|---|---|---|---|
| **F8** | | | | | | | | | **FF** |
| **F0** | B | | | | | | | | **F7** |
| **E8** | | MD0 | MD1 | MD2 | MD3 | MD4 | MD5 | ARCON | **EF** |
| **E0** | ACC | PPG | DRPG | DWPG | | | | | **E7** |
| **D8** | WDCON | | | | | | | | **DF** |
| **D0** | PSW | | | | | | | | **D7** |
| **C8** | T2CON | | CRCL | CRCH | TL2 | TH2 | | | **CF** |
| **C0** | IRCON | CCEN | CCL1 | CCH1 | CCL2 | CCH2 | CCL3 | CCH3 | **C7** |
| **B8** | IEN1 | IP1 | S0RELH | S1RELH | | | | | **BF** |
| **B0** | P3 | | | FPUS | FPUR3 | FPUR2 | FPUR1 | FPUR0 | **B7** |
| **A8** | IEN0 | IP0 | S0RELL | FPCS | OPB3 | OPB2 | OPB1 | OPB0 | **AF** |
| **A0** | P2 | | | FPUCON | OPA3 | OPA2 | OPA1 | OPA0 | **A7** |
| **98** | S0CON | S0BUF | IEN2 | S1CON | S1BUF | S1RELL | PCCON | POPC | **9F** |
| **90** | P1 | | DPS | | | | LZCON | LZC | **97** |
| **88** | TCON | TMOD | TL0 | TL1 | TH0 | TH1 | CKCON | | **8F** |
| **80** | P0 | SP | DPL | DPH | DPL1 | DPH1 | WDTREL | PCON | **87** |

## Special Function Register Descriptions

The following tables describe the predefined special function registers in the order of their addresses.

| **P0: PORT0** | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| 80h | FFh | Corresponds to the PORT0[7:0] pins. Writing a '1' to any bit allows the corresponding pin to float; writing a '0' holds the pin low ($V_{SSQ}$). See PORT0 description on page 4. |

| **Bit Map** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| PORT0[7] | PORT0[6] | PORT0[5] | PORT0[4] | PORT0[3] | PORT0[2] | PORT0[1] | PORT0[0] |

| **SP: STACK POINTER** | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| 81h | 07h | Contains the program stack location. It is incremented before PUSH and CALL instructions; the stack begins at location 08h. |

## Special Function Register Descriptions (Continued)

**DPL: DATA POINTER (LOW)**

| Address | Reset Value | Description |
|---|---|---|
| 82h | 00h | Lower byte of the first data pointer; can be accessed separately (MOV DPL,#data8) or in combination with DPH (MOV DPTR,#data16). Only used when DPS.0 = 0. Generally used to access external code (MOVC A,@A+DPTR) or data space (MOV A,@DPTR). |

**DPH: DATA POINTER (HIGH)**

| Address | Reset Value | Description |
|---|---|---|
| 83h | 00h | Upper byte of the first data pointer; can be accessed separately (MOV DPH,#data8) or in combination with DPL (MOV DPTR,#data16). See DPL, above. |

**DPL1: DATA POINTER (LOW)**

| Address | Reset Value | Description |
|---|---|---|
| 84h | 00h | Lower byte of the second data pointer; used in place of DPL when DPS.0 =1 (see DPL, above). |

**DPH1: DATA POINTER (HIGH)**

| Address | Reset Value | Description |
|---|---|---|
| 85h | 00h | Upper byte of the second data pointer; used in place of DPH when DPS.0=1 (see DPH, above). |

**WDTREL: WATCHDOG TIMER RELOAD**

| Address | Reset Value | Description |
|---|---|---|
| 86h | 00h | Bits 6-0 are loaded into the watchdog timer when a refresh is triggered by a consecutive setting of bits WDT (IEN0.6) and SWDT (IEN1.6).<br>Bit 7 is the Prescaler Select (PS) bit. When PS = 1 the watchdog is clocked through an additional divide-by-16 prescaler. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PS | Watchdog Timer Reload Value | | | | | | |

**PCON: POWER CONTROL REGISTER**

| Address | Reset Value | Description | | |
|---|---|---|---|---|
| 87h | 00h | PCON is used for general power control. Bits 6-4 are reserved. | | |
| | | **Position** | **Name** | **Bit Function** |
| | | PCON.7 | SMOD | When set, doubles the baud rate of serial port 0 in modes 1, 2, and 3. For details, see page 52. |
| | | PCON.3<br>PCON.2 | GF1<br>GF2 | General Purpose Flags |
| | | PCON.1 | PD | Power-Down: setting to 1 invokes power down (see page 58). |
| | | PCON.0 | IDL | Idle: setting to 1 invokes idle mode (see page 58). |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SMOD | – | – | – | GF1 | GF0 | PD | IDL |

## Special Function Register Descriptions (Continued)

**TCON: TIMER/COUNTER CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| 88h | 00h | TCON, along with TMOD, controls Timer 0 and Timer 1 properties. For more detail, see the discussion on page 45. |

| Position | Name | Bit Function |
|---|---|---|
| TCON.7 | TF1 | Timer 1 overflow flag, set by hardware when Timer 1 overflows. Can be cleared by software; automatically cleared when interrupt is processed. |
| TCON.6 | TR1 | Timer 1 run control bit. If cleared, Timer 1 stops. |
| TCON.5 | TF0 | Timer 0 overflow flag, set by hardware when Timer 0 overflows. Can be cleared by software; automatically cleared when interrupt is processed. |
| TCON.4 | TR0 | Timer 0 run control bit. If cleared, Timer 0 stops. |
| TCON.3 | IE1 | Interrupt 1 Edge: Set by hardware when pin INT1 triggers an interrupt. Cleared when interrupt is processed. |
| TCON.2 | IT1 | Interrupt 1 Type: Selects falling edge (1) or low level (0) on INT1 pin to trigger an interrupt. |
| TCON.1 | IE0 | Interrupt 0 Edge: Set by hardware when pin INT0 triggers an interrupt. Cleared when interrupt is processed. |
| TCON.0 | IT0 | Interrupt 0 Type: Selects falling edge (1) or low level (0) on INT0 pin to trigger an interrupt. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |

**TMOD: TIMER/COUNTER MODE CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| 89h | 00h | TMOD, along with TCON, controls Timer 0 and Timer 1 properties. For more detail, see discussion on page 45. |

| Position | Name | Bit Function |
|---|---|---|
| TMOD.7 | GATE1 | Setting GATE1 allows INT1 to act as an external gate for Timer 1. |
| TMOD.6 | C/T1 | Counter/Timer selector for Timer 1: 0 = timer; 1 = counter. |
| TMOD.5<br>TMOD.4 | M1-1<br>M0-1 | M1-1 and M0-1 select the timer/counter 1 mode (see table below). |
| TMOD.3 | GATE0 | Setting GATE0 allows INT0 to act as an external gate for Timer 0. |
| TMOD.2 | C/T0 | Counter/Timer selector for Timer 0: 0 = timer; 1 = counter. |
| TMOD.1<br>TMOD.0 | M1-0<br>M0-0 | M1-0 and M0-0 select the timer/counter 0 mode (see table below). |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| GATE1 | C/T1 | M1-1 | M0-1 | GATE0 | C/T0 | M1-0 | M0-0 |

## Special Function Register Descriptions (Continued)

| M1-x | M0-x | TMOD: Timer 0 / Timer 1 Function Table |
|---|---|---|
| 0 | 0 | 13-bit Counter/Timer; the 3 high order bits of TLx are unused (undetermined). |
| 0 | 1 | 16-bit Counter/Timer. |
| 1 | 0 | 8-bit auto-reload Counter/Timer. The reload value is in THx. When TLx overflows, THx is copied into TLx. |
| 1 | 1 | Timer 1: Halt. Timer 0: Two independent 8-bit Timers / Counters (see page 45). |

| TL0: TIMER 0 (LOW BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| 8Ah | 00h | Less significant byte of 16-bit Timer 0; the other byte is TH0. Timer 0 can be configured (using the TMOD register) as either a timer or a counter, and in any of four operating modes.<br>In timer mode, Timer 0 is incremented once every 12 clock cycles.<br>In counter mode, Timer 0 is incremented when a falling edge is observed at pin T0 (PORT3[4]).<br>Timer 0 can also be affected by the INT0 pin and the TCON register; see page 45 for details. |

| TL1: TIMER 1 (LOW BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| 8Bh | 00h | Less significant byte of 16-bit Timer 1; the other byte is TH1. Timer 1 can be configured (using the TMOD register) as either a timer or a counter, and in any of four operating modes.<br>In timer mode, Timer 1 is incremented once every 12 clock cycles.<br>In counter mode, Timer 1 is incremented when a falling edge is observed at pin T1 (PORT3[5]).<br>Timer 1 can also be affected by the INT1 pin and the TCON register; see page 45 for details. |

| TH0: TIMER 0 (HIGH BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| 8Ch | 00h | The more significant byte of 16-bit Timer 0; the other byte is TL0. For function, see TL0. |

| TH1: TIMER 1 (HIGH BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| 8Dh | 00h | The more significant byte of 16-bit Timer 1; the other byte is TL1. For function, see TL1.<br>TH1 can also set the baud rate for serial port 0; see discussion on page 52. |

## Special Function Register Descriptions (Continued)

**CKCON: CLOCK CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| 8Eh | 00h | Bits 7-3 are not implemented; bits 2-0 control the length of the memory access timer, stretching the cycle for slow memory types. Because this device contains high speed RAM, CKCON[2:0] should be left at the default high-speed setting. Any changes will degrade device performance. CKCON[2:0] stretches the memory cycle access time as shown below: |

| ckcon register | | | Stretch value | Read signals width | | Write signal width | |
|---|---|---|---|---|---|---|---|
| ckcon.2 | ckcon.1 | ckcon.0 | | memaddr | memrd | memaddr | memwr |
| 0 | 0 | 0 | 0 | 1 | 1 | 2 | 1 |
| **0** | **0** | **1** | **1** | **2** | **2** | **3** | **1** |
| 0 | 1 | 0 | 2 | 3 | 3 | 4 | 2 |
| 0 | 1 | 1 | 3 | 4 | 4 | 5 | 3 |
| 1 | 0 | 0 | 4 | 5 | 5 | 6 | 4 |
| 1 | 0 | 1 | 5 | 6 | 6 | 7 | 5 |
| 1 | 1 | 0 | 6 | 7 | 7 | 8 | 6 |
| 1 | 1 | 1 | 7 | 8 | 8 | 9 | 7 |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | CKCON.2 | CKCON.1 | CKCON.0 |

**P1: PORT1**

| Address | Reset Value | Description |
|---|---|---|
| 90h | FFh | P1 corresponds to the PORT1[7:0] pins. Writing a '1' to any P1 bit sets the corresponding pin high ($V_{DDQ}$); writing a '0' holds the pin low ($V_{SSQ}$). See PORT1 description on page 4. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PORT1[7] | PORT1[6] | PORT1[5] | PORT1[4] | PORT1[3] | PORT1[2] | PORT1[1] | PORT1[0] |

**DPS: DATA POINTER SELECT**

| Address | Reset Value | Description |
|---|---|---|
| 92h | 00h | Data pointer select – When DPS.0 is 0 (cleared), all data pointer activity uses DPH and DPL. When DPS.0 is set to 1, data pointer activity uses DPH1 and DPL1. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | DPS.0 |

## Special Function Register Descriptions (Continued)

**LZCON: LEAD ZERO COUNT CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| 96h | 0Fh | Bits 7-5 are not used; bits 4-0 controls the circuitry that counts leading zeros written to the LZC register. For this application, 'leading zeros' are 0 bits written before a 1 is written; more significant bits are written before less significant bits. Once a '1' has been written, the leading zero count does not change until the internal 32-bit leading zero count register is cleared – either by setting the LZCLR bit or by reading the least significant byte of the register while the LZM bit is set. |

| Position | Name | Bit Function |
|---|---|---|
| LZCON.4 | LZOF | Leading Zero Overflow – asserted by the hardware when the leading zero count overflows. The count uses an internal 32-bit register, so LZOF is asserted when the count reaches $2^{32}$. LZOF is read only; it is cleared when the internal 32-bit count register is cleared. |
| LZCON.3 | LZM | Leading Zero Mode – When LZM is set to 1, reading the least significant byte of the internal 32-bit leading zero count register will clear the count. |
| LZCON.2<br>LZCON.1 | LZRS1<br>LZRS0 | Leading Zero Read Select – Determines which byte of the 32-bit internal register is available in the 8-bit LZC register, as shown in the table below. The LZRS1/LZRS0 value decrements after each read of LZC so that four consecutive reads will provide all four bytes of the internal 32-bit register. The bytes are read from more significant toward less significant. After 00b (least significant byte), the value cycles to 11b (most significant byte).<br>Writing LZRS1/0 selects which byte to read next; reading LZRS1/0 reports the next byte to be read. |
| LZCON.0 | LZCLR | Leading Zero Clear – Setting LZCLR clears the internal 32-bit leading zero count register and the LZOF bit. Clearing LZCLR has no effect except changing the value of the bit itself. LZCLR is cleared each time the 8-bit LZC register is written. |

| LZRS1 | LZRS0 | Next Byte Read by LZC |
|---|---|---|
| 0 | 0 | 0 (LSB) |
| 0 | 1 | 1 |
| 1 | 0 | 2 |
| 1 | 1 | 3 (MSB) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | LZ_OF | LZM | LZRS1 | LZRS0 | LZCLR |

**LZC: LEADING ZERO COUNT**

| Address | Reset Value | Description |
|---|---|---|
| 97h | 00h | An internal 32-bit leading zero count register records the number of leading zeros written to this register. For this application, 'leading zeros' are 0 bits written before a 1 is written. More significant bits are considered to written 'before' less significant bits. Once a 1 has been written, the leading zero count does not change until the internal 32-bit leading zero count register is cleared (see the LZCON register for clearing instructions).<br>The internal 32-bit register is read a byte at a time by reading this 8-bit LZC register. The byte to be read from the internal 32-bit register is determined by bits LZRS1 and LZRS0 in LZCON. |

## Special Function Register Descriptions (Continued)

**S0CON: SERIAL PORT 0 CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| 98h | 00h | S0CON controls serial port 0 (not PORT0). For details, see page 51. |

| Position | Name | Bit Function |
|---|---|---|
| S0CON.7<br>S0CON.6 | SM0<br>SM1 | Serial Mode: determines the operating mode of serial port 0. |
| S0CON.5 | SM20 | Enables multiprocessor communication feature for serial port 0. |
| S0CON.4 | REN0 | Receive Enable: 1 enables serial reception, 0 disables reception. |
| S0CON.3 | TB80 | Transmit Bit: If serial port 0 is in mode 2 or 3, this is transmitted as the ninth data bit. Can be set or cleared to support a given function (e.g. parity or multiprocessor communication). |
| S0CON.2 | RB80 | Receive Bit: In mode 2 or 3, this receives the ninth data bit. In mode 1, it receives the stop bit (can be cleared by software). Mode 0: not used. |
| S0CON.1 | TI0 | Transmit Interrupt for serial port 0. Set by hardware after completion of a serial port 0 transmission; must be cleared by software. |
| S0CON.0 | RI0 | Receive Interrupt for serial port 0. Set by hardware after completion of a serial port 0 reception; must be cleared by software. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SM0 | SM1 | SM20 | REN0 | TB80 | RB80 | TI0 | RI0 |

**S0BUF: SERIAL PORT 0 TRANSMIT/RECEIVE BUFFER**

| Address | Reset Value | Description |
|---|---|---|
| 99h | 00h | This register accesses both a transmit buffer and a separate receive buffer. Writing to S0BUF fills the transmit buffer and starts transmission. Reading from S0BUF accesses the receive buffer. Serial port 0 can simultaneously transmit and receive. It buffers 1 byte at receive. |

**IEN2: INTERRUPT ENABLE 2**

| Address | Reset Value | Description |
|---|---|---|
| 9Ah | 00h | IEN2 is one of three registers that control the interrupt circuitry. Only one bit is supported: |

| Position | Name | Function |
|---|---|---|
| IEN2.0 | ES1 | If 0, disables the serial channel 1 interrupt. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | ES1 |

## Special Function Register Descriptions (Continued)

**S1CON: SERIAL PORT 1 CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| 9Bh | 00h | S1CON controls serial port 1 (not PORT1). For details, see page 51. |

| Position | Name | Bit Function |
|---|---|---|
| S1CON.7 | SM | Serial Mode for serial port 1: 0 = Mode A, 1 = Mode B. |
| S1CON.6 | – | Reserved |
| S1CON.5 | SM21 | Enables multiprocessor communication feature |
| S1CON.4 | REN1 | Receive Enable: 1 enables serial port reception; 0 disables reception. |
| S1CON.3 | TB81 | If serial port 1 is in mode A, this is transmitted as the ninth data bit. Can be set or cleared to support a given function (e.g. parity or multiprocessor communication). |
| S1CON.2 | RB81 | If serial port 1 is in mode A, this receives the ninth data bit. In mode B, it receives the stop bit (can be cleared by software). |
| S1CON.1 | TI1 | Transmit Interrupt for serial port 1. Set by hardware after completion of a serial port 1 transmission; must be cleared by software. |
| S1CON.0 | RI1 | Receive Interrupt for serial port 1. Set by hardware after completion of a serial port 1 reception; must be cleared by software. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SM | – | SM21 | REN1 | TB81 | RB81 | TI1 | RI1 |

**S1BUF: SERIAL PORT 1 TRANSMIT/RECEIVE BUFFER**

| Address | Reset Value | Description |
|---|---|---|
| 9Ch | 00h | This register accesses both a transmit buffer and a separate receive buffer. Writing to S1BUF fills the transmit buffer and starts transmission. Reading from S1BUF accesses the receive buffer. Serial port 1 can simultaneously transmit and receive. It buffers 1 byte at receive. |

**S1RELL: SERIAL PORT 1 RELOAD (LOW BYTE)**

| Address | Reset Value | Description |
|---|---|---|
| 9Dh | 00h | Lower byte of S1REL (serial port 1 reload register); the upper two bits are in S1RELH. Serial port 1 baud rate = System Clock Frequency / (32 x (1024 – S1REL)) |

## Special Function Register Descriptions (Continued)

**PCCON: POPULATION COUNT CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| 9Eh | 0Fh | Bits 7-5 are not used; bits 4-0 control the population count circuitry, which counts the number of '1's that are written to the population count register (POPC). |

| Position | Name | Bit Function |
|---|---|---|
| PCCON.4 | POPOF | Population Count Overflow – Read-only; asserted by the hardware when the population count overflows. The count is stored in a 32-bit register, so POPOF is asserted when the count reaches $2^{32}$. POPF is cleared by setting POPCLR or by reading the least significant byte of the internal 32-bit count register when the POPM bit is set. |
| PCCON.3 | POPM | Population Count Mode – When set to 1, the population count is reset by reading the 8-bit POPC register when POPRS1=0 and POPRS0=0; this reads the least significant byte of the internal 32-bit count register. |
| PCCON.2<br>PCCON.1 | POPRS1<br>POPRS0 | Population Read Select – Determines which byte of the 32-bit internal population count register is to be read via the 8-bit POPC register. Writing POPRS1/0 selects the byte to read; reading these bits shows which byte will be read. The POPRS1/POPRS0 value decrements after each read of POPC, so four consecutive reads will provide all four bytes of the internal 32-bit population count register. The bytes are read from more significant toward less significant. After 00b (least significant byte), the value cycles to 11b (most significant byte). |
| PCCON.0 | POPCLR | Population Count Clear – Setting this bit clears the internal 32-bit population count register and the Population Count Overflow Flag (POPOF). Clearing POPCLR has no effect except changing the value of the bit itself. POPCLR is cleared each time POPC is written. |

| POPRS1 | POPRS0 | Next Byte to Read via POPC |
|---|---|---|
| 0 | 0 | 0 (LSB) |
| 0 | 1 | 1 |
| 1 | 0 | 2 |
| 1 | 1 | 3 (MSB) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | POPOF | POPM | POPRS1 | POPRS0 | POPCLR |

**POPC: POPULATION COUNT**

| Address | Reset Value | Description |
|---|---|---|
| 9Fh | 00h | An internal 32-bit population counter records the number of '1' bits written to POPC. The count increases with every '1' written to POPC until the internal counter overflows or is cleared (see PCCON for clearing instructions).<br>The internal 32-bit population counter is read a byte at a time by reading POPC, controlled by bits POPRS1 and POPRS0 in the PCCON register. |

## Special Function Register Descriptions (Continued)

| **P2: PORT2 CONTROL** | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| A0h | 00h | P2 corresponds to the PORT2[7:0] pins. Writing a '1' to any bit of P2 sets the corresponding pin high ($V_{DDQ}$); writing a '0' holds the pin low ($V_{SSQ}$). See PORT2 description in section 8. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PORT2[7] | PORT2[6] | PORT2[5] | PORT2[4] | PORT2[3] | PORT2[2] | PORT2[1] | PORT2[0] |

| **FPUCON: FLOATING-POINT UNIT CONTROL** | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| A3h | 00h | Bits 7-5 are not used; bits 4-0 control the function of the floating-point unit (FPU). |

| **Position** | **Name** | **Bit Function** |
|---|---|---|
| FPUCON.4<br>FPUCON.3 | FPU_M1<br>FPU_M0 | FPU Rounding Mode: |

| **FPU_M1** | **FPU_M0** | **Rounding Mode** |
|---|---|---|
| 0 | 0 | Round to nearest even number |
| 0 | 1 | Round to zero |
| 1 | 0 | Round up (to +INF, positive infinity) |
| 1 | 1 | Round down (to –INF, negative infinity) |

| **Position** | **Name** | **Bit Function** |
|---|---|---|
| FPUCON.2<br>FPUCON.1<br>FPUCON.0 | FPU_OP2<br>FPU_OP1<br>FPU_OP0 | FPU Operation: |

| **FPU_OP2** | **FPU_OP1** | **FPU_OP0** | **Operation** |
|---|---|---|---|
| 0 | 0 | 0 | Add A and B |
| 0 | 0 | 1 | Subtract B from A |
| 0 | 1 | 0 | Multiply A by B |
| 0 | 1 | 1 | Divide A by B |
| 1 | 0 | 0 | Convert Integer A to Float |
| 1 | 0 | 1 | Convert Float A to Integer |
| 1 | 1 | x | Undefined (Reserved) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | FPU_M1 | FPU_M0 | FPU_OP2 | FPU_OP1 | FPU_OP0 |

| **OPA3: FPU FLOATING POINT OPERAND A3 (MSB)** | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| A4h | 00h | Contains the most significant byte of the 32-bit Floating Point Operand A (OPA). |

| **OPA2: FPU FLOATING POINT OPERAND A2** | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| A5h | 00h | Contains the second most significant byte of the 32-bit Floating Point Operand A (OPA). |

## Special Function Register Descriptions (Continued)

| OPA1: FPU FLOATING POINT OPERAND A1 | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| A6h | 00h | Contains the second least significant byte of the 32-bit Floating Point Operand A (OPA). |

| OPA0: FPU FLOATING POINT OPERAND A0 (LSB) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| A7h | 00h | Contains the least significant byte of the 32-bit Floating Point Operand A (OPA). |

**IEN0: INTERRUPT ENABLE 0**

| Address | Reset Value | Description | | |
|---|---|---|---|---|
| A8h | 00h | IEN0 controls the interrupt circuitry (with IEN1 and IEN2). In addition, bit 6 (WDT) is part of the watchdog timer circuitry. | | |
| | | **Position** | **Name** | **Bit Function** |
| | | IEN0.7 | EAL | If 0, disables all interrupts. |
| | | IEN0.6 | WDT | Watchdog timer refresh flag, set to initiate a refresh of the watchdog timer. WDT must be set directly before SWDT (IEN1.6) to refresh the watchdog timer. WDT is reset by hardware 12 clock cycles after it has been set. |
| | | IEN0.5 | ET2 | If 0, disables timer 2 overflow and external reload interrupts. |
| | | IEN0.4 | ES0 | If 0, disables the serial channel 0 interrupt. |
| | | IEN0.3 | ET1 | If 0, disables the Timer 1 overflow interrupt. |
| | | IEN0.2 | EX1 | If 0, disables external interrupt 1. |
| | | IEN0.1 | ET0 | If 0, disables the Timer 0 overflow interrupt. |
| | | IEN0.0 | EX0 | If 0, disables external interrupt 0. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EAL | WDT | ET2 | ES0 | ET1 | EX1 | ET0 | EX0 |

## Special Function Register Descriptions (Continued)

**IP0: INTERRUPT PRIORITY 0**

| Address | Reset Value | Description |
|---|---|---|
| A9h | 00h | IP0, combined with IP1, sets the priority level for each of the six interrupt groups. In addition, bits 6 & 7 are part of the watchdog circuitry. There are four interrupt priority levels: |

| IP1.x | IP0.x | Priority Level |
|---|---|---|
| 0 | 0 | 0 (lowest) |
| 0 | 1 | 1 |
| 1 | 0 | 2 |
| 1 | 1 | 3 (highest) |

| Position | Name | Bit Function |
|---|---|---|
| IP0.7 | OWDS | Oscillator Watchdog Status (not supported) |
| IP0.6 | WDTS | Watchdog Timer Status: Set by the hardware when the watchdog timer value reaches 7CFFh; reset begins two clock cycles later. |
| IP0.5 | IP0.5 | Lower bit, interrupt group 5 priority (Timer 2, External Interrupt 6) |
| IP0.4 | IP0.4 | Lower bit, interrupt group 4 priority (Serial channel 0, External Interrupt 5) |
| IP0.3 | IP0.3 | Lower bit, interrupt group 3 priority (Timer 1, External Interrupt 4) |
| IP0.2 | IP0.2 | Lower bit, interrupt group 2 priority (External Interrupts 1 and 3) |
| IP0.1 | IP0.1 | Lower bit, interrupt group 1 priority (Timer 0, External Interrupt 2) |
| IP0.0 | IP0.0 | Lower bit, interrupt group 0 priority (Serial Channel 1, External Interrupt 0) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OWDS | WDTS | IP0.5 | IP0.4 | IP0.3 | IP0.2 | IP0.1 | IP0.0 |

**S0RELL: SERIAL PORT 0 RELOAD (LOW BYTE)**

| Address | Reset Value | Description |
|---|---|---|
| AAh | D9h | Lower byte of the serial port 0 reload register (S0REL); the upper two bits are in S0RELH. When serial port 0 is in mode 1 or 3 and BD = 1, then:<br>serial port 0 baud rate = $2^{SMOD}$ x System Clock Frequency / (64 x (1024 – S0REL))<br>(Mode is determined by S0CON; SMOD is PCON.7; BD is WDCON.7) |

## Special Function Register Descriptions (Continued)

| FPCS: FLOATING POINT COMPARATOR STATUS | | | | |
|---|---|---|---|---|
| **Address** | **Reset Value** | **Description** | | |
| ABh | 00h | Compares the FPU operand registers (OPA and OPB) to each other and against special values: infinity, zero, and Not a Number [NaN]. Bits 7 and 6 are not used. | | |
| | | **Position** | **Name** | **Bit Function** |
| | | FPCS.5 | UNORD | Unordered: Set when either OPA or OPB is “Not a Number” (NaN). For more details, see FPU description on page 57. |
| | | FPCS.4 | ALTB | A < B: Set when OPA is less than OPB. |
| | | FPCS.3 | BLTA | B < A: Set when OPB is less than OPA. |
| | | FPCS.2 | AEQB | A = B: Set when OPA and OPB are equal. |
| | | FPCS.1 | OP_INF | Operand Infinite: Set when either OPA or OPB is infinite. |
| | | FPCS.0 | OP_ZERO | Operand Zero: Set when OPA is zero. |

| Bit Map | | | | | | | |
|---|---|---|---|---|---|---|---|
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| - | - | UNORD | ALTB | BLTA | AEQB | OP_INF | OP_ZERO |

| OPB3: FPU FLOATING POINT OPERAND B3 (MSB) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| ACh | 00h | Contains the most significant byte of 32-bit Floating Point Operand B (OPB). |

| OPB2: FPU FLOATING POINT OPERAND B2 | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| ADh | 00h | Contains the second most significant byte of 32-bit Floating Point Operand B (OPB). |

| OPB1: FPU FLOATING POINT OPERAND B1 | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| AEh | 00h | Contains the second least significant byte of 32-bit Floating Point Operand B (OPB). |

| OPB0: FPU FLOATING POINT OPERAND B0 (LSB) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| AFh | 00h | Contains the least significant byte of 32-bit Floating Point Operand B (OPB). |

| P3: PORT3 CONTROL | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| B0h | FFh | P3 corresponds to the PORT3[7:0] pins. Writing a ‘1’ to P3 sets the corresponding pin high ($V_{DDQ}$); writing a ‘0’ holds the pin low ($V_{SSQ}$). See description of PORT3 in section 8. |

| Bit Map | | | | | | | |
|---|---|---|---|---|---|---|---|
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| PORT3[7] | PORT3[6] | PORT3[5] | PORT3[4] | PORT3[3] | PORT3[2] | PORT3[1] | PORT3[0] |

## Special Function Register Descriptions (Continued)

**FPUS: FLOATING POINT UNIT STATUS**

| Address | Reset Value | Description |
|---|---|---|
| B3h | 00h | FPUS reports special values in the FPU results (FPUR) or in an operand (OPA or OPB). |

| Position | Name | Bit Function |
|---|---|---|
| FPUS.7 | SNAN | Signaling Not A Number (SNaN): Set when either of the FPU operands is an SNaN. For more details, see discussion on page 57. |
| FPUS.6 | QNAN | Quiet Not A Number (QNaN): Set when the FPU result is a QNaN. For more details, see discussion on page 57. |
| FPUS.5 | INF | Infinity: Set when the FPU result is infinite. |
| FPUS.4 | INE | Inexact: Set when the FPU result is inexact. |
| FPUS.3 | OVRFLW | Overflow: Set when an FPU operation uses a floating-point number with an absolute value greater than $(2\text{-}2^{-23}) \times 2^{127}$. |
| FPUS.2 | UFLW | Underflow: Set when an FPU operation uses a floating-point number that has a non-zero absolute value less than $2^{-149}$. |
| FPUS.1 | DBZ | Divide by Zero: Set when FPU operation is set to divide (see FPUCON) and operand B (OPB) is set to zero. |
| FPUS.0 | ZERO | Zero: Set when the FPU operation result is zero. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SNAN | QNAN | INF | INE | OVRFLW | UFLW | DBZ | ZERO |

**FPUR3: FPU FLOATING POINT RESULT 3 (MSB)**

| Address | Reset Value | Description |
|---|---|---|
| B4h | 00h | The most significant byte of the 32-bit floating-point unit result (FPUR) of an FPU operation. |

**FPUR2: FPU FLOATING POINT RESULT 2**

| Address | Reset Value | Description |
|---|---|---|
| B5h | 00h | Second most significant byte of the 32-bit floating-point unit result (FPUR). |

**FPUR1 – FPU FLOATING POINT RESULT 1**

| Address | Reset Value | Description |
|---|---|---|
| B6h | 00h | Second least significant byte of the 32-bit floating-point unit result (FPUR) of an FPU operation. |

**FPUR0 – FPU FLOATING POINT RESULT 0 (LSB)**

| Address | Reset Value | Description |
|---|---|---|
| B7h | 00h | Least significant byte of the 32-bit floating-point unit result (FPUR) of an FPU operation. |

## Special Function Register Descriptions (Continued)

**IEN1: INTERRUPT ENABLE 1**

| Address | Reset Value | Description |
|---|---|---|
| B8h | 00h | IEN1 controls the interrupt circuitry (with IEN0 and IEN2). In addition, bit 6 (SWDT) is part of the watchdog timer circuitry. |

| Position | Name | Bit Function |
|---|---|---|
| IEN1.7 | EXEN2 | If 0, disables the Timer 2 external reload interrupt. |
| IEN1.6 | SWDT | Start Watchdog Timer: If the timer is not running, setting SWDT activates it. If the timer is running, setting SWDT directly after setting WDT (IEN0.6) performs a watchdog timer refresh. SWDT is cleared by the hardware 12 clock cycles after it has been set. |
| IEN1.5 | EX6 | If 0, disables external interrupt 6 [INT6] |
| IEN1.4 | EX5 | If 0, disables external interrupt 5 [INT5] |
| IEN1.3 | EX4 | If 0, disables external interrupt 4 [INT4] |
| IEN1.2 | EX3 | If 0, disables external interrupt 3 [INT3] |
| IEN1.1 | EX2 | If 0, disables external interrupt 2 [INT2] |
| IEN1.0 | EADC | Enable A/D Converter (not supported). |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EXEN2 | SWDT | EX6 | EX5 | EX4 | EX3 | EX2 | EADC |

**IP1: INTERRUPT PRIORITY 1**

| Address | Reset Value | Description |
|---|---|---|
| B9h | 00h | IP1, combined with IP0, sets the priority level for each of the six interrupt groups. There are four priority levels: |

| IP1.x | IP0.x | Priority Level |
|---|---|---|
| 0 | 0 | 0 (lowest) |
| 0 | 1 | 1 |
| 1 | 0 | 2 |
| 1 | 1 | 3 (highest) |

| Position/Name | Function |
|---|---|
| IP1.5 | Upper bit, interrupt group 5 priority (Timer 2, External Interrupt 6) |
| IP1.4 | Upper bit, interrupt group 4 priority (Serial channel 0, External Interrupt 5) |
| IP1.3 | Upper bit, interrupt group 3 priority (Timer 1, External Interrupt 4) |
| IP1.2 | Upper bit, interrupt group 2 priority (External Interrupts 1 and 3) |
| IP1.1 | Upper bit, interrupt group 1 priority (Timer 0, External Interrupt 2) |
| IP1.0 | Upper bit, interrupt group 0 priority (Serial Channel 1, External Interrupt 0) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | IP1.5 | IP1.4 | IP1.3 | IP1.2 | IP1.1 | IP1.0 |

## Special Function Register Descriptions (Continued)

| **S0RELH: SERIAL PORT 0 RELOAD (UPPER 2 BITS)** | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| BAh | 03h | Contains the upper two bits of S0REL (serial port 0 reload register); the lower byte is in S0RELL. See S0RELL for functional description. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | S0RELH.1 | S0RELH.0 |

| **S1RELH: SERIAL PORT 1 RELOAD (UPPER 2 BITS)** | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| BBh | 03h | Contains the upper two bits of S1REL (serial port 1 reload register; the lower byte is in S1RELL. See S1RELL for functional description. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | S1RELH.1 | S1RELH.0 |

**IRCON: INTERRUPT REQUEST**

| **Address** | **Reset Value** | **Description** | | |
|---|---|---|---|---|
| C0h | 00h | Bits are set by Timer 2 and external interrupts and must be cleared by software. | | |
| | | **Position** | **Name** | **Bit Function** |
| | | IRCON.7 | EXF2 | Timer 2 external reload flag |
| | | IRCON.6 | TF2 | Timer 2 overflow flag |
| | | IRCON.5 | IEX6 | External Interrupt 6 [INT6] Edge flag |
| | | IRCON.4 | IEX5 | External Interrupt 5 [INT5] Edge flag |
| | | IRCON.3 | IEX4 | External Interrupt 4 [INT4] Edge flag |
| | | IRCON.2 | IEX3 | External Interrupt 3 [INT3] Edge flag |
| | | IRCON.1 | IEX2 | External Interrupt 2 [INT2] Edge flag |
| | | IRCON.0 | IADC | A to D Converter Interrupt (not supported) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EXF2 | TF2 | IEX6 | IEX5 | IEX4 | IEX3 | IEX2 | IADC |

## Special Function Register Descriptions (Continued)

**CCEN: COMPARE / CAPTURE ENABLE**

| Address | Reset Value | Description |
|---|---|---|
| C1h | 00h | Sets the mode of the Capture/Reload/Compare and Capture/Compare registers (CRC, CC1, CC2, and CC3). Each register is controlled by two bits, COCAHx and COCALx: |

| COCAHx | COCALx | Compare / Capture Mode |
|---|---|---|
| 0 | 0 | Compare / capture disabled |
| 0 | 1 | Capture on the rising edge of pin CCx (See bit I3FR [T2CON.6] for CC0 falling edge detection) |
| 1 | 0 | Compare enabled |
| 1 | 1 | Capture on write operation into register |

| Position | Name | Bit Function |
|---|---|---|
| CCEN.7 | COCAH3 | Compare/Capture Mode Select for CC3 (high) |
| CCEN.6 | COCAL3 | Compare/Capture Mode Select for CC3 (low) |
| CCEN.5 | COCAH2 | Compare/Capture Mode Select for CC2 (high) |
| CCEN.4 | COCAL2 | Compare/Capture Mode Select for CC2 (low) |
| CCEN.3 | COCAH1 | Compare/Capture Mode Select for CC1 (high) |
| CCEN.2 | COCAL1 | Compare/Capture Mode Select for CC1 (low) |
| CCEN.1 | COCAH0 | Compare/Capture Mode Select for CRC (high) |
| CCEN.0 | COCAL0 | Compare/Capture Mode Select for CRC (low) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| COCAH3 | COCAL3 | COCAH2 | COCAL2 | COCAH1 | COCAL1 | COCAH0 | COCAL0 |

**CCL1: COMPARE / CAPTURE 1 (LOW BYTE)**

| Address | Reset Value | Description |
|---|---|---|
| C2h | 00h | Less significant byte of CC1 (16-bit Compare/Capture register 1); the other byte is CCH1. Depending on the mode set in CCEN, CC1 either captures the value of Timer 2 or compares against the value of Timer 2. |

**CCH1: COMPARE / CAPTURE 1 (HIGH BYTE)**

| Address | Reset Value | Description |
|---|---|---|
| C3h | 00h | More significant byte of CC1; the other byte is CCL1. See description in CCL1. |

**CCL2: COMPARE / CAPTURE 2 (LOW BYTE)**

| Address | Reset Value | Description |
|---|---|---|
| C4h | 00h | Less significant byte of CC2 (16-bit Compare/Capture Register 2); the other byte is CCH2. Depending on the mode set in CCEN, CC2 either captures the value of Timer 2 or compares against the value of Timer 2. |

**CCH2: COMPARE / CAPTURE 2 (HIGH BYTE)**

| Address | Reset Value | Description |
|---|---|---|
| C5h | 00h | More significant byte CC2; the other byte is CCL2. See description in CCL2. |

## Special Function Register Descriptions (Continued)

| CCL3: COMPARE / CAPTURE 3 (LOW BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| C6h | 00h | Less significant byte of CC3 (16-bit Compare/Capture Register 3); the other byte is CCH3. Depending on the mode set in CCEN, CC3 either captures the value of Timer 2 or compares against the value of Timer 2. |

| CCH3: COMPARE / CAPTURE 3 (HIGH BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| C7h | 00h | More significant byte of CC3; the other byte is CCL3. See description in CCL3. |

**T2CON: TIMER 2 CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| C8h | 00h | Controls Timer 2 properties. In addition, bits 5 and 6 select active edges for INT2 and INT3. |

| Position | Name | Bit Function |
|---|---|---|
| T2CON.7 | T2PS | Timer 2 Prescaler Select: 0 = 1/12 system clock, 1 = 1/24 system clock |
| T2CON.6 | I3FR | Selects active edge for INT3: 0 = Falling edge, 1 = Rising edge |
| T2CON.5 | I2FR | Selects active edge for INT2: 0 = Falling edge, 1 = Rising edge |
| T2CON.4<br>T2CON.3 | T2R1<br>T2R0 | Timer 2 Reload Mode |

| T2R1 | T2R0 | Reload Mode |
|---|---|---|
| 0 | x | Reload disabled |
| 1 | 0 | Mode 0: Reload is triggered by Timer 2 overflow |
| 1 | 1 | Mode 1: Reload is triggered by negative transition of pin T2EX (PORT1[5]) |

| | | |
|---|---|---|
| T2CON.2 | T2CM | Timer 2 Compare Mode<br>0: If Timer 2 matches a compare register, the corresponding pin CCx is set high until the next Timer 2 overflow.<br>1: If Timer 2 matches a compare register, the pre-written value in P1.x is sent to pin CCx. Overflow does not cause any change. |
| T2CON.1<br>T2CON.0 | T2I1<br>T2I0 | Timer 2 Input Mode |

| T2I1 | T2I0 | Timer 2 Input Mode |
|---|---|---|
| 0 | 0 | Timer 2 stops. |
| 0 | 1 | Timer 2 is a timer, incremented according to T2PS. |
| 1 | 0 | Timer 2 is a counter, incremented by an external signal at pin T2 (PORT1[7]). |
| 1 | 1 | Timer 2 is a gated timer, incremented according to T2PS and started/stopped by external signals on pin T2. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T2PS | I3FR | I2FR | T2R1 | T2R0 | T2CM | T2I1 | T2I0 |

## Special Function Register Descriptions (Continued)

| CRCL: COMPARE / RELOAD / CAPTURE (LOW BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| CAh | 00h | Less significant byte of CRC (16-bit Compare/Reload/Capture register); the other byte is CRCH. Depending on the mode set by CCEN, CRC either captures the value or compares against the value of Timer 2. |

| CRCH: COMPARE / RELOAD / CAPTURE (HIGH BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| CBh | 00h | More significant byte of the 16-bit CRC; the other byte is CRCL. See description in CRCL. |

| TL2: TIMER 2 (LOW BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| CCh | 00h | Less significant byte of 16-bit Timer 2; the other byte is TH2. Timer 2 is configured by T2CON. |

| TH2: TIMER 2 (HIGH BYTE) | | |
|---|---|---|
| **Address** | **Reset Value** | **Description** |
| CDh | 00h | More significant byte of 16-bit Timer 2; the other byte is TL2. Timer 2 is configured by T2CON. |

**PSW: PROGRAM STATUS WORD**

| Address | Reset Value | Description | | |
|---|---|---|---|---|
| D0h | 00h | PSW contains program status information. | | |
| | | **Position** | **Name** | **Bit Function** |
| | | PSW.7 | CY | Carry Flag |
| | | PSW.6 | AC | Auxiliary Carry flag for Binary Coded Decimal (BCD) operations |
| | | PSW.5 | F0 | General Purpose Flag 0, available to user software |
| | | PSW.4<br>PSW.3 | RS1<br>RS0 | RS1 and RS0 select the register bank: |
| | | PSW.2 | OV | Overflow Flag |
| | | PSW.1 | – | User Defined Flag |
| | | PSW.0 | P | Parity Flag – An even number of '1' bits in the accumulator sets this bit (even parity), an odd number of ones clears it (odd parity). |

| RS1 | RS0 | Bank | Location |
|---|---|---|---|
| 0 | 0 | 0 | 0h – 7h |
| 0 | 1 | 1 | 8h – Fh |
| 1 | 0 | 2 | 10h – 17h |
| 1 | 1 | 3 | 18h – 1Fh |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CY | AC | F0 | RS1 | RS0 | OV | - | P |

## Special Function Register Descriptions (Continued)

**WDCON: POWER FAIL CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| D8h | 00h | Only bit 7 (BD) is supported. BD controls the baud rate of serial port 0 in modes 1 and 3. Other registers that affect the serial port 0 baud rate include S0RELL, S0RELH, TH1, PCON.7 (SMOD bit), and S0CON. |

| Position | Name | Bit Function |
|---|---|---|
| WDCON.7 | BD | 0: Serial port 0 baud rate = $\frac{2^{SMOD} \times \text{System Clock Frequency}}{384 \times (256 - TH1)}$<br>1: Serial port 0 baud rate = $\frac{2^{SMOD} \times \text{System Clock Frequency}}{64 \times (1024 - S0REL)}$ |

**Bit Map**

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

**ACC: ACCUMULATOR**

| Address | Reset Value | Description |
|---|---|---|
| E0h | 00h | Accumulator. Most instructions use the accumulator to hold the operand. The mnemonics for accumulator-specific instructions refer to accumulator as A, not ACC. |

**PPG: PROGRAM MEMORY PAGING**

| Address | Reset Value | Description |
|---|---|---|
| E1h | 00h | Bits 7-3 are not used; bits 2-0 select the memory page to use as program memory. The location of each page in physical memory depends upon the state of the PMODE pin.<br>When PMODE is low there are two 64 KByte pages of SRAM memory.<br>When PMODE is high there are eight 16 KByte pages of memory.<br>For full description, see section 11.2 on page 32. |

| Position | Name | Bit Function |
|---|---|---|
| PPG.2 | PPM2 | Program Page Memory Select 2 (msb) |
| PPG.1 | PPM1 | Program Page Memory Select 1 |
| PPG.0 | PPM0 | Program Page Memory Select 0 (lsb) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | PPM2 | PPM1 | PPM0 |

## Special Function Register Descriptions (Continued)

**DRPG: DATE READ MEMORY PAGING**

| Address | Reset Value | Description |
|---|---|---|
| E2h | 01h | Bits 7-3 are not used; bits 2-0 select which memory page to read as data memory. The location of each page in physical memory depends on the state of the PMODE pin. See PPG register (page 28) for details. |

| Position | Name | Bit Function |
|---|---|---|
| DRPG.2 | DRPM2 | Data Read Page Memory Select 2 (msb) |
| DRPG.1 | DRPM1 | Data Read Page Memory Select 1 |
| DRPG.0 | DRPM0 | Data Read Page Memory Select 0 (lsb) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | DRPM2 | DRPM1 | DRPM0 |

**DWPG: DATE WRITE MEMORY PAGING**

| Address | Reset Value | Description |
|---|---|---|
| E3h | 01h | Bits 7-3 are not used; bits 2-0 select which memory page to write as data memory. The location of each page in physical memory depends upon the state of the PMODE pin. See PPG register (page 28) for details. **Note:** Writing to page 0 is not allowed. |

| Position | Name | Bit Function |
|---|---|---|
| DWPG.2 | DWPM2 | Data Write Page Memory Select 2 (msb) |
| DWPG.1 | DWPM1 | Data Write Page Memory Select 1 |
| DWPG.0 | DWPM0 | Data Write Page Memory Select 0 (lsb) |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | DWPM2 | DWPM1 | DWPM0 |

**MD0: MULTIPLICATION / DIVISION 0**

| Address | Reset Value | Description |
|---|---|---|
| E9h | 00h | One of six registers that hold MDU operands (write) and results (read). Its function varies depending on the operation being performed: |

| Arithmetic Operation | MD0 Function (Write) | MD0 Function (Read) |
|---|---|---|
| 32-bit / 16-bit<br>16-bit / 16-bit | Dividend LSB | Quotient LSB |
| 16-bit x 16-bit | Multiplicand LSB | Product LSB |
| 32-bit Shift Register<br>32-bit Normalize | LSB | |

## Special Function Register Descriptions (Continued)

**MD1: MULTIPLICATION / DIVISION 1**

| Address | Reset Value | Description |
|---|---|---|
| EAh | 00h | One of six registers that hold MDU operands (write) and results (read). Its function varies depending on the operation being performed: |

| Arithmetic Operation | MD1 Function (Write) | MD1 Function (Read) |
|---|---|---|
| 32-bit / 16-bit | Dividend Second LSB | Quotient Second LSB |
| 16-bit / 16-bit | Dividend MSB | Quotient MSB |
| 16-bit x 16-bit | Multiplicand MSB | Product Second LSB |
| 32-bit Shift Register<br>32-bit Normalize | Second LSB | |

**MD2: MULTIPLICATION / DIVISION 2**

| Address | Reset Value | Description |
|---|---|---|
| EBh | 00h | One of six registers that hold MDU operands (write) and results (read). Its function varies depending on the operation being performed: |

| Arithmetic Operation | MD2 Function (Write) | MD2 Function (Read) |
|---|---|---|
| 32-bit / 16-bit | Dividend Second MSB | Quotient Second MSB |
| 16-bit / 16-bit | Dividend MSB | Not used |
| 16-bit x 16-bit | Not used | Product Second MSB |
| 32-bit Shift Register<br>32-bit Normalize | Second MSB | |

**MD3: MULTIPLICATION / DIVISION 3**

| Address | Reset Value | Description |
|---|---|---|
| ECh | 00h | One of six registers that hold MDU operands (write) and results (read). Its function varies depending on the operation being performed: |

| Arithmetic Operation | MD3 Function (Write) | MD3 Function (Write) |
|---|---|---|
| 32-bit / 16-bit | Dividend MSB | Quotient MSB |
| 16-bit / 16-bit | Not used | |
| 16-bit x 16-bit | Not used | Product MSB |
| 32-bit Shift Register<br>32-bit Normalize | MSB | |

**MD4: MULTIPLICATION / DIVISION 4**

| Address | Reset Value | Description |
|---|---|---|
| EDh | 00h | One of six registers that hold MDU operands (write) and results (read). Its function varies depending on the operation being performed: |

| Arithmetic Operation | MD4 Function (Write) | MD4 Function (Read) |
|---|---|---|
| 32-bit / 16-bit<br>16-bit / 16-bit | Divisor LSB | Remainder LSB |
| 16-bit x 16-bit | Multiplier LSB | Not used |
| 32-bit Shift Register<br>32-bit Normalize | Not used | |

## Special Function Register Descriptions (Continued)

**MD5: MULTIPLICATION / DIVISION 5**

| Address | Reset Value | Description |
|---|---|---|
| EEh | 00h | One of six registers hold MDU operands (write) and results (read). Its function varies depending on the operation being performed: |

| Arithmetic Operation | MD5 Function(Write) | MD5 Function(Read) |
|---|---|---|
| 32-bit / 16-bit<br>16-bit / 16-bit | Divisor MSB | Remainder MSB |
| 16-bit x 16-bit | Multiplier MSB | Not used |
| 32-bit Shift Register<br>32-bit Normalize | Not used | |

**ARCON: ARITHMETIC CONTROL**

| Address | Reset Value | Description |
|---|---|---|
| EFh | 00h | ARCON controls the functions of the MDU (Multiplication/Division Unit). |

| Position | Name | Bit Function |
|---|---|---|
| ARCON.7 | MDEF | Multiply Divide Error Flag, set by the hardware when an operation is performed improperly (restarted or interrupted). |
| ARCON.6 | MDOV | Multiply Divide Overflow Flag |
| ARCON.5 | SLR | Shift Direction: SLR = 1 = shift left SLR = 0 = shift right |
| ARCON.4<br>ARCON.3<br>ARCON.2<br>ARCON.1<br>ARCON.0 | SC.4<br>SC.3<br>SC.2<br>SC.1<br>SC.0 | Five-bit shift counter (SC). Setting SC to zero selects “normalize”. After the normalize function is completed, SC contains the number of normalization shifts that were performed. Setting SC to a non-zero value selects “Shift” and specifies the number of shifts to be performed. |

**Bit Map**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MDEF | MDOV | SLR | SC.4 (MSB) | SC.3 | SC.2 | SC.1 | SC.0 (LSB) |

**B: B REGISTER**

| Address | Reset Value | Description |
|---|---|---|
| F0h | 00h | B is used during multiply and divide instructions. It can also be used as a scratch-pad register to hold temporary data. |

## 11. Memory

The TSCR8051 contains 256 bytes of “scratch pad” memory and 128KB of SRAM.

### 11.1. Scratch Pad Memory

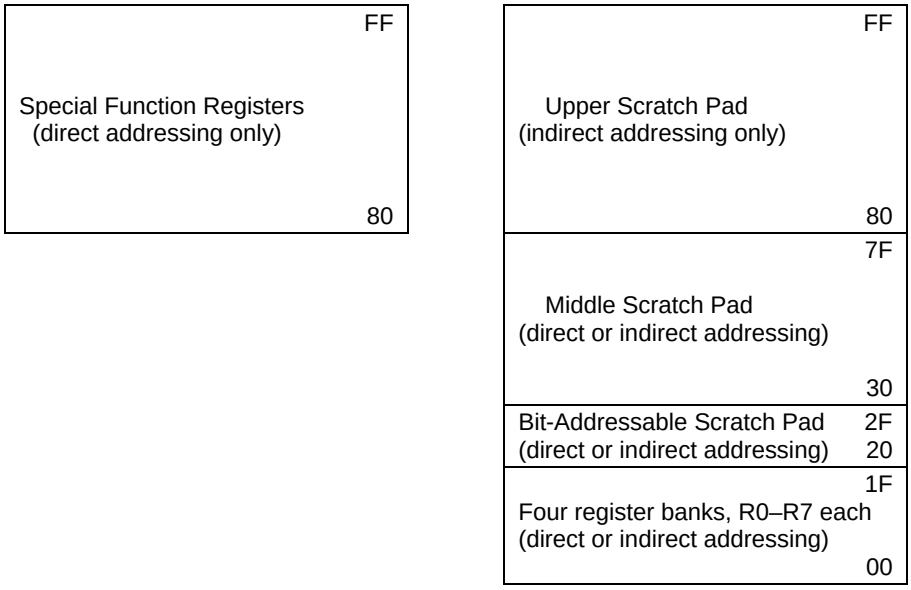
Internal “scratch pad” memory is 256 bytes (00 to FF). Addressing for this data area is always one byte wide.

The upper 128 bytes of scratch pad memory (80 to FF) overlaps the Special Function Registers (SFRs). Direct addressing accesses the SFRs; indirect addressing accesses the upper scratch pad.

The lower 128 bytes of scratch pad memory may be addressed either directly or indirectly. It is further divided into three sections:

The bottommost 32 bytes contain four register banks, with registers R0 to R7 in each bank. Bits RS0 and RS1 in the PSW register determine which register bank is in use.

**Figure 3 – Scratch Pad Memory**




### 11.2. SRAM Memory

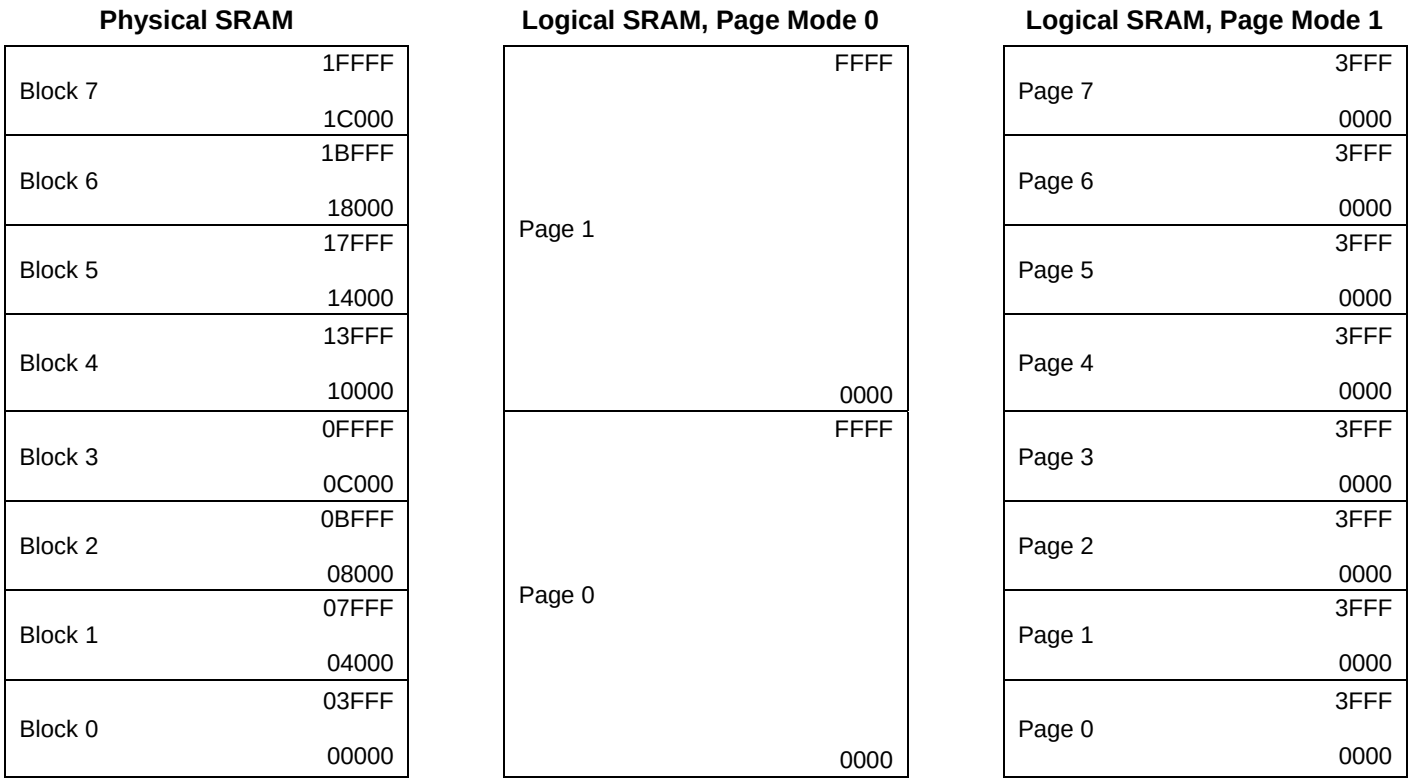
The TSC8051 contains 128 KBytes of high-speed SRAM that support both program and data memory, using either of two paging modes as determined by the PMODE pin. Memory organization for both modes is shown in Figure 4 below.

When PMODE is low (mode 0), SRAM memory is divided into two logical pages of 64 KBytes each.

When PMODE is high (mode 1), there are eight logical pages of SRAM memory with 16 KBytes in each page.

**Figure 4 – SRAM Memory Layout**




### 11.2.1. Address Mapping

In either paging mode, the program specifies a memory location with a three-bit Page number and a sixteen-bit Address. Memory addressing is mapped as follows:

| | | | |
|---|---|---|---|
| Physical Address: | Block (5 bits) | Address [13:0] | |
| Mode 0 Logical Address: | Page (3 bits) | Address [15:0] | |
| Mode 1 Logical Address: | Addr. [15:14] | Page (3 bits) | Address [13:0] |

In the physical address, the two most significant bits of the Block are always 00.

In mode 0, the two most significant bits of the Page are always 00.

In mode 1, the two most significant bits of the Address (Addr.[15:14]) are always 00.

### 11.2.2. Specifying the Page Number

Page numbers are specified in three different registers – PPG, DRPG, and DWPG. The PPG register specifies the current page for program memory – it defaults to 0. DRPG specifies the current page for data reads and DWPG for data writes; both of these default to 1. Page register usage is determined by the type of instruction being performed:

MOVC and program fetch instructions use the page number in PPG.

MOVX @Ri,A and MOVX @DPTR,A use DWPG.

MOVX A,@R1 and MOVX A,@DPTR use DRPG.

### 11.2.3. Program and Data Addressing

Address pointers for program and data use 16 bits; paging adds another three bits to each address, giving a logical address range of 00000 to 7FFFF. There is no physical distinction between program memory and data memory – the entire SRAM data area is available for both program and data.

## 11.3. Dual Data Pointers

DPRT is the standard 16-bit data pointer, made up of registers DPL and DPH. A secondary data pointer, DPTR1, is stored in registers DPL1 and DPH1. The active pointer for any DPTR-related instruction is determined by the value of register DPS. When moving large blocks of data, the user can accelerate the process by storing the source address in one pointer an the destination in the other, and switching between pointers by toggling the DSP.0 bit.

# 12. Instruction Set

All TSCR8051 instructions are binary code compatible with the industry standard 8051. The following tables give a summary of the instruction set.

Table 3 and Table 4 contain notes for mnemonics used in Instruction Set tables.

Table 5 through Table 9 show instruction hexadecimal codes with the number of bytes and number of cycles used by each instruction.

Table 10 lists all instructions in hexadecimal code order.

**Table 3: Data Addressing Mnemonics**

| | |
|---|---|
| Rn | Working register R0-R7 |
| direct | 256 internal RAM locations, any Special Function Registers |
| @Ri | Indirect internal or external RAM location addressed by register R0 or R1 |
| #data | 8-bit constant included in instruction |
| #data 16 | 16-bit constant included as bytes 2 and 3 of instruction |
| bit | 256 software flags, any bit-addressable I/O pin, control or status bit |
| A | Accumulator |

**Table 4: Program Addressing Mnemonics**

| | |
|---|---|
| addr16 | Destination address for LCALL and LJMP may be anywhere within the 64-Kbyte of program memory address space. |
| addr11 | Destination address for ACALL and AJMP will be within the same 2-Kbytepage of program memory as the first byte of the following instruction. |
| rel | SJMP and all conditional jumps include an 8-bit offset byte. Range is +127/-128 bytes relative to the first byte of the following instruction |

**Table 5: Arithmetic Instructions**

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| ADD A,Rn | Add register to accumulator | 28-2F | 1 | 1 |
| ADD A,direct | Add direct byte to accumulator | 25 | 2 | 2 |
| ADD A,@Ri | Add indirect RAM to accumulator | 26-27 | 1 | 2 |
| ADD A,#data | Add immediate data to accumulator | 24 | 2 | 2 |
| ADDC A,Rn | Add register to accumulator with carry flag | 38-3F | 1 | 1 |
| ADDC A,direct | Add direct byte to A with carry flag | 35 | 2 | 2 |
| ADDC A,@Ri | Add indirect RAM to A with carry flag | 36-37 | 1 | 2 |
| ADDC A,#data | Add immediate data to A with carry flag | 34 | 2 | 2 |
| SUBB A,Rn | Subtract register from A with borrow | 98-9F | 1 | 1 |
| SUBB A,direct | Subtract direct byte from A with borrow | 95 | 2 | 2 |
| SUBB A,@Ri | Subtract indirect RAM from A with borrow | 96-97 | 1 | 2 |
| SUBB A,#data | Subtract immediate data from A with borrow | 94 | 2 | 2 |
| INC A | Increment accumulator | 04 | 1 | 1 |
| INC Rn | Increment register | 08-0F | 1 | 2 |
| INC direct | Increment direct byte | 05 | 2 | 3 |
| INC @Ri | Increment indirect RAM | 06-07 | 1 | 3 |
| INC DPTR | Increment data pointer | A3 | 1 | 1 |
| DEC A | Decrement accumulator | 14 | 1 | 1 |
| DEC Rn | Decrement register | 18-1F | 1 | 2 |
| DEC direct | Decrement direct byte | 15 | 2 | 3 |
| DEC @Ri | Decrement indirect RAM | 16-17 | 1 | 3 |
| MUL AB | Multiply A and B | A4 | 1 | 5 |
| DIV | Divide A by B | 84 | 1 | 5 |
| DA A | Decimal adjust accumulator | D4 | 1 | 1 |

**Table 6: Logic Instructions**

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| ANL A,Rn | AND register to accumulator | 58-5F | 1 | 1 |
| ANL A,direct | AND direct byte to accumulator | 55 | 2 | 2 |
| ANL A,@Ri | AND indirect RAM to accumulator | 56-57 | 1 | 2 |
| ANL A,#data | AND immediate data to accumulator | 54 | 2 | 2 |
| ANL direct,A | AND accumulator to direct byte | 52 | 2 | 3 |
| ANL direct,#data | AND immediate data to direct byte | 53 | 3 | 4 |
| ORL A,Rn | OR register to accumulator | 48-4F | 1 | 1 |
| ORL A,direct | OR direct byte to accumulator | 45 | 2 | 2 |
| ORL A,@Ri | OR indirect RAM to accumulator | 46-47 | 1 | 2 |
| ORL A,#data | OR immediate data to accumulator | 44 | 2 | 2 |
| ORL direct,A | OR accumulator to direct byte | 42 | 2 | 3 |
| ORL direct,#data | OR immediate data to direct byte | 43 | 3 | 4 |
| XRL A,Rn | Exclusive OR register to accumulator | 68-6F | 1 | 1 |
| XRL A,direct | Exclusive OR direct byte to accumulator | 65 | 2 | 2 |
| XRL A,@Ri | Exclusive OR indirect RAM to accumulator | 66-67 | 1 | 2 |
| XRL A,#data | Exclusive OR immediate data to accumulator | 64 | 2 | 2 |
| XRL direct,A | Exclusive OR accumulator to direct byte | 62 | 2 | 3 |
| XRL direct,#data | Exclusive OR immediate data to direct byte | 63 | 3 | 4 |
| CLR A | Clear accumulator | E4 | 1 | 1 |
| CPL A | Complement accumulator | F4 | 1 | 1 |
| RL A | Rotate accumulator left | 23 | 1 | 1 |
| RLC A | Rotate accumulator left through carry | 33 | 1 | 1 |
| RR A | Rotate accumulator right | 03 | 1 | 1 |
| RRC A | Rotate accumulator right through carry | 13 | 1 | 1 |
| SWAP A | Swap nibbles within the accumulator | C4 | 1 | 1 |

**Table 7: Data Transfer Instructions**

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| MOV A,Rn | Move register to accumulator | E8-EF | 1 | 1 |
| MOV A,direct | Move direct byte to accumulator | E5 | 2 | 2 |
| MOV A,@Ri | Move indirect RAM to accumulator | E6-E7 | 1 | 2 |
| MOV A,#data | Move immediate data to accumulator | 74 | 2 | 2 |
| MOV Rn,A | Move accumulator to register | F8-FF | 1 | 2 |
| MOV Rn,direct | Move direct byte to register | A8-AF | 2 | 4 |
| MOV Rn,#data | Move immediate data to register | 78-7F | 2 | 2 |
| MOV direct,A | Move accumulator to direct byte | F5 | 2 | 3 |
| MOV direct,Rn | Move register to direct byte | 88-8F | 2 | 3 |
| MOV direct1,direct2 | Move direct byte to direct byte | 85 | 3 | 4 |
| MOV direct,@Ri | Move indirect RAM to direct byte | 86-87 | 2 | 4 |
| MOV direct,#data | Move immediate data to direct byte | 75 | 3 | 3 |
| MOV @Ri,A | Move accumulator to indirect RAM | F6-F7 | 1 | 3 |
| MOV @Ri,direct | Move direct byte to indirect RAM | A6-A7 | 2 | 5 |
| MOV @Ri,#data | Move immediate data to indirect RAM | 76-77 | 2 | 3 |
| MOV DPTR,#data16 | Load data pointer with a 16-bit constant | 90 | 3 | 3 |
| MOVC A,@A+DPTR | Move code byte relative to DPTR to accumulator | 93 | 1 | 3 |
| MOVC A,@A+PC | Move code byte relative to PC to accumulator | 83 | 1 | 3 |
| MOVX A,@Ri | Move external RAM (8-bit address) to A | E2-E3 | 1 | 3-10 |
| MOVX A,@DPTR | Move external RAM (16-bit address) to A | E0 | 1 | 3-10 |
| MOVX @Ri,A | Move A to external RAM (8-bit address) | F2-F3 | 1 | 4-11 |
| MOVX @DPTR,A | Move A to external RAM (16-bit address) | F0 | 1 | 4-11 |
| PUSH direct | Push direct byte onto stack | C0 | 2 | 4 |
| POP direct | Pop direct byte from stack | D0 | 2 | 3 |
| XCH A,Rn | Exchange register with accumulator | C8-CF | 1 | 2 |
| XCH A,direct | Exchange direct byte with accumulator | C5 | 2 | 3 |
| XCH A,@Ri | Exchange indirect RAM with accumulator | C6-C7 | 1 | 3 |
| XCHD A,@Ri | Exchange low-order nibble indirect RAM with A | D6-D7 | 1 | 3 |

**Table 8: Program Branch Instructions**

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| ACALL addr11 | Absolute subroutine call | xxx11 | 2 | 6 |
| LCALL addr16 | Long subroutine call | 12 | 3 | 6 |
| RET | from subroutine | 22 | 1 | 4 |
| RETI | from interrupt | 32 | 1 | 4 |
| AJMP addr11 | Absolute jump | xxx01 | 2 | 3 |
| LJMP addr16 | Long jump | 02 | 3 | 4 |
| SJMP rel | Short jump (relative address) | 80 | 2 | 3 |
| JMP @A+DPTR | Jump indirect relative to the DPTR | 73 | 1 | 2 |
| JZ rel | Jump if accumulator is zero | 60 | 2 | 3 |
| JNZ rel | Jump if accumulator is not zero | 70 | 2 | 3 |
| JC rel | Jump if carry flag is set | 40 | 2 | 3 |
| JNC | Jump if carry flag is not set | 50 | 2 | 3 |
| JB bit,rel | Jump if direct bit is set | 20 | 3 | 4 |
| JNB bit,rel | Jump if direct bit is not set | 30 | 3 | 4 |
| JBC bit,direct rel | Jump if direct bit is set and clear bit | 10 | 3 | 4 |
| CJNE A,direct rel | Compare direct byte to A and jump if not equal | B5 | 3 | 4 |
| CJNE A,#data rel | Compare immediate to A and jump if not equal | B4 | 3 | 4 |
| CJNE Rn,#data rel | Compare immed. to reg. and jump if not equal | B8-BF | 3 | 4 |
| CJNE @Ri,#data rel | Compare immed. to ind. and jump if not equal | B6-B7 | 3 | 4 |
| DJNZ Rn,rel | Decrement register and jump if not zero | D8-DF | 2 | 3 |
| DJNZ direct,rel | Decrement direct byte and jump if not zero | D5 | 3 | 4 |
| NOP | No operation | 00 | 1 | 1 |

**Table 9: Boolean Manipulation Instructions**

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| CLR C | Clear carry flag | C3 | 1 | 1 |
| CLR bit | Clear direct bit | C2 | 2 | 3 |
| SETB C | Set carry flag | D3 | 1 | 1 |
| SETB bit | Set direct bit | D2 | 2 | 3 |
| CPL C | Complement carry flag | B3 | 1 | 1 |
| CPL bit | Complement direct bit | B2 | 2 | 3 |
| ANL C,bit | AND direct bit to carry flag | 82 | 2 | 2 |
| ANL C,/bit | AND complement of direct bit to carry | B0 | 2 | 2 |
| ORL C,bit | OR direct bit to carry flag | 72 | 2 | 2 |
| ORL C,/bit | OR complement of direct bit to carry | A0 | 2 | 2 |
| MOV C,bit | Move direct bit to carry flag | A2 | 2 | 2 |
| MOV bit,C | Move carry flag to direct bit | 92 | 2 | 3 |

**Table 10: Instruction Set in Hexadecimal Order**

| Opcode | Mnemonic | Opcode | Mnemonic | Opcode | Mnemonic | Opcode | Mnemonic |
|---|---|---|---|---|---|---|---|
| 00 H | NOP | 10 H | JBC bit,rel | 20 H | JB bit,rel | 30 H | JNB bit,rel |
| 01 H | AJMP addr11 | 11 H | ACALL addr11 | 21 H | AJMP addr11 | 31 H | ACALL addr11 |
| 02 H | LJMP addr16 | 12 H | LCALL addr16 | 22 H | RET | 32 H | RETI |
| 03 H | RR A | 13 H | RRC A | 23 H | RL A | 33 H | RLC A |
| 04 H | INC A | 14 H | DEC A | 24 H | ADD A,#data | 34 H | ADDC A,#data |
| 05 H | INC direct | 15 H | DEC direct | 25 H | ADD A,direct | 35 H | ADDC A,direct |
| 06 H | INC @R0 | 16 H | DEC @R0 | 26 H | ADD A,@R0 | 36 H | ADDC A,@R0 |
| 07 H | INC @R1 | 17 H | DEC @R1 | 27 H | ADD A,@R1 | 37 H | ADDC A,@R1 |
| 08 H | INC R0 | 18 H | DEC R0 | 28 H | ADD A,R0 | 38 H | ADDC A,R0 |
| 09 H | INC R1 | 19 H | DEC R1 | 29 H | ADD A,R1 | 39 H | ADDC A,R1 |
| 0A H | INC R2 | 1A H | DEC R2 | 2A H | ADD A,R2 | 3A H | ADDC A,R2 |
| 0B H | INC R3 | 1B H | DEC R3 | 2B H | ADD A,R3 | 3B H | ADDC A,R3 |
| 0C H | INC R4 | 1C H | DEC R4 | 2C H | ADD A,R4 | 3C H | ADDC A,R4 |
| 0D H | INC R5 | 1D H | DEC R5 | 2D H | ADD A,R5 | 3D H | ADDC A,R5 |
| 0E H | INC R6 | 1E H | DEC R6 | 2E H | ADD A,R6 | 3E H | ADDC A,R6 |
| 0F H | INC R7 | 1F H | DEC R7 | 2F H | ADD A,R7 | 3F H | ADDC A,R7 |

| Opcode | Mnemonic | Opcode | Mnemonic | Opcode | Mnemonic |
|---|---|---|---|---|---|
| 40 H | JC rel | 50 H | JNC rel | 60 H | JZ rel |
| 41 H | AJMP addr11 | 51 H | ACALL addr11 | 61 H | AJMP addr11 |
| 42 H | ORL direct,A | 52 H | ANL direct,A | 62 H | XRL direct,A |
| 43 H | ORL direct,#data | 53 H | ANL direct,#data | 63 H | XRL direct,#data |
| 44 H | ORL A,#data | 54 H | ANL A,#data | 64 H | XRL A,#data |
| 45 H | ORL A,direct | 55 H | ANL A,direct | 65 H | XRL A,direct |
| 46 H | ORL A,@R0 | 56 H | ANL A,@R0 | 66 H | XRL A,@R0 |
| 47 H | ORL A,@R1 | 57 H | ANL A,@R1 | 67 H | XRL A,@R1 |
| 48 H | ORL A,R0 | 58 H | ANL A,R0 | 68 H | XRL A,R0 |
| 49 H | ORL A,R1 | 59 H | ANL A,R1 | 69 H | XRL A,R1 |
| 4A H | ORL A,R2 | 5A H | ANL A,R2 | 6A H | XRL A,R2 |
| 4B H | ORL A,R3 | 5B H | ANL A,R3 | 6B H | XRL A,R3 |
| 4C H | ORL A,R4 | 5C H | ANL A,R4 | 6C H | XRL A,R4 |
| 4D H | ORL A,R5 | 5D H | ANL A,R5 | 6D H | XRL A,R5 |
| 4E H | ORL A,R6 | 5E H | ANL A,R6 | 6E H | XRL A,R6 |
| 4F H | ORL A,R7 | 5F H | ANL A,R7 | 6F H | XRL A,R7 |

| Opcode | Mnemonic | Opcode | Mnemonic | Opcode | Mnemonic |
|---|---|---|---|---|---|
| 70 H | JNZ rel | 80 H | SJMP rel | 90 H | MOV DPTR,#data16 |
| 71 H | ACALL addr11 | 81 H | AJMP addr11 | 91 H | ACALL addr11 |
| 72 H | ORL C,direct | 82 H | ANL C,bit | 92 H | MOV bit,C |
| 73 H | JMP @A+DPTR | 83 H | MOVC A,@A+PC | 93 H | MOVC A,@A+DPTR |
| 74 H | MOV A,#data | 84 H | DIV AB | 94 H | SUBB A,#data |
| 75 H | MOV direct,#data | 85 H | MOV direct,direct | 95 H | SUBB A,direct |
| 76 H | MOV @R0,#data | 86 H | MOV direct,@R0 | 96 H | SUBB A,@R0 |
| 77 H | MOV @R1,#data | 87 H | MOV direct,@R1 | 97 H | SUBB A,@R1 |
| 78 H | MOV R0.#data | 88 H | MOV direct,R0 | 98 H | SUBB A,R0 |
| 79 H | MOV R1.#data | 89 H | MOV direct,R1 | 99 H | SUBB A,R1 |
| 7A H | MOV R2.#data | 8A H | MOV direct,R2 | 9A H | SUBB A,R2 |
| 7B H | MOV R3.#data | 8B H | MOV direct,R3 | 9B H | SUBB A,R3 |
| 7C H | MOV R4.#data | 8C H | MOV direct,R4 | 9C H | SUBB A,R4 |
| 7D H | MOV R5.#data | 8D H | MOV direct,R5 | 9D H | SUBB A,R5 |
| 7E H | MOV R6.#data | 8E H | MOV direct,R6 | 9E H | SUBB A,R6 |
| 7F H | MOV R7.#data | 8F H | MOV direct,R7 | 9F H | SUBB A,R7 |

| Opcode | Mnemonic | Opcode | Mnemonic | Opcode | Mnemonic |
|---|---|---|---|---|---|
| A0 H | ORL C,bit | B0 H | ANL C,bit | C0 H | PUSH direct |
| A1 H | AJMP addr11 | B1 H | ACALL addr11 | C1 H | AJMP addr11 |
| A2 H | MOV C,bit | B2 H | CPL bit | C2 H | CLR bit |
| A3 H | INC DPTR | B3 H | CPL C | C3 H | CLR C |
| A4 H | MUL AB | B4 H | CJNE A,#data,rel | C4 H | SWAP A |
| A5 H | - | B5 H | CJNE A,direct,rel | C5 H | XCH A,direct |
| A6 H | MOV @R0,direct | B6 H | CJNE @R0,#data,rel | C6 H | XCH A,@R0 |
| A7 H | MOV @R1,direct | B7 H | CJNE @R1,#data,rel | C7 H | XCH A,@R1 |
| A8 H | MOV R0,direct | B8 H | CJNE R0,#data,rel | C8 H | XCH A,R0 |
| A9 H | MOV R1,direct | B9 H | CJNE R1,#data,rel | C9 H | XCH A,R1 |
| AA H | MOV R2,direct | BA H | CJNE R2,#data,rel | CA H | XCH A,R2 |
| AB H | MOV R3,direct | BB H | CJNE R3,#data,rel | CB H | XCH A,R3 |
| AC H | MOV R4,direct | BC H | CJNE R4,#data,rel | CC H | XCH A,R4 |
| AD H | MOV R5,direct | BD H | CJNE R5,#data,rel | CD H | XCH A,R5 |
| AE H | MOV R6,direct | BE H | CJNE R6,#data,rel | CE H | XCH A,R6 |
| AF H | MOV R7,direct | BF H | CJNE R7,#data,rel | CF H | XCH A,R7 |

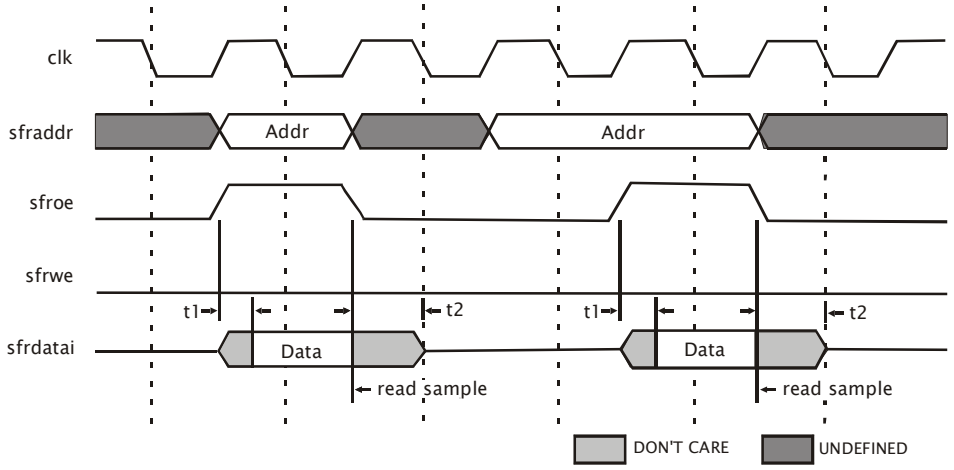
 

| Opcode | Mnemonic | Opcode | Mnemonic | Opcode | Mnemonic |
|---|---|---|---|---|---|
| D0 H | POP direct | E0 H | MOVX A,@DPTR | F0 H | MOVX @DPTR,A |
| D1 H | ACALL addr11 | E1 H | AJMP addr11 | F1 H | ACALL addr11 |
| D2 H | SETB bit | E2 H | MOVX A,@R0 | F2 H | MOVX @R0,A |
| D3 H | SETB C | E3 H | MOVX A,@R1 | F3 H | MOVX @R1,A |
| D4 H | DA A | E4 H | CLR A | F4 H | CPL A |
| D5 H | DJNZ direct,rel | E5 H | MOV A,direct | F5 H | MOV direct,A |
| D6 H | XCHD A,@R0 | E6 H | MOV A,@R0 | F6 H | MOV @R0,A |
| D7 H | XCHD A,@R1 | E7 H | MOV A,@R1 | F7 H | MOV @R1,A |
| D8 H | DJNZ R0,rel | E8 H | MOV A,R0 | F8 H | MOV R0,A |
| D9 H | DJNZ R1,rel | E9 H | MOV A,R1 | F9 H | MOV R1,A |
| DA H | DJNZ R2,rel | EA H | MOV A,R2 | FA H | MOV R2,A |
| DB H | DJNZ R3,rel | EB H | MOV A,R3 | FB H | MOV R3,A |
| DC H | DJNZ R4,rel | EC H | MOV A,R4 | FC H | MOV R4,A |
| DD H | DJNZ R5,rel | ED H | MOV A,R5 | FD H | MOV R5,A |
| DE H | DJNZ R6,rel | EE H | MOV A,R6 | FE H | MOV R6,A |
| DF H | DJNZ R7,rel | EF H | MOV A,R7 | FF H | MOV R7,A |

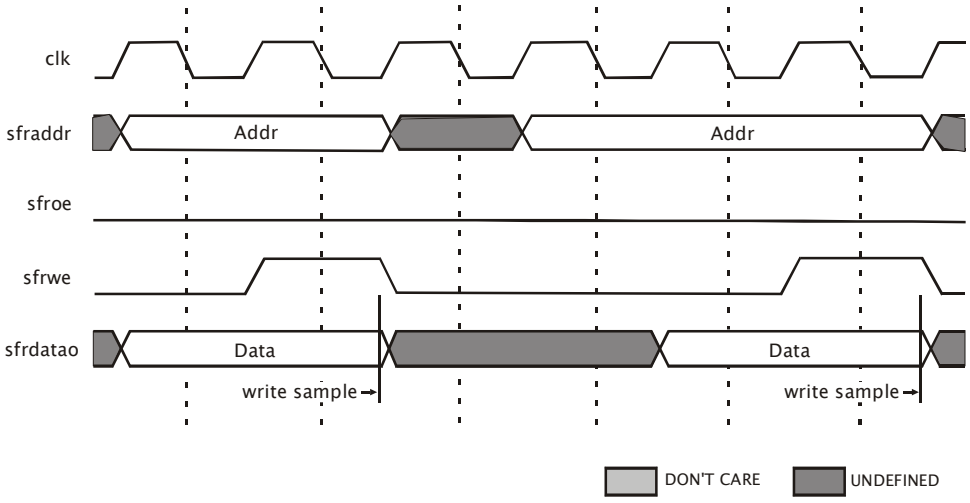
## 13. External SFR Timing

**Figure 5 – External Use of Special Function Register Bus (read)**




Notes: For both t1 and t2, maximum = time period of clock signal.
Addr = address of special function register
Data = data returned from register
Read Sample = point at which data is read from the bus

**Figure 6 – External Use of Special Function Register Bus (write)**




Notes: Addr = address of special function register
Data = data written into register
Write Sample = point at which data is written from bus to register

## 14. Hardware Overview

The structure of the TSCR8051 consists of the following units:

- Core Engine:
  Arithmetic/Logic (ALU)
  SRAM Memory Control
  Scratch Pad and Special Function Register (SFR) Control
- Multiplication-Division
- Clock Control
- Timers: 0, 1, 2, Capture-Compare, and Watchdog
- Serial Ports 0 & 1
- Interrupt Service
- Floating-Point
- Extended Computing
- SPI Memory Loader
- Reset
- Power Management

## 15. Core Engine

The core engine of the TSCR8051 contains:

- Arithmetic and Logic (ALU)
- SRAM Memory Control
- Scratch Pad and SFR (Special Function Register) Control
- I/O Ports

The engine fetches instructions from program memory and executes them, fetching and storing data to/from data memory and the ports as needed. The core engine components use the following registers (described in section 10, starting on page 29):

| NAME | MNEMONIC | LOCATION (HEX) | NOTES |
|---|---|---|---|
| Program Counter | PC | n/a | 2 bytes; initialized to 00H. |
| Program Status Word | PSW | D0 | 1 byte |
| Stack Pointer | SP | 81 | 1 byte |
| Data Pointer | DPL & DPH or DPL1 & DPH1 | 82, 83 or 84, 85 | 2 bytes |
| Accumulator | ACC or A | E0 | 1 byte |
| B Register | B | F0 | 1 byte |
| Ports | P0, P1, P2, P3 | 80, 90, A0, B0 | 1 byte each |

## 16. Multiplication / Division Unit (MDU)

This on-chip arithmetic unit provides 32-bit division and 16-bit multiplication, shift, and normalize features. All operations are unsigned integer operations.

The MDU uses seven registers which are memory mapped as special function registers. This unit operates concurrently with, and independent of, the CPU. Any MDU calculation overwrites its operands.

The MDU registers are:

| NAME | MNEMONIC | LOCATION (HEX) | NOTES |
|---|---|---|---|
| Multiplication Division 0-5 (Operands and Results) | MD0, MD1, MD2, MD3, MD4, MD5 | E9 through EE | See descriptions in section 10, starting on page 29. |
| Arithmetic Control | ARCON | EF | |

## 16.1. MDU Operation and Timing

Operation of the MDU occurs in three phases:

**Phase 1: Load the MDx registers**

The type of calculation to be performed is determined by the order in which the MDx registers are written. For the operands to write to each register, see the tables in the Special Function Register Descriptions (page 29 and following).

**Table 11: MDU Register Write Sequence**

| OPERATION | 32BIT/16BIT | 16BIT/16BIT | 16BIT X 16BIT | SHIFT/NORMALIZE |
|---|---|---|---|---|
| first write | MD0 | MD0 | MD0 | MD0 |
| | MD1 | MD1 | MD4 | MD1 |
| | MD2 | MD4 | MD1 | MD2 |
| | MD3 | -- | -- | MD3 |
| | MD4 | -- | -- | -- |
| last write | MD5 | MD5 | MD5 | ARCON |

In all cases, a write to MD0 is the first transfer. The remaining writes must be performed in the order shown. The last write triggers the selected operation.

**Phase 2: Execute the calculation**

During execution, the MDU works on its own, in parallel with the CPU.

**Table 12: MDU Execution Times**

| OPERATION | MAX. TIME (IN TCLK) | MIN. TIME (IN TCLK) |
|---|---|---|
| Division 32bit/16bit | 50 (division by 1) | 19 (divider > 7FFFh) |
| Division 16bit/16bit | 34 (division by 1) | 3 (divider > 7FFFh) |
| Multiplication | 17 (result) +1 (set MDOV flag) | 17 (result) +1 (set MDOV flag) |
| Shift | 33 (sc = 1Fh) | 3 (sc = 01h) |
| Normalize | 34 (sc < –1Fh) | 4 (sc < –01h) |

**Phase 3: Read results from the MDx registers**

For the values to read from each register, see the descriptions in section 10, starting on page 29.

**Table 13: MDU Register Read Sequence**

| OPERATION | 32BIT/16BIT | 16BIT/16BIT | 16BIT X 16BIT | SHIFT/NORMALIZE |
|---|---|---|---|---|
| first read | MD0 | MD0 | MD0 | MD0 |
| | MD1 | MD1 | MD1 | MD1 |
| | MD2 | MD4 | MD2 | MD2 |
| | MD3 | -- | -- | -- |
| | MD4 | -- | -- | -- |
| last read | MD5 | MD5 | MD3 | MD3 |

**Shifting**

The SLR bit (ARCON.5) specifies the shift direction; ARCON.4 to ARCON.0 specify the shift count (which must not be 0). During shift, zeroes come into the left end of register MD0 or the right end of register MD3.

Normalizing

This performs repeated shift left operations to remove all leading zeroes of the integer variable stored in registers MD0 to MD3. Normalization is complete when the MSB (most significant bit) of the MD3 register contains a '1'. After normalizing, bits ARCON.4 (MSB) to ARCON.0 (LSB) contain the number of shift left operations that were done.

MDEF Flag

The MDEF error flag (read-only) indicates an improperly performed operation – that is, an arithmetic operation that has been restarted or interrupted by a new operation.

The error flag mechanism is automatically enabled with the first write to MD0 and disabled with the final read instruction from MD3 (multiplication, shift, or normalize) or MD5 (division) in phase three.

The error flag is set when:

a) Phase two is in process and a write access occurs to any MDx register (calculation restarted/interrupted)

b) Phase one or two is in progress and a read access occurs to any MDx register (does not interrupt calculation)

The read-only error flag is reset when Phase Two completes successfully and a read access occurs to the MDx registers.

**MDOV Flag**

MDOV is a read-only overflow flag. It is set when any of these occurs:

a) Division by zero

b) Multiplication with a result greater than 0000 FFFFh

c) MD3.7 (most significant bit) is '1' when normalization begins

The read-only overflow flag is reset when Phase One starts (initial write to MD0).

## 17. Timers

There are three 16-bit timers: Timer 0, Timer 1 and Timer 2. All can be configured for counter or timer operations. Timer 1 can also be used to generate the baud rate for serial port 0; Timer 2 has a Capture/Compare Unit (CCU) and several additional features.

### 17.1. Timers 0 and 1

**Registers**

The 16-bit value for Timer 0 is held in registers TL0 and TH0; Timer 1 is in TL1 and TH1. The behavior of these timers is controlled by various bits in registers TCON and TMOD (pages 11 and 11).

**Timer vs. Counter**

Each timer can be used either as a counter or as a timer by configuring bit C/Tx in TMOD.

- For a **timer,** the value is incremented once every 12 clock cycles.

- For a **counter**, the value increments when a falling edge is observed at input pin T0 (Timer 0) or T1 (Timer 1) of PORT3. (PORT3 must be properly configured.) It takes 2 clock cycles to recognize a 1-to-0 event. There are no restrictions on the duty cycle; however, to ensure proper recognition of the state, each input signal should be stable for at least 1 clock cycle.

**General Operations**

Timer x is started or halted by setting bit TRx in TCON to 1 or 0, respectively.

External control can be implemented by setting bit GATEx in TMOD. If GATEx is set to 1, Timer x runs only when TRx is 1 and the INTx pin of PORT3 is asserted (low). (INTx must be properly configured.)

When Timer x overflows, bit TFx in TCON is set. Bit ETx in register IEN0 (page 19) can configure TFx to cause an interrupt; in this case, TFx is cleared when the interrupt is processed. TFx can also be cleared by software.

Timer 1 can be used to generate the baud rate for serial port 0; see page 52.

**Operating Modes**

Each counter/timer can operate in the following modes, as configured by bits M0-x and M1-x in the TMOD register:

Mode 0: 13-bit counter/timer. The upper three bits of the TLx register are undetermined and should not be used.

Mode 1: 16-bit counter/timer.

Mode 2: 8-bit counter/timer with auto-reload. The reload value is held in THx while TLx acts as the counter/timer. When THx overflows, the value from THx is copied into TLx.

Mode 3: This mode is different for Timer 0 and Timer 1.
**Timer 0:** Acts as two independent 8-bit counters/timers. TL0 behaves according to the Timer 0 control bits, but TH0 is locked into "timer" mode and takes over the use of bits TR1 and TF1. Timer 1 retains the use of all its other pins and bits (T1, INT1, GATE1, C/T1, M0-1, and M1-1). Timer 1 thus retains the ability to generate the serial port 0 baud rate and perform other functions, but it can no longer generate an interrupt. When Timer 0 is in this mode, Timer 1 can be halted and started by switching in and out of this same mode.
**Timer 1:** Halts; holds the current value; does not increment.

**Figure 7 – Timer/Counter 1 in Mode 0**

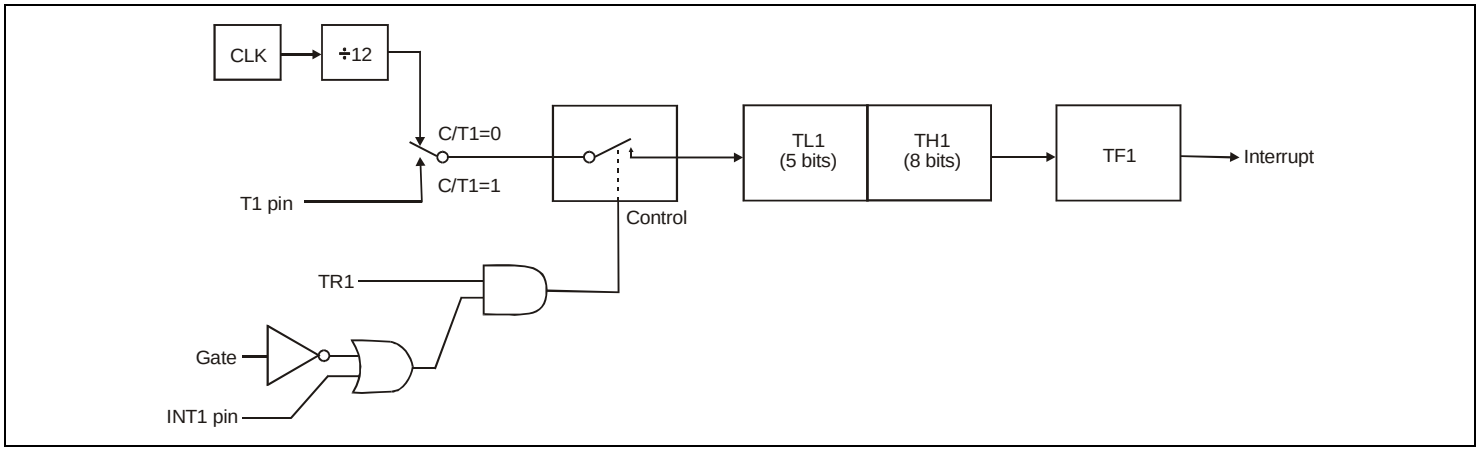


**Figure 8 – Timer/Counter 1 in Mode 2**

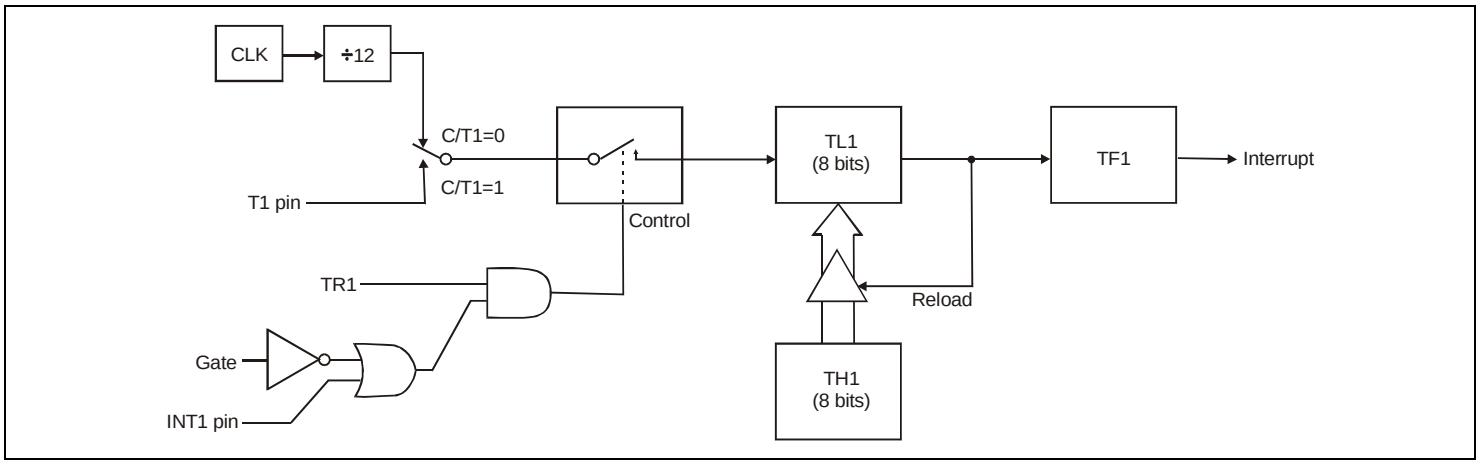
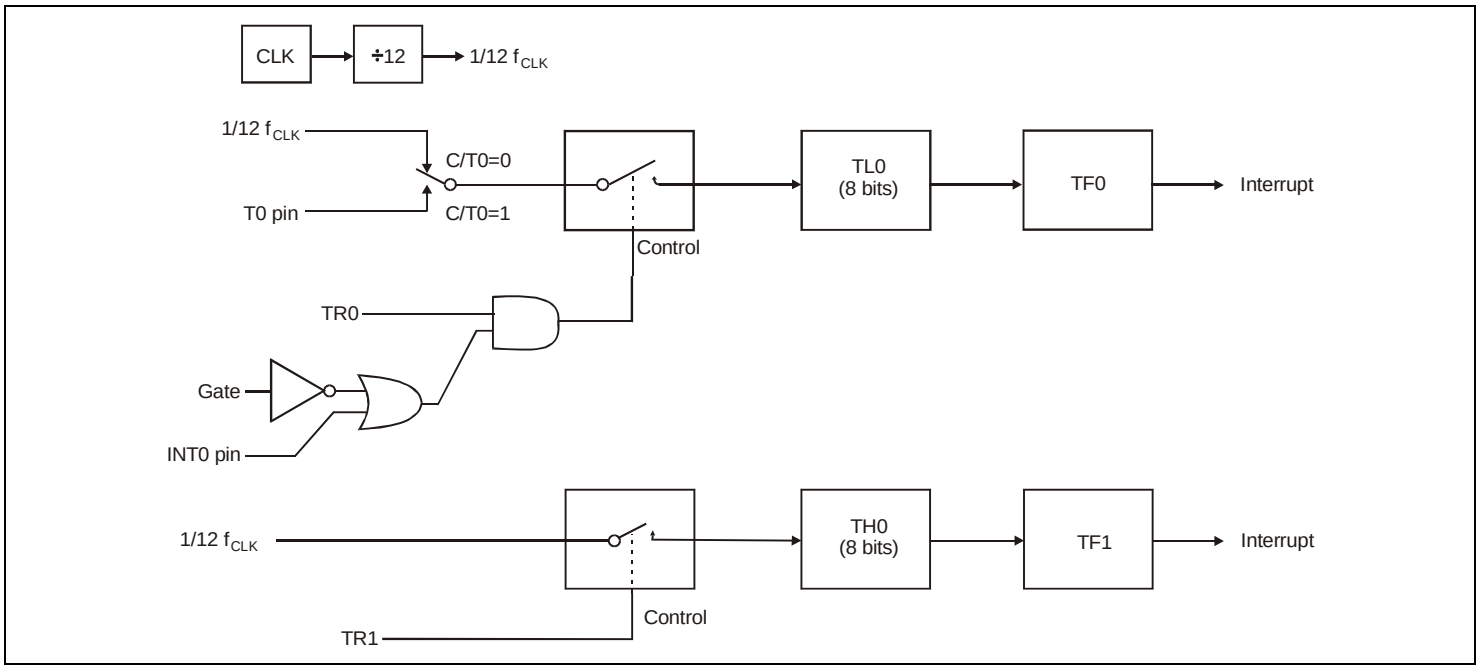



**Figure 9 – Timer/Counter 0 in Mode 3**




## 17.2. Timer 2 and Capture/Compare Unit

### Registers

The 16-bit value for Timer 2 is stored in registers TL2 and TH2. It is controlled by register T2CON. Timer 2 includes a Capture/Compare Unit (CCU) which is enabled and configured by register CCEN. The CCU uses three 16-bit Capture/Compare values (6 registers) and one 16-bit Capture/Reload/Compare value (2 registers), as listed in the table below. For descriptions of the CCU registers, see pages 24 through 27.

| ADDRESS | NAME | USAGE |
|---|---|---|
| 90h | P1 (PORT1) | Bits 0 through 3 can be configured for use by the CCU. |
| C1h | CCEN | Configures and enables all Compare/Capture modes. |
| C2h & C3h | CC1 (CCL1 and CCH1) | Holds 16-bit Capture 1 value or Compare 1 value. |
| C4h & C5h | CC2 (CCL2 and CCH2) | Holds 16-bit Capture 2 value or Compare 2 value. |
| C6h & C7h | CC3 (CCL3 and CCH3) | Holds 16-bit Capture 3 value or Compare 3 value. |
| C8h | T2CON | Configures & enables Timer 2 |
| CAh & CBh | CRC (CRCL and CRCH) | Holds 16-bit reload value or Capture 0 value or Compare 0 value. |
| CCh & CDh | T2 (TL2 and TH2) | Contains 16-bit Timer 2 value |

**Signals**

The use of Timer 2 and the Capture/Compare Unit can involve several signals, as listed in the table below:

| SIGNAL | LOCATION | USAGE |
|---|---|---|
| T2EX | PORT1[5] | External trigger for reload. |
| T2 | PORT1[7] | External trigger for counter or gate for timer. |
| CC0 – CC3 | PORT1[0 – 3] | External trigger for capture or signal generated by compare. |
| EXF2 | IRCON register | Request interrupt on external reload. |
| TF2 | IRCON register | Request interrupt on overflow. |
| IEX3 – IEX6 | IRCON register | Request interrupt on capture or compare. |

**Start/Stop and Timer/Counter**

Timer 2 can be halted or operated as either a timer or a counter by configuring bits T2I0 and T2I1 in T2CON.

- For a **timer**, the value is incremented according to the Prescaler Select bit, T2PS. T2PS = 0 increments every 12 clock cycles; T2PS = 1 increments every 24 clock cycles. Input pin T2 of PORT1 may be configured as a gate to start and stop the timer.
- For a **counter**, the value is incremented one cycle after a falling edge is observed at input pin T2 of PORT1. (PORT1 must be properly configured.) It takes 2 clock cycles to recognize a 1-to-0 event. There are no restrictions on the duty cycle; however, to ensure proper recognition of the state, each input signal should be stable for at least 1 clock cycle.

**Overflow**

Whenever Timer 2 overflows, bit TF2 in the IRCON register is set. This can be configured to request an interrupt. Bit TF2 should be cleared by the interrupt routine.

**Reload**

Timer 2 can operate with no reload or in either of two reload modes, as configured by bits T2R0 and T2R1 in T2CON. In either mode, a reload copies the values from registers CRCL and CRCH into registers TL2 and TH2.

**Mode 0:** Reload is triggered by a Timer 2 overflow.

**Mode 1:** Reload is triggered by a negative transition on pin T2EX of PORT1. This can be configured to request an interrupt via flag EXF2 in register IRCON.

**Capture**

The 16-bit value from Timer 2 can be saved in any of the four pairs of capture/compare registers, as configured in the CCEN register. There are two capture modes available for each pair of registers. In either mode, a capture writes the values from TL2 and TH2 into CRCL and CRCH or into CCxL and CCxH.

**Mode 0:** Capture is triggered by a transition at pin CCx of PORT1. For CC0, the trigger transition is configured by bit I3FR of T2CON; the other three pins always trigger on a rising edge. If configured to do so, the CCx transition will request an interrupt by setting flag IEXx in IRCON.

**Mode 1:** Capture is triggered by any write into the lower register of a capture pair. The value written is irrelevant. No interrupt is requested.

**Compare**

As the Timer 2 value changes, it can be automatically compared to the value in any of the four pairs of capture/compare registers, as configured by CCEN. If the values are equal, an appropriate signal is sent to pin CCx of PORT1. (If configured to do so, CCx transitions may request an interrupt by setting flag IEXx in IRCON.) The signal is determined by the compare mode, as specified by bit T2CM in T2CON:

**Mode 0**: On an equal compare, pin CCx changes from low to high. In this mode, writing to PORT1 will have no effect, because the input line from the internal bus and the write-to-latch line are disconnected.

**Mode 1**: Before the compare, software writes a value to bit x of register P1, but the value is not transmitted to PORT1. On an equal compare, the bit x value is transmitted to pin CCx. Timer 2 overflow has no effect in this mode; software controls both transitions of pin CCx.

**Interrupts**

Timer 2 and the CCU can be configured to request interrupts in any of several conditions:

Any overflow – TF2

Reload Mode 1 – EXF2

Capture Mode 0 – IEXx

Any Compare – IEXx

For more about interrupts, see section 19 on page 54.

**Figure 10 – Timer 2 as Gated Timer: Prescaler Select = 1, Reload Mode = 1**

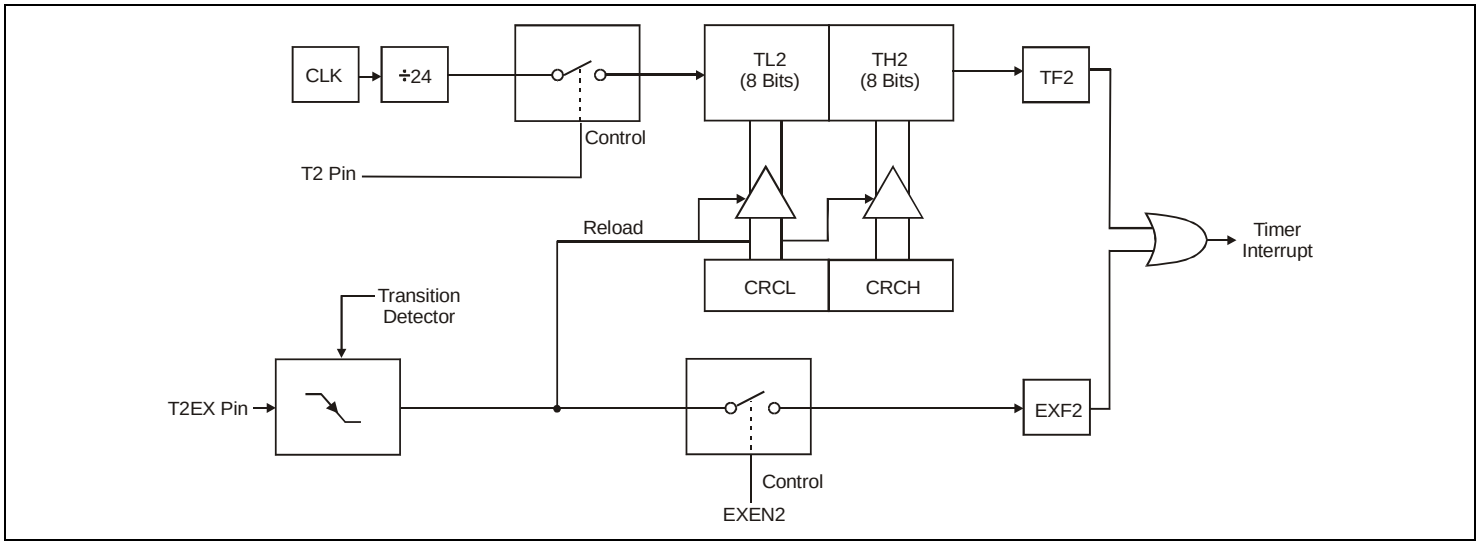


**Figure 11 – Timer 2 as Counter: Capture Mode = 0 (using CC3)**

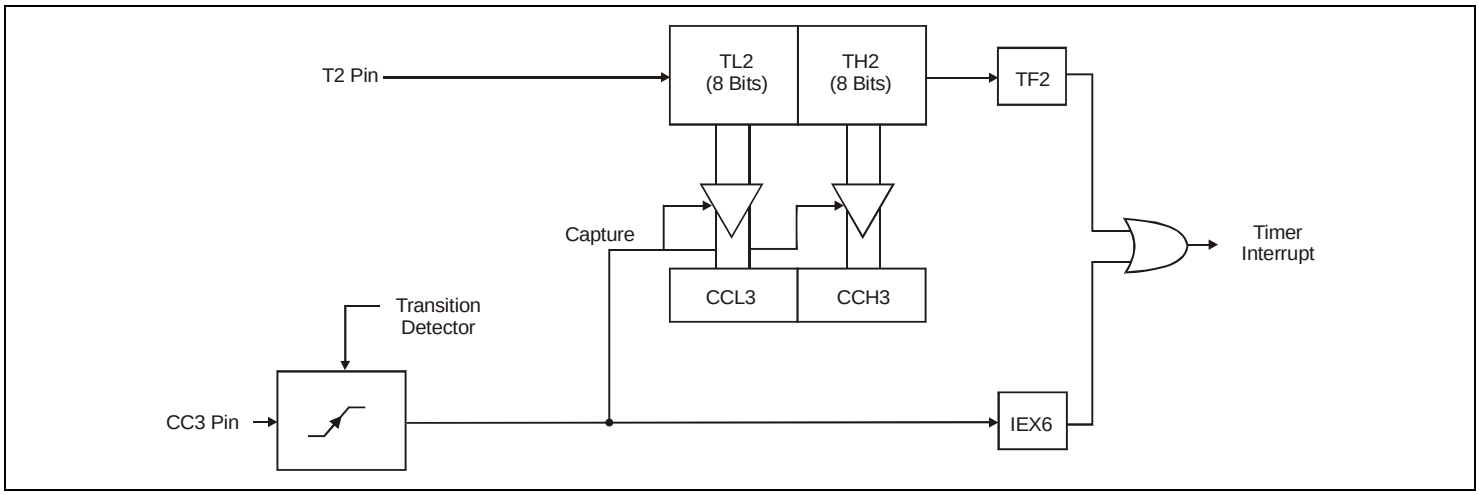
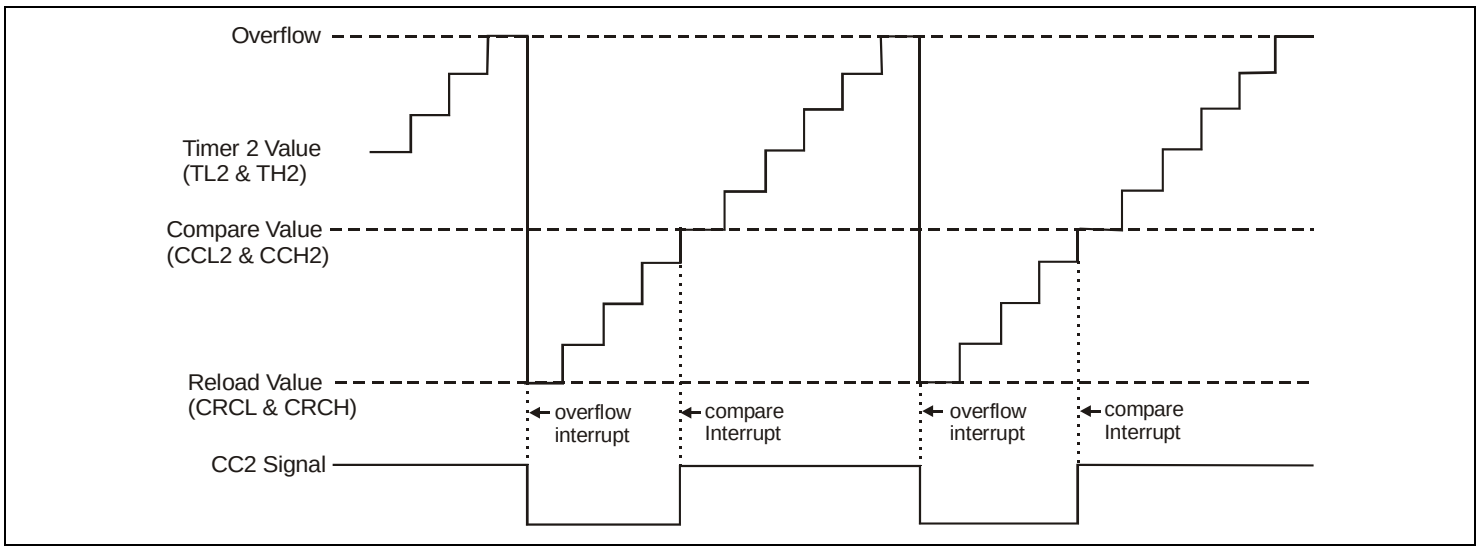



**Figure 12 – Timer 2: Reload Mode = 0, Compare Mode = 0 (using CC2)**




## 17.3. Watchdog Timer

The watchdog timer provides a means to trigger a system reset in case of software upset. When this timer is running, it must receive regular "refresh" signals from the software. If these signals cease, the timer sends a signal that causes a reset.

**Registers**

The 15-bit watchdog timer value is stored internally and is not accessible to software. A seven-bit reload value is stored in register WDTREL. Other important bits are in registers IEN0, IEN1, and IP0.

**Signals**

Input pin SWD starts the watchdog timer. Bit SWDT (in IEN1) can also start the timer. Bits WDT (in IEN0) and SWDT are used to refresh the timer. Bit WDTS (in IP0) is used to trigger a system reset.

**Operation**

Starting: Each external reset disables the watchdog timer, clears its value, and sets WDTREL to 00h. The watchdog timer starts to run if the SWD signal is asserted during reset. Otherwise, the watchdog timer can be started at any time by setting bit SWDT. Once activated, the watchdog timer cannot be stopped except by a reset.

Running: The watchdog timer value is incremented according to the value of the PS bit in WDTREL (can be written or read at any time). If PS = 0, the timer increments every 24 clock cycles; if PS = 1, a divide-by-16 prescaler is added, and the timer increments once every 384 clock cycles. When the watchdog timer value reaches 7CFFh, it sets bit WDTS (in register IP0). After two clock cycles, the system initiates a reset.

Refreshing: To prevent a system reset, the software must periodically refresh the value of the watchdog timer. This is a two-step process: first, a '1' is written to bit WDT; then, within the next 12 clock cycles, a '1' must be written to bit SWDT. The watchdog timer is then reloaded with the value stored in the lower seven bits of WDTREL (can be written or read at any time). Bits WDT and SWDT are each cleared twelve clock cycles after they are written, or when the timer is reloaded, whichever occurs first.

## 18. Serial Ports

There are two serial ports, serial port 0 and serial port 1. These ports are entirely separate from the Serial Peripheral Interface (SPI). Each serial port can simultaneously transmit and receive; one byte of received data is buffered to prevent data loss. The two serial ports offer various operating modes and baud rates and allow multiprocessor communication.

**Registers**

Each serial port uses one special function register (S0BUF or S1BUF) to access two separate internal buffers, one for transmit and one for receive. PORT1 and PORT3 contain the RXD and TXD pins. S0CON and S1CON control and configure the serial ports. IEN0 and IEN2 enable/disable the serial channel interrupts. Additional baud rate data may be contained in S0RELL, S0RELH, S1RELL, S1RELH, PCON, WDCON, and TH1.

**Operation**

Serial data is transmitted one bit at a time. Each serial port has one transmit pin and one receive pin. The ports are full-duplex – that is, each port can simultaneously send and receive data. Assuming that the ports have been correctly configured, operation proceeds as follows:

Receive: Bit RENx must be set in order to enable reception. As data bits arrive on the RXDx pin, they are stored in the serial port's receive buffer. When a full byte has been received, the hardware sets the RIx flag in the SxCON register to request an interrupt; the received byte is now available in the RxBUF register and new data bits can be received in the receive buffer (except Mode 0; see below).

Transmit: Writing a byte of data into the SxBUF register fills the corresponding output buffer and begins transmission. When the byte has been sent, hardware sets the TIx flag in the SxCON register to request an interrupt; the next data byte can now be written to SxBUF.

### 18.1. Serial Port 0 Modes

Serial port 0 can operate in four different modes, as configured by bits SM0 and SM1 in the S0CON register.

**Mode 0** (Shift Register): Serial data is transmitted and received through pin RXD0 (PORT3[0]) while pin TXD0 (PORT3[1]) outputs the shift clock. Bit RI0 must be cleared to enable reception. Each byte transmitted/received contains 8 bits; the least significant bit (LSB) is always first.

**Mode 1** (8-bit UART): Serial data bits are transmitted through pin TXD0 and received through pin RXD0. No external shift clock is used. Each byte uses 10 bits: a start bit (always 0), 8 data bits (LSB first), and a stop bit (always 1). On receive, the start bit synchronizes the communication, the 8 data bits are available in S0BUF, and the stop bit sets flag RB80 in register S0CON; RB80 may be cleared by software.

**Mode 2** (9-bit UART): Much like Mode 1, but each byte uses 11 bits: start bit (0), 8 data bits (LSB first), a programmable ninth data bit, and a stop bit (1). The $9^{th}$ bit can be used for parity or to support multiprocessor communication (see below). On transmit, the $9^{th}$ data bit is taken from TB80 in S0CON. On receive, the $9^{th}$ data bit goes into RB80 and the stop bit is discarded.

**Mode 3** (9-bit UART): Exactly like Mode 2 except that the baud rate is calculated differently.

## 18.2. Serial Port 1 Modes

Serial port 1 can operate in two different modes, as configured by bit SM in register S1CON. Transmit uses pin TXD1; receive uses pin RXD1.

**Mode A** (9-bit UART): This is exactly like Modes 2 and 3 of serial port 0 except for bit locations and baud rates. Data bits are in S1BUF; TB81 in S1CON determines the 9th data bit on transmission; on receive, the 9th data bit is stored in RB81.

**Mode B** (8-bit UART): This is exactly like Mode 1 of serial port 0 except for bit locations and baud rates. Data bits are in S1BUF; on receive, the stop bit sets flag RB81 in S1CON (may be cleared by software).

## 18.3. Multiprocessor Communication

Any of the 9-bit UART modes can be used for multi-processor communication. In this case, the slave processor(s) must have a '1' value in bit SM2x of SxCON. When the master processor transmits the slave's address, it sends a '1' value as the 9th bit, causing a serial port receive interrupt in all slave processors. Each slave processor then compares the received byte to its network address. If there is a match, the addressed slave clears bit SM2x and receives the rest of the message; the other slaves leave the bit set to '1' and ignore the message. The master processor sends the rest of the message with each 9th bit set to '0' so that no interrupts are generated in the unselected slaves.

## 18.4. Serial Port Baud Rates

**Mode 0:** Baud rate is fixed at 1/12 clock rate.

**Mode 1:** Baud rate depends on bit BD in register WDCON and SMOD bit in register PCON.

If BD = 0, the baud rate is determined by the value of register TH1 (part of Timer 1), using this formula:

baud rate = ($2^{SMOD}$ x ClockRate) / (384 x (256 – TH1))

If BD = 1, the baud rate is determined by the value of S0REL (registers S0RELL and S0RELH) using this formula:

baud rate = ($2^{SMOD}$ x ClockRate) / (64 x (1024 – S0REL))

**Mode 2:** Baud rate is determined by bit SMOD in PCON using this formula:

baud rate = ($2^{SMOD}$ x ClockRate) / 64

Therefore, if SMOD = 0, baud rate is 1/64 clock rate; if BD = 1, baud rate is 1/32 clock rate.

**Mode 3:** Same as Mode 1.

**Modes A and B:** Baud rate is determined by S1REL (registers S1RELL and S1RELH) using this formula:

baud rate = ClockRate / (32 x (1024 – S1REL))

## 18.5. Serial Port Timing

**Figure 13 – Receive Timing, Mode 0**

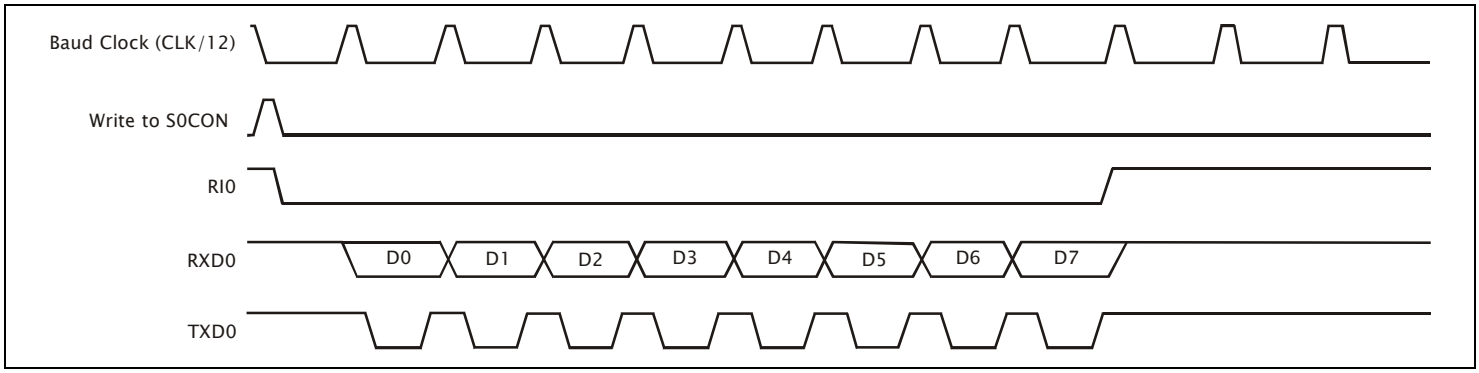


**Figure 14 – Receive Timing, Modes 1 and B**

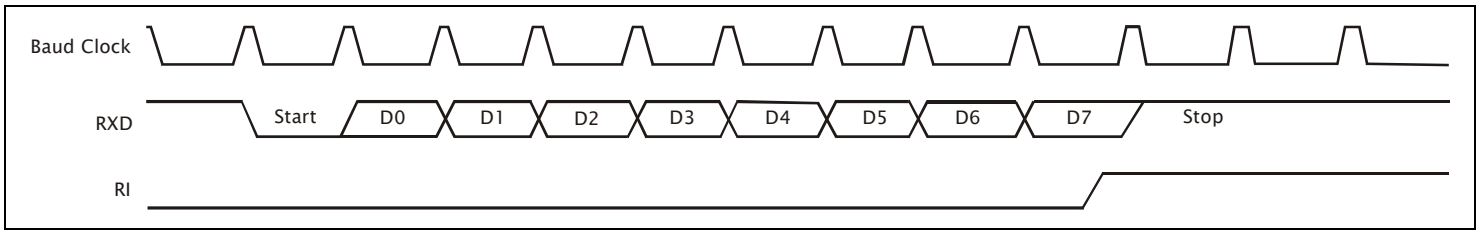



**Figure 15 – Receive Timing, Modes 2, 3, and A**

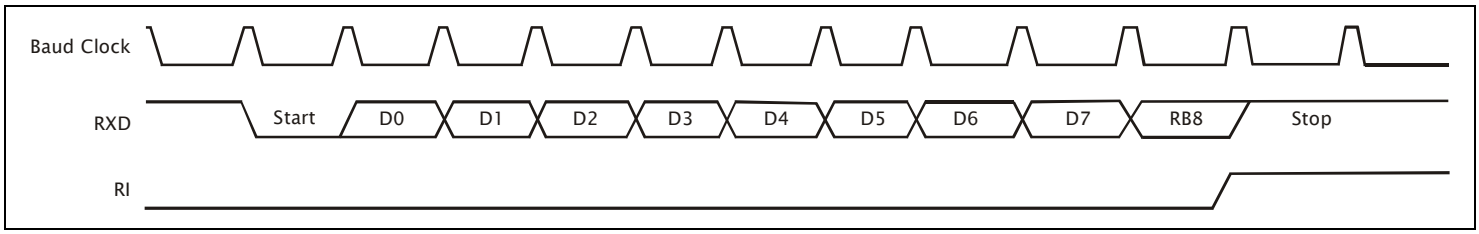



**Figure 16 – Transmit Timing, Mode 0**

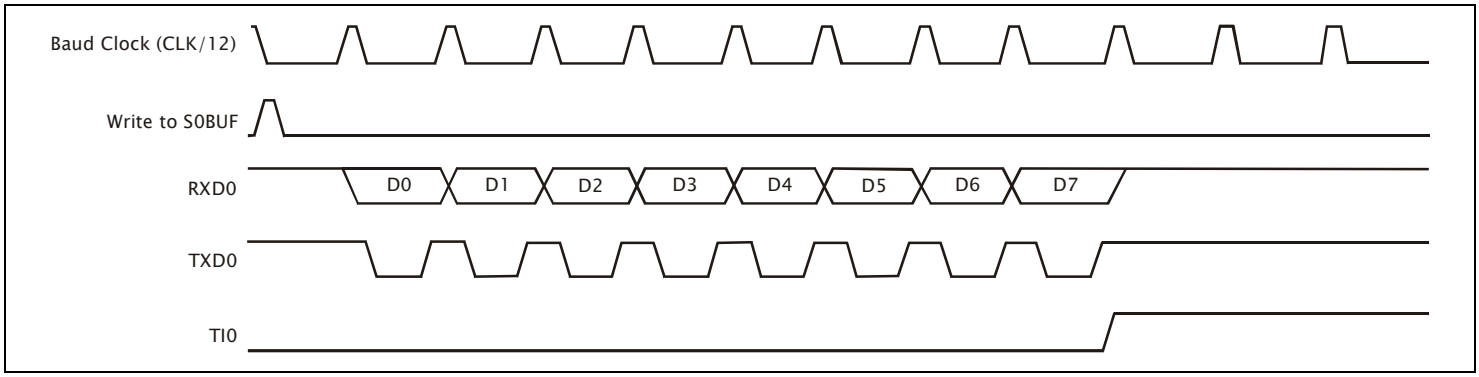



**Figure 17 – Transmit Timing, Modes 1 and B**

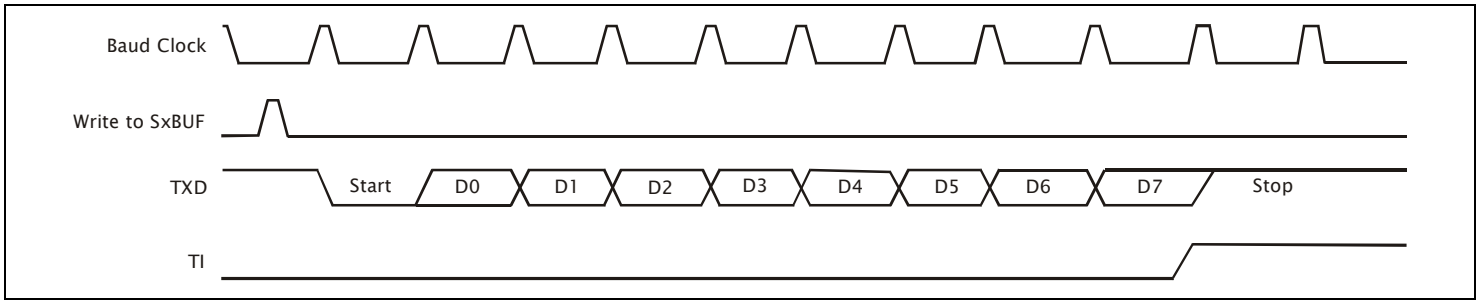
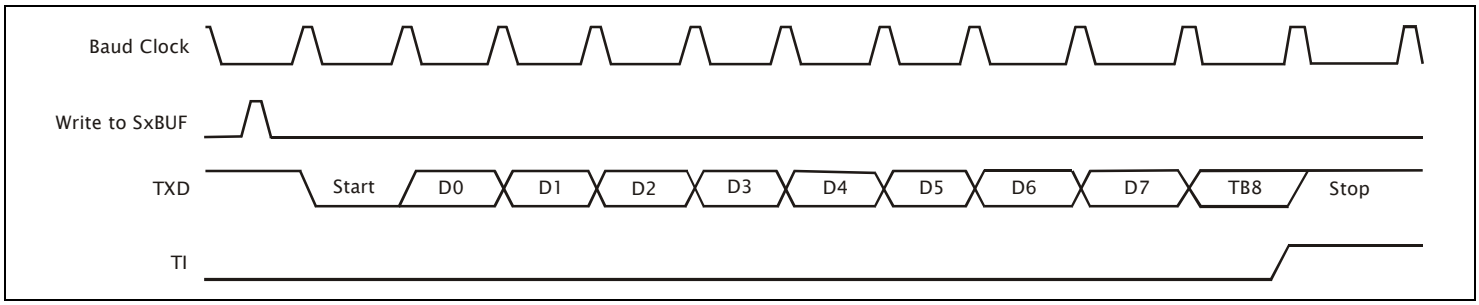


**Figure 18 – Transmit Timing, Modes 2, 3, and A**




## 19. Interrupts

The TSCR8051 supports twelve interrupt sources; some are triggered by external signals, others by internal events. Each source sets a request flag in a special function register. Each source can be enabled, disabled, prioritized, and otherwise configured by special function register settings. After a reset, all interrupts are disabled.

There are four levels of priority for interrupts. Priorities are assigned to each of six interrupt "groups", where a "group" contains two interrupt sources. Priorities are controlled by registers IP0 and IP1.

**Table 14: Interrupt Summary**

| NAME | SIGNAL (PORT) | TRIGGER | PRIORITY GROUP | FLAG (REGISTER) | CONFIGURATION REGISTERS |
|---|---|---|---|---|---|
| External Interrupt 0 | INT0 (3) | Signal Low (or Fall) | 0 | IE0 (TCON) | IEN0, TCON |
| External Interrupt 1 | INT1 (3) | Signal Low (or Fall) | 2 | IE1 (TCON) | IEN0, TCON |
| External Interrupt 2 | INT2 (0) | Signal Fall (or Rise) | 1 | IEX2 (IRCON) | IEN0, IEN1, T2CON |
| External Interrupt 3 | INT3 (0) | Signal Fall (or Rise) * | 2 | IEX3 (IRCON) | IEN0, IEN1, T2CON |
| External Interrupt 4 | INT4 (0) | Signal Rise * | 3 | IEX4 (IRCON) | IEN0, IEN1 |
| External Interrupt 5 | INT5 (0) | Signal Rise * | 4 | IEX5 (IRCON) | IEN0, IEN1 |
| External Interrupt 6 | INT6 (0) | Signal Rise * | 5 | IEX6 (IRCON) | IEN0, IEN1 |
| Serial Channel 0 | (none) | Rx/Tx Complete | 4 | RI0, TI0 (S0CON) | IEN0 |
| Serial Channel 1 | (none) | Rx/Tx Complete | 0 | RI1, TI1 (S1CON) | IEN0, IEN2 |
| Timer 0 | (none) | Overflow | 1 | TF0 (TCON) | IEN0 |
| Timer 1 | (none) | Overflow | 3 | TF1 (TCON) | IEN0 |
| Timer 2 | T2EX (1) | Signal Fall, Overflow * | 5 | EXF2, TF2 (IRCON) | IEN0, IEN1 |

* Timer 2 Compare/Capture events can trigger interrupts on pins CC0-CC3 (INT3-INT6); see page 47.

When an interrupt condition occurs, the corresponding flag is set, regardless of whether that interrupt is enabled. The flags are continually polled by the hardware.

If a flag indicates a pending interrupt and that interrupt is enabled, the next instruction cycle will force an LCALL to the appropriate vector address.

Once interrupt service has begun, it can be interrupted only by a higher priority interrupt. The interrupt service is terminated by an RETI instruction. The fastest possible response to an interrupt is 7 cycles – one to detect the interrupt and six to perform the LCALL.

Most flags must be cleared by software, normally within the interrupt service routine. The only exceptions are TF0 and TF1, which are automatically cleared when the service routine is called.

**Table 15: Interrupt Polling Sequence**

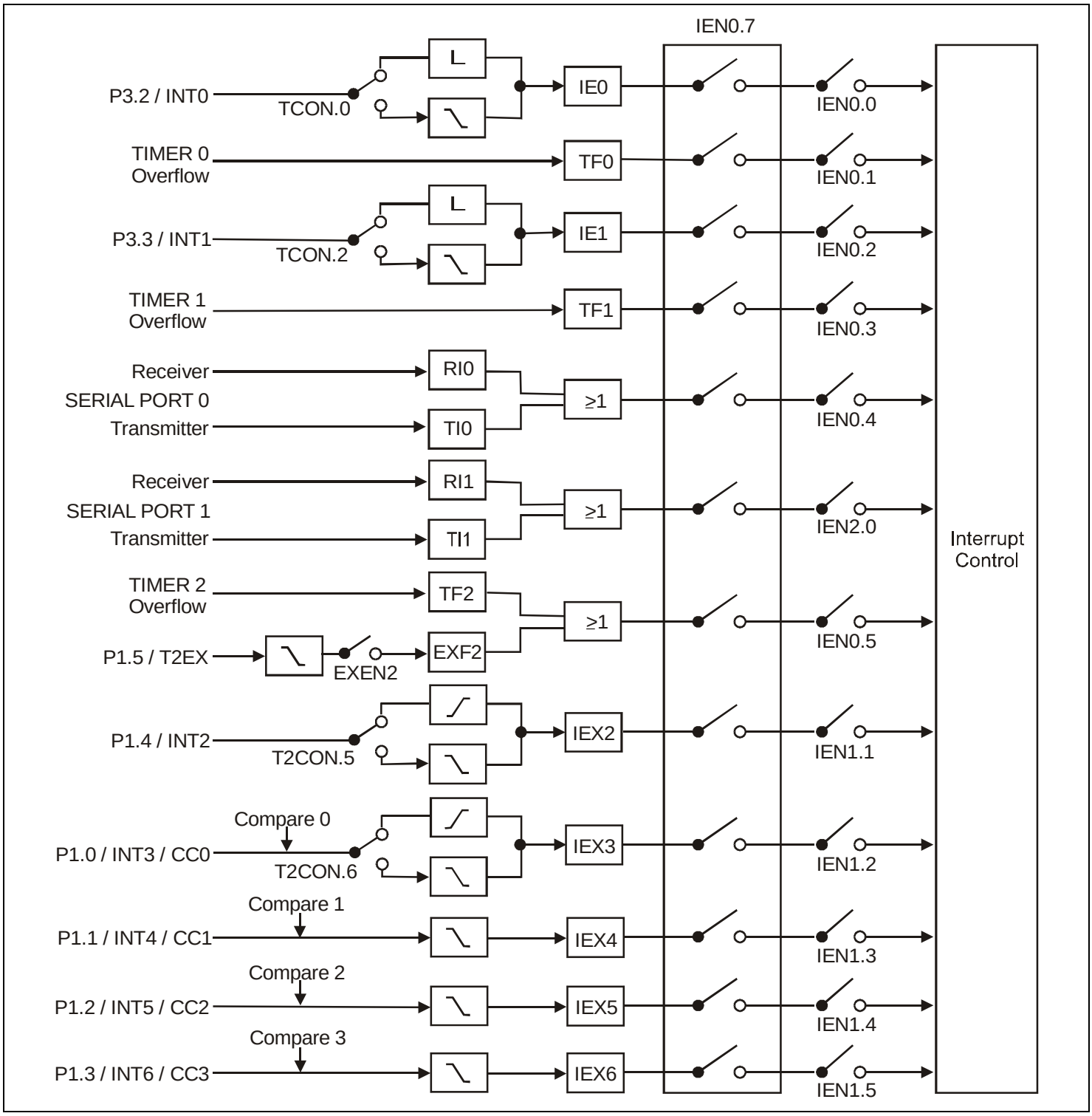
| |
|---|
| External interrupt 0 |
| Serial channel 1 |
| Timer 0 |
| External interrupt 2 |
| External interrupt 1 |
| External interrupt 3 |
| Timer 1 |
| External interrupt 4 |
| Serial channel 0 |
| External interrupt 5 |
| Timer 2 |
| External interrupt 6 |

**Table 16: Interrupt Vectors**

| FLAG – INTERRUPT SOURCE | VECTOR ADDRESS |
|---|---|
| IE0 – External interrupt 0 | 0003h |
| TF0 – Timer 0 | 000Bh |
| IE1 – External interrupt 1 | 0013h |
| TF1 – Timer 1 | 001Bh |
| RI0/TI0 – Serial channel 0 | 0023h |
| TF2/EXF2 – Timer 2 | 002Bh |
| RI1/TI1 – Serial channel 1 | 0083h |
| IEX2 – External interrupt 2 | 004Bh |
| IEX3 – External interrupt 3 | 0053h |
| IEX4 – External interrupt 4 | 005Bh |
| IEX5 – External interrupt 5 | 0063h |
| IEX6 – External interrupt 6 | 006Bh |

**Figure 19 – Interrupt Processing**



## 20. Floating Point Unit (FPU)

The FPU is a single-precision (32-bit) floating point unit, fully IEEE 754 compliant, with a 32-bit comparator.

**Table 17: Floating-Point Unit (FPU) Registers**

| LOCATION | NAME | FUNCTION |
|---|---|---|
| A3h | FPUCON | Specifies FPU operation and rounding |
| A4h – A7h | OPA3/2/1/0 | Floating Point Operand A |
| ABh | FPCS | Comparator Flags – Zero, Equal, Infinite, etc. |
| ACh – AFh | OPB3/2/1/0 | Floating Point Operand B |
| B3h | FPUS | Status Flags – QNaN, Inexact, etc. |
| B4h – B7h | FPUR3/2/1/0 | Floating Point Result |

**General Operation**

Each FPU operation requires four clock cycles. The FPU processes its operands continuously, in pipelined fashion, updating the results every clock cycle. The order in which the registers are written does not matter. When the correct values are in FPUCON and the operand registers, the results and flags will appear four cycles later.

**NaN Values**

Per the IEEE 754 standard, valid FPU operations can generate “Not-a-Number” (NaN) results; for example, subtracting infinity from infinity produces NaN. There are two categories of NaN: QNaN (Quiet Nan) indicates an indeterminate operation and SNaN (Signaling NaN) indicates an invalid operation. Although either operand may be SNaN, the FPU will never generate SNaN output; any NaN result will be QNaN.

**Conversions**

When a floating point number is converted to an integer, the integer output may be NaN or infinity, both of which are legal integer values. In those cases, the INF or QNAN flag is not set. However, if the input is NaN, the SNaN flag is set.

## 21. Extended Computing Functions

These functions allow the TSCR8051 to perform some of the standard vector processing used in super-computing applications.

### 21.1. 32-bit Leading-Zero Counter

An internal 32-bit register counts leading zeroes written to an 8-bit register. For this application, ‘leading zeros’ are 0 bits written before a 1 is written; more significant bits are written before less significant bits. Once a ‘1’ has been written, the leading zero count does not change until the internal register is cleared.

**Table 18: Leading Zero Count Registers**

| LOCATION | NAME | FUNCTION |
|---|---|---|
| 96h | LZCON | Controls and reports leading-zero conditions. |
| 97h | LZC | Write: Updates the internal counter.<br>Read: returns one byte of the internal counter |

Various bits in LZCON are used to clear the counter, report an overflow, and control the “read” function of LZC. Two bits in LZCON determine which byte of the internal counter is available via LZC. If these bits are 11b, reading LZC returns the most significant byte of the counter. Whenever LZC is read, the two-bit value is decremented so that the next reading of LZC returns the next lower byte. When the two-bit value is 00b, reading LZC returns the least significant byte of the counter and the two-bit value cycles back to 11b. If the LZM bit (LZCON.3) is set, reading the least significant byte will also clear the counter.

## 21.2. 32-bit Population Counter

An internal 32-bit register counts the number of ones written to an 8-bit register. This function behaves much like the leading-zero function except that the count does not halt when a zero is written.

**Table 19: Leading Zero Count Registers**

| LOCATION | NAME | FUNCTION |
|---|---|---|
| 9Eh | PCCON | Controls and reports population count conditions. |
| 9Fh | POPC | Write: Updates the internal counter.<br>Read: Returns one byte of the internal counter |

Various bits in PCCON are used to clear the counter, report an overflow, and control the "read" function of POPC. Two bits in PCCON determine which byte of the internal counter is available via POPC. If these bits are 11b, reading POPC returns the most significant byte of the counter. Whenever POPC is read, the two-bit value is decremented so that the next reading of POPC returns the next lower byte. When the two-bit value is 00b, reading POPC returns the least significant byte of the counter and the two-bit value cycles back to 11b. If the POPM bit (PCCON.3) is set, reading the least significant byte will also clear the counter.

## 22. SPI Memory Loader

The SPI (Serial Peripheral Interface) is a special-purpose interface, completely separate from the I/O ports and the serial ports. Immediately upon power-up or system reset, the microprocessor's program memory is loaded through the SPI from an SPI-compatible device (e.g., Atmel AT25xxxx). The memory loader uses the SPI pins and some of the Clock pins as described on page 7.

## 23. Reset Control

This unit can be reset by an external signal (RESET) or by the watchdog timer's internal signal (WDTS). When either of these signals is held high for two cycles while the oscillator is running, all registers and flip-flops are reset. Immediately after the reset, program memory is loaded via the SPI circuitry (see above).

## 24. Power Management

Two power-saving modes, IDLE and POWER-DOWN, can be invoked by setting bits in the PCON register.

**IDLE**

This mode is invoked by setting the IDL bit. The core becomes non-active, reducing power consumption. The CPU, ALU, and memory unit are stopped, but the internal clocks and peripherals continue to run. Serial ports, timers, etc. are active, and memory retains its state. A reset or any interrupt will cause the unit to exit the IDLE mode.

**POWER-DOWN**

This state is invoked by setting the PD bit. All internal clocking is turned off. Memory retains its state. A reset or any external non-clocked interrupt causes the unit to exit this state. (Internal interrupts require clocking, and will not occur in the POWER-DOWN state.)

# 25. Device Specifications

## 25.1. Absolute Maximum Ratings

| SYMBOL | PARAMETER | MIN | MAX | UNIT |
|---|---|---|---|---|
| $V_{DDQ}$ | 3.3 V DC Supply Voltage | -0.5 | 4.6 | V |
| $V_{DDQF}$ | 3.3 V DC Filtered Supply Voltage | -0.5 | 4.6 | V |
| $V_{DD}$ | 1.8 V DC Supply Voltage | -0.5 | 3.5 | V |
| $V_{IN}$ | Input Voltage | -0.5 | 6.0 | V |
| $V_{OUT}$ | Output Voltage (outputs active) | 0 | $V_{DD}$ + 0.5 | V |
| | Output Voltage (outputs disabled) | 0 | 6.0 | V |
| $T_{STG}$ | Storage Temperature | -65 | 150 | °C |

Absolute maximum ratings are those values beyond which damage to the device may occur. Exposure to conditions beyond those indicated may adversely affect device reliability. Functional operation under absolute maximum ratings is not implied.

## 25.2. Recommended Operating Conditions

| SYMBOL | PARAMETER | MIN | MAX | UNIT |
|---|---|---|---|---|
| $V_{DDQ}$ | 3.3 V DC Supply Voltage | 3.0 | 3.6 | V |
| $V_{DDQF}$ | 3.3 V DC Filtered Supply Voltage | 3.0 | 3.6 | V |
| $V_{DD}$ | 1.8 V DC Supply Voltage | 1.62 | 1.98 | V |
| $V_{IN}$ | Normal 3.3 Input Voltage | 0 | 3.6 | V |
| | 5V Tolerant Input Voltage | 0 | 5.5 | V |
| $V_{OUT}$ | Output Voltage (outputs active) | 0 | $V_{DD}$ | V |
| | Output Voltage (outputs disabled) | 0 | 5.5 | V |
| $T_J$ | Junction Temperature | 0 | 125 | °C |

## 25.3. DC Characteristics

| SYMBOL | PARAMETER | MIN | MAX | UNIT |
|---|---|---|---|---|
| $V_{IH}$ | Input High Voltage | 2.0 | – | V |
| $V_{IL}$ | Input Low Voltage | – | 0.8 | V |
| $V_{OH}$ | Output High Voltage | 2.4 | – | V |
| $V_{OL}$ | Output Low Voltage | – | 0.4 | V |
| $I_{OZ}$ | 3-State Output Leakage Current | -10 | 10 | µA |

## 25.4. AC Characteristics

*Not available at time of revision.*

# 26. Package Dimensions

*Not available at time of revision.*

## 27. Document History

| REVISION | DATE | NOTES |
|---|---|---|
| Draft 0.1 | June 8, 2010 | incomplete |
| | | |