SX48BD

Configurable Communications Controllers with EE/Flash Program Memory, In-System Programming Capability and On-Chip Debug

1.0 PRODUCT OVERVIEW

1.1. Introduction

The Parallax SX48BD is a member of the SX family of configurable communications controllers fabricated in an advanced CMOS process technology. The advanced process, combined with a RISC-like architecture, allows high-speed computation, flexible I/O control, and efficient data manipulation. Throughput is enhanced by operating the device at frequencies up to 75 MHz and by optimizing the instruction set to include mostly single-cycle instructions. In addition, the SX architecture is deterministic and fully reprogrammable. The unique combination of these characteristics enable the device to serve in multitasking applications, high-speed communications, and virtually every application where speed and accuracy are essential.

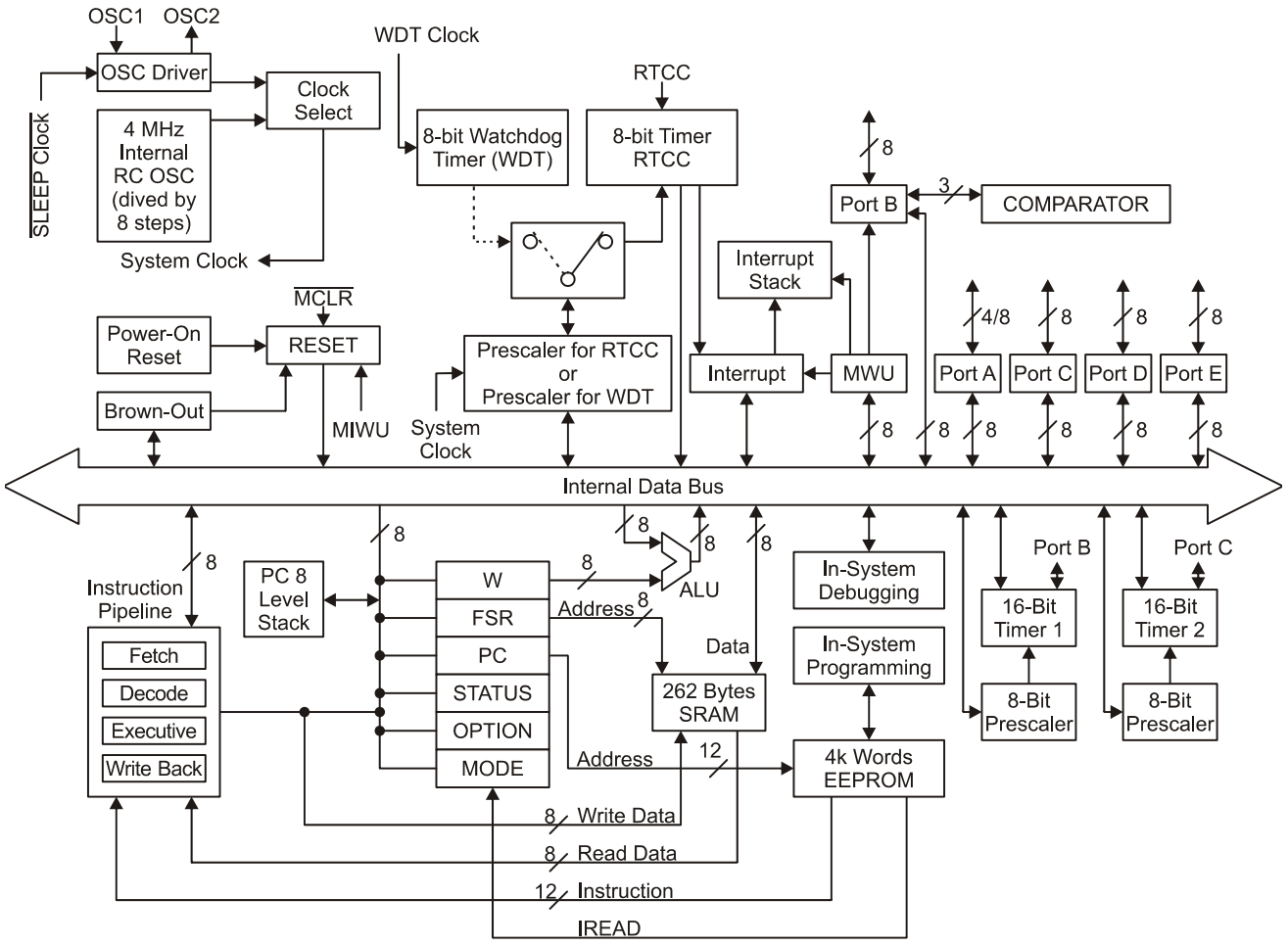


Figure 1-1: Block Diagram

Table of Contents

1.2. Key Features

75 MIPS Performance

- DC - 75 MHz operation
- 13.3 ns instruction cycle, 39.9 ns internal interrupt response at 75 MHz
- 1 instruction per clock for most instructions (skips require 2 clocks, branches require 3 clocks, IREAD requires 4)

EE/FLASH Program Memory and SRAM Data Memory

- Access time of < 13.3 ns provides single cycle access
- EE/Flash rated for > 10,000 rewrite cycles
- 4096 Words of EE/Flash program memory
- 262x8 bits SRAM data memory

CPU Features

- Compact, RISC-like instruction set
- All non-branch instructions are single cycle
- Eight-level push/pop hardware stack for subroutine linkage
- Fast table lookup capability through run-time readable code (IREAD instruction)
- Predictable program execution rate for hard real-time applications

Fast and Deterministic Interrupt

- Jitter-free 3-cycle internal interrupt response
- Hardware context save/restore of key resources such as PC, W, STATUS, and FSR within the 3-cycle interrupt response time
- External wakeup/interrupt capability on Port B (8 pins)

Flexible I/O

- All port pins individually programmable as I/O
- Inputs are TTL or CMOS level selectable
- All pins have selectable internal pull-ups
- Selectable Schmitt Trigger inputs on Ports B, C, D, and E
- All output pins capable of sourcing/sinking 30 mA
- Port A outputs have symmetrical drive
- Analog comparator support on Port B (RB0 OUT, RB1 IN-, RB2 IN+)
- Selectable I/O operation synchronous to the oscillator clock

Hardware Peripheral Features

- Two 16-bit timers with 8-bit prescalers supporting:
- Software Timer mode
- PWM mode
- Simultaneous PWM/Capture mode
- External Event mode
- One 8-bit Real Time Clock/Counter (RTCC) with programmable 8-bit prescaler
- Watchdog Timer (shares the RTCC prescaler)
- Analog comparator
- Brown-out detector
- Multi-Input Wakeup logic on 8 pins
- Internal RC oscillator with configurable rate from 31.25 kHz to 4 MHz
- Power-On-Reset

Package

- 48-pin Tiny PQFP

Programming and Debugging Support

- On- chip in-system programming support with serial or parallel interface
- In-system serial programming via oscillator pins
- On-chip in-System debugging support logic
- Real-time emulation, full program debug, and integrated development environment offered by third party tool vendors

Software Support

- Native assembly instruction set
- Expanded assembly instruction set available in the SASM assembler of the Parallax SX-Key IDE
- Parallax SX/B compiler (BASIC)
- Several “C” compliers available from third-party vendors

1.3. Architecture

The SX devices use a modified Harvard architecture. This architecture uses two separate memories with separate address buses, one for the program and one for data, while allowing transfer of data from program memory to SRAM. This ability allows accessing data tables from program memory. The advantage of this architecture is that instruction fetch and memory transfers can be overlapped with a multi-stage pipeline, which means the next instruction can be fetched from program memory while the current instruction is being executed using data from the data memory.

The SX uses a revolutionary RISC-like architecture and memory design technique that is 15 times faster than conventional MCUs, deterministic, jitter free, and fully reprogrammable. The SX family implements a four-stage pipeline (fetch, decode, execute, and write back), which results in execution of one instruction per clock cycle. At the maximum operating frequency of 75 MHz, instructions are executed at the rate of one per 13.3-ns clock cycle.

1.4. Programming Benefits in Assembly and High-Level Languages

The SX's high speed enables a "software system on a chip" approach. Programming in assembly language provides a particularly high-level of access to the interrupt service routine, the stack and registers to take the highest advantage of the SX's deterministic timing. The primary technical resources for programming the SX in assembly language include the following:

- The SX48BD datasheet
- *SX-Key Development System User's Manual* by Parallax, Inc.
- *Programming the SX Microcontroller – A Complete Guide* by Guenther Daubach

Customers with a high-level programming language background may prefer the use of a C or BASIC compiler.

1.4.1. Parallax SX/B Basic Compiler

Parallax's SX/B is a free BASIC language compiler for the SX microcontroller (SX20, SX28, and SX48). The compiler speeds the programming of the SX microcontrollers by providing a simple, yet robust high-level language familiar to Parallax customers. SX/B includes the following features and commands:

- ASM directive to support in-line assembly language
- Program structure commands including BRANCH, DO..LOOP, GOTO, GOSUB, IF..THEN..ELSE
- Numeric formatters
- WORD variable support
- Frequency generation with FREQOUT
- Synchronous serial communication for I2C, 1-Wire, SPI
- Asynchronous serial communication with SERIN and SEROUT
- Table data storage and retrieval with LOOKUP, LOOKDOWN
- I/O pin control with HIGH, LOW, TOGGLE, REVERSE
- Timing and delay with PAUSE, SLEEP
- PULSIN and PULSOUT
- Resistor/capacitor A/D with RCTIME
- RANDOM for pseudo-random number generation
- Non-volatile EEPROM memory access with DATA, READ
- Low-current SLEEP command

The complete SX/B command reference and examples are installed with the SX-Key IDE.

1.5. Programming and Debugging Support

The SX devices are supported by Parallax's programming and debugging tools. The Parallax SX-Blitz is a programming tool. The SX-Key supports programming and source-level debugging. On-chip in-system debug capabilities allow the Parallax tool to be an all-in-one integrated development environment with editor, macro assembler, debugger, and programmer. Unobtrusive in-system programming is provided through the OSC pins. Visit www.parallax.com for the SX-Key development tools, the IDE and support forum information.

The in-system programming specification is available to other 3rd party tool vendors upon request.

1.6. Applications

The SX may be used as a solution for process controllers, electronic appliances/tools, security/monitoring systems, sound and signal generation, GPS interface, robotic control, motor control, sensor interfacing and personal communication devices. Applications such as interactive toys, magnetic-stripe readers, infrared decoders, and other timing-sensitive projects are also common with the SX. Examples of customer applications may be seen on the Parallax web site.

1.7. Support

Parallax and our distributors provide support for the SX microcontroller. Support is available free of charge via phone (888) 512-1024 in the U.S. Also be sure to participate in the SX discussion forum at http://forums.parallax.com/forums/. The on-line SX support community is actively involved in customer support 24 hours a day.

1.8. Part Numbering

Table 20-1: Ordering Information							
Device Part#	**Pins**	**I/O**	**EE/Flash (Words)**	**RAM (Bytes)**	**Voltage Range (V)**	**Operating Temp @ 3.0 – 5.5 V, 50 MHz***	**Operating Temp @ 4.5 – 5.5 V, 75 MHz***
SX48BD/TQ	48	36	4 K	262	3.0 – 5.5	-40 °C to +85 °C	0 °C to +70 °C
SX48BD/TQ-G	48	36	4 K	262	3.0 – 5.5	-40 °C to +85 °C	0 °C to +70 °C

***Ratings are preliminary**

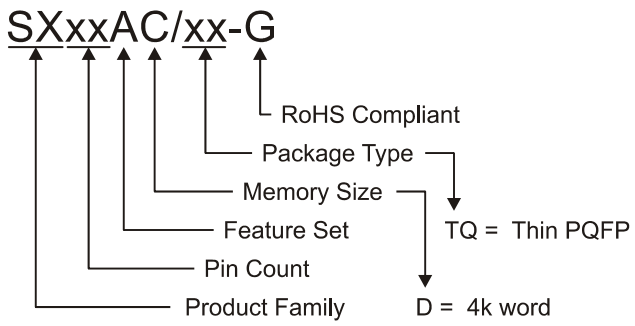
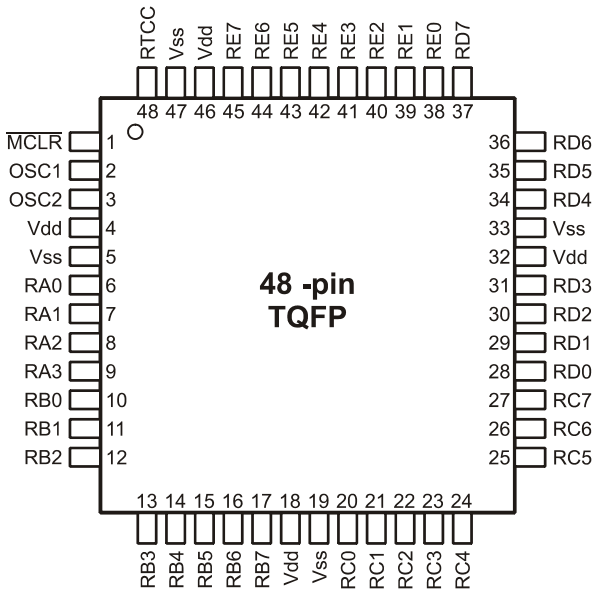


Figure 1-1

Part Number Reference Guide

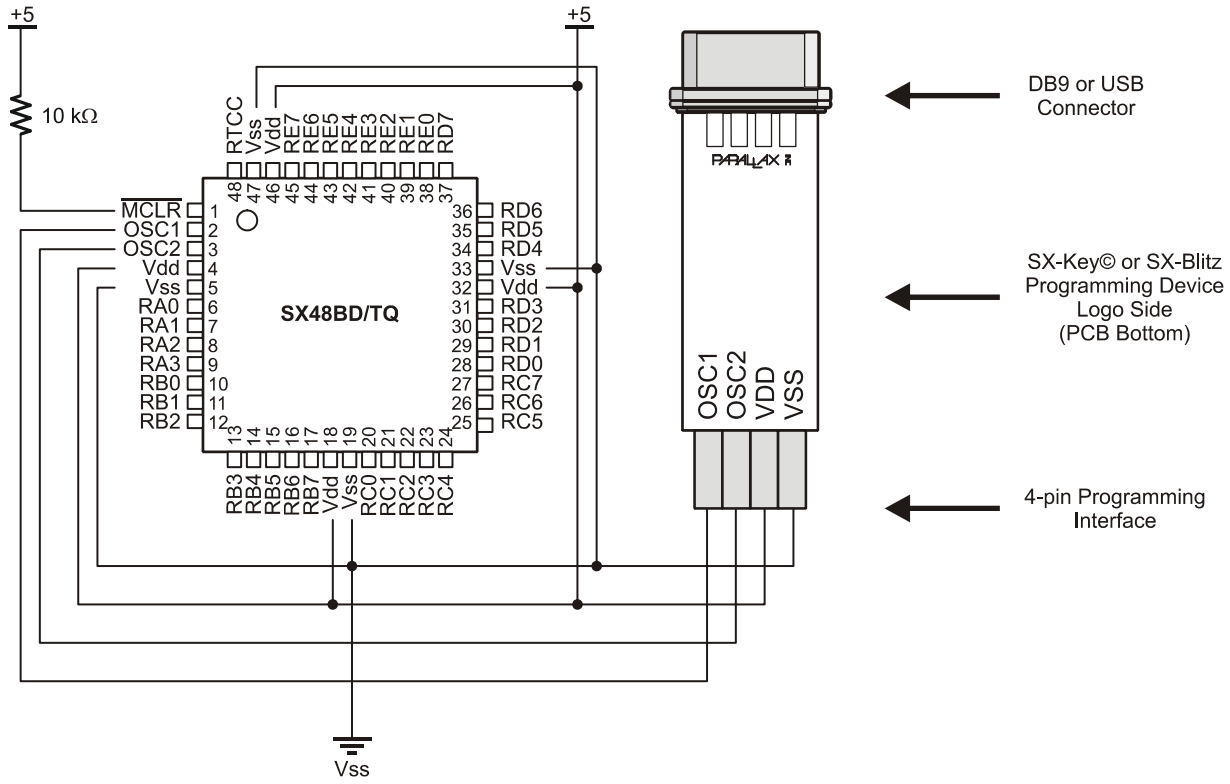
2.0 CONNECTION DIAGRAMS

2.1. Pin Assignments



Top View

2.2. Typical Connection Diagram



2.3. Pin Descriptions

Table 2-1: Pin Descriptions

Name	Pin Type	Input Levels	Description
RA0	I/O	TTL/CMOS	Bidirectional I/O Pin; symmetrical source / sink capability
RA1	I/O	TTL/CMOS	Bidirectional I/O Pin; symmetrical source / sink capability
RA2	I/O	TTL/CMOS	Bidirectional I/O Pin; symmetrical source / sink capability
RA3	I/O	TTL/CMOS	Bidirectional I/O Pin; symmetrical source / sink capability
RB0	I/O	TTL/CMOS/ST	Bidirectional I/O Pin; comparator output; MIWU/Interrupt input
RB1	I/O	TTL/CMOS/ST	Bidirectional I/O Pin; comparator negative input; MIWU/Interrupt input
RB2	I/O	TTL/CMOS/ST	Bidirectional I/O Pin; comparator positive input; MIWU/Interrupt input
RB3	I/O	TTL/CMOS/ST	Bidirectional I/O Pin; MIWU/Interrupt input,
RB4	I/O	TTL/CMOS/ST	Bidirectional I/O Pin; MIWU/Interrupt input, Timer T1 Capture Input 1
RB5	I/O	TTL/CMOS/ST	Bidirectional I/O Pin; MIWU/Interrupt input, Timer T1 Capture Input 2
RB6	I/O	TTL/CMOS/ST	Bidirectional I/O Pin; MIWU/Interrupt input, Timer T1 PWM/Compare Output
RB7	I/O	TTL/CMOS/ST	Bidirectional I/O Pin; MIWU/Interrupt input, Timer T1 External Event Counter Input
RC0	I/O	TTL/CMOS/ST	Bidirectional I/O Pin, Timer T2 Capture Input 1
RC1	I/O	TTL/CMOS/ST	Bidirectional I/O Pin, Timer T2 Capture Input 2
RC2	I/O	TTL/CMOS/ST	Bidirectional I/O Pin, Timer T2 PWM/compare Output
RC3	I/O	TTL/CMOS/ST	Bidirectional I/O Pin; Timer T2 External Event Counter Input
RC4	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RC5	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RC6	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RC7	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RD0	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RD1	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RD2	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RD3	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RD4	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RD5	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RD6	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RD7	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RE0	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RE1	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RE2	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RE3	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RE4	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RE5	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RE6	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RE7	I/O	TTL/CMOS/ST	Bidirectional I/O Pin
RTCC	I	ST	Input to Real-time Clock/Counter
$\overline{\text{MCLR}}$	I	ST	Master Clear reset input – active low. When not controlled externally, this pin must be pulled high with a 10 kΩ resistor.
OSC1/In/Vpp	I	ST	Crystal oscillator input – External Clock source input
OSC2/Out	O	CMOS	Crystal oscillator output; in R/C mode, internally pulled to V_{dd} through weak pull-up
V_{dd}	P	-	Positive supply pins (a total of 4, one on each side of the device)
V_{ss}	P	-	Ground pins (a total of 4, one on each side of the device)

Note: I = input, O = output, I/O = Input/Output, P = Power, TTL = TTL input, CMOS = CMOS input, ST = Schmitt Trigger input, MIWU = Multi-Input Wakeup input.

3.0 PORT DESCRIPTIONS

The device contains one 4-bit port (Port A) and four 8-bit I/O ports (Port B through Port E). Port A provides symmetrical drive capability. Each port has four associated 8-bit registers (Direction, Data, TTL/CMOS Select, and Pull-Up Enable) to configure each port pin as Hi-Z input or output, to select TTL or CMOS voltage levels, and to enable/disable the weak pull-up resistor. The least significant bit of the registers corresponds to the least significant port pin. To access these configuration registers, an appropriate value must be written into the MODE register. Upon power-up, all bits in these registers are initialized to "1".

The associated registers allow for each port bit to be individually configured under software control as shown in Table 3-2.

Ports B, C, D, and E have additional associated registers (Schmitt-Trigger Enable Registers ST_B and ST_C) to enable or disable the Schmitt Trigger function on each individual port pin as indicated in Table 3-1.

Table 3-1: Schmitt Trigger Select

Schmitt Trigger Enable Registers :ST_B, ST_C, ST_D, ST_E	
0	1
Enable	Disable

Port B also supports the on-chip differential comparator. Ports RB1 and RB2 are the comparator negative and positive inputs, respectively, while Port RB0 is the comparator output pin. Port B also supports the Multi-Input Wakeup feature on all eight pins.

Port B and Port C also support the multi-function timers T1 and T2. RB4 and RB5 are the T1 capture inputs, RB6 is the T1 PWM output, and RB7 is the T1 external event counter input. Similarly, RC0 and RC1 are the T2 capture inputs, RC2 is the T2 PWM output, and RC3 is the T2 external event counter input. Figure 3-1 shows the internal hardware structure and configuration registers for each pin of Port A. Figure 3-2 shows the same for each pin of Port B, C, D, or E.

Table 3-2: Port Configuration

Data Direction Registers: RA, RB, RC		TTL/CMOS Selected Registers: LVL_A, LVL_B, LVL_C, LVL_D, LVL_E		Pullup Enable Registers: PLP_A, PLP_B, PLP_C, PLP_D, PLP_E	
0	1	0	1	0	1
Output	Hi-Z Input	CMOS	TTL	Enable	Disable

3.1. Reading and Writing the Ports

The five ports are memory-mapped into the data memory address space. To the CPU, the five ports are available as the RA, RB, RC, RD, and RE file registers at data memory addresses 05h through 09h, respectively. Writing to a port data register sets the voltage levels of the corresponding port pins that have been configured to operate as outputs to a corresponding level, 1 = 5 V, 0 = 0 V. Reading from a data register reads either the voltage levels of the corresponding port pins or the data contained in the port data register depending on the status PORTRD bit contained in the T2CNTB register.

For example, suppose all four Port A pins are configured as outputs. To make RA0 and RA1 high and the remaining Port A pins low, you could use the following code:

```
mov     W,#$03  ;load W with the value 03h
                ;(bits 0 and 1 high)
mov     $05,W   ;write 03h to Port A data
                ;register
```

The second “mov” instruction in this example writes the Port A data register (RA), which controls the output levels of the Port A pins, RA0 through RA7. When a write is performed to a port bit position that has been configured as an input, a write to the port data register is still performed, but it has no immediate effect on the pin. If later that pin is configured to operate as an output, it will reflect the value that has been written to the data register.

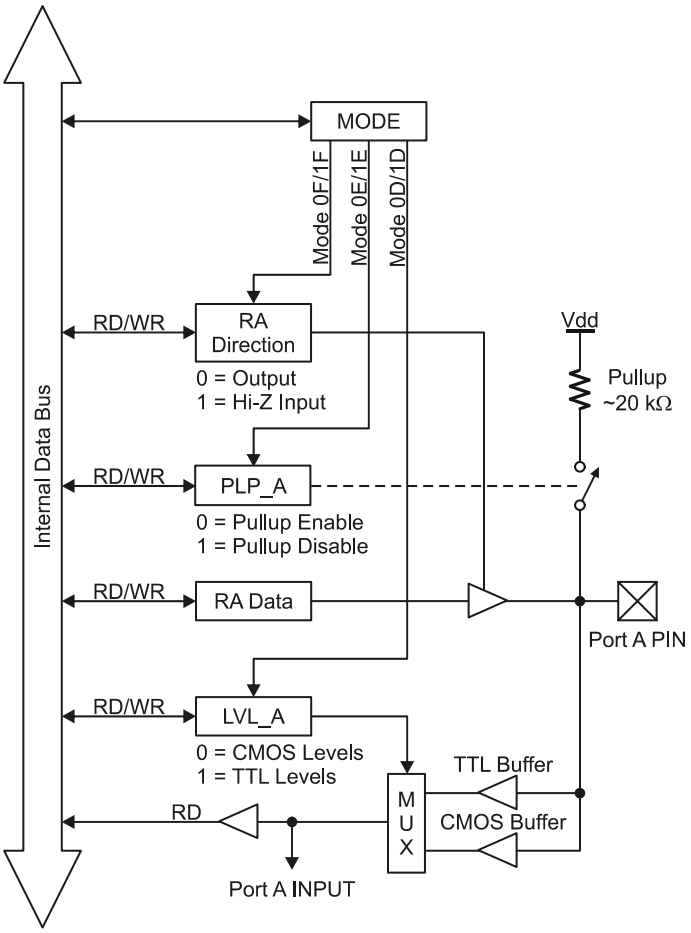


Figure 3-1

Port A Configuration

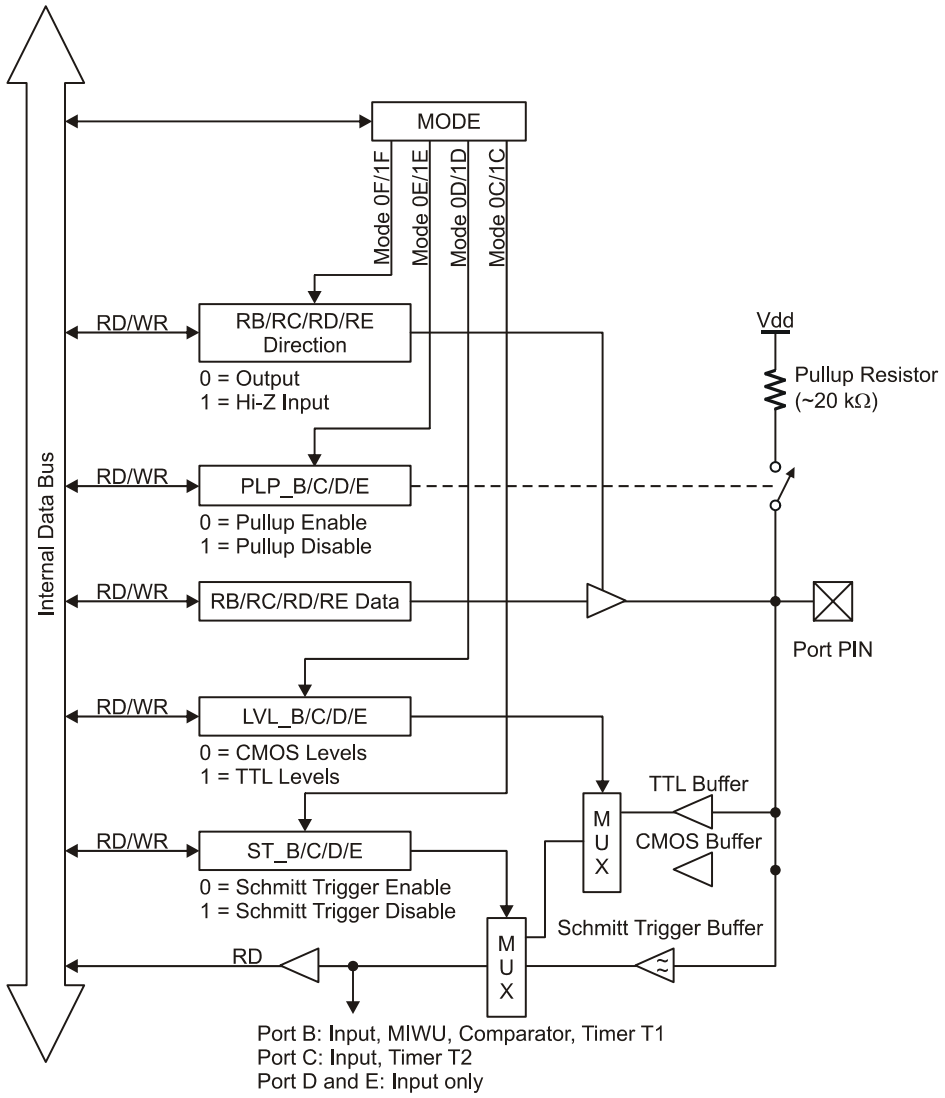


Figure 3-2: Port B, Port C, Port D, Port E Configuration

In the default device configuration, when a read is performed from a port bit position, the operation is actually reading the voltage level on the pin itself, and not the bit value stored in the port data register. This is true whether the pin is configured to operate as an input or an output. Therefore, with the pin configured to operate as an input, the data register contents have no effect on the value that you read. With the pin configured to operate as an output, what is read generally matches what has been written to the register. PORTRD of the T2CNT2 register determines how the device reads data from its I/O ports (Port A through Port E). Clear this bit to 0 to have the device read data from the port I/O pins directly. Set this bit to 1 to have the device read data from the port data registers. Under normal conditions, it should not matter which method you use to read the port data. However, if a port pin is configured as an output and an external circuit forces the pin to the opposite value, the value read from the port will depend on the reading mode used. Note that this control bit is not related to multi-function timers T1 and T2.

3.2. Read-Modify-Write Considerations

When two successive instructions are used on the same I/O port (except "mov Rx,W") with a very high clock rate, the "write" part of one instruction might not occur soon enough before the "read" part of the very next instruction, resulting in getting "old" data for the second instruction. To ensure predictable results, avoid using two successive read-modify-write instructions that access the same port data register if the clock rate is high or, insert 3 NOP instructions between the successive read-modify-write instructions (if SYNC bit in the FUSE register is enabled, 5 NOP instructions are required). For operating frequencies of 50 MHz or lower, if bit 7 of the T2CNTB (PORTRD) is set, the port reads data from the data register instead of port pins. In this case, the NOP instructions are not required.

3.3. Port Configuration

Each port pin offers the following configuration options:

- data direction
- input voltage levels (TTL or CMOS)
- pullup type (enable or disable)
- Schmitt trigger input (except for Port A)

Port B offers the additional option to use the port pins for the Multi-Input Wakeup/Interrupt function, the analog comparator function, or Timer T1 I/O. Port C offers the additional option to use the port pins for Timer T2 I/O. Port configuration is performed by writing to a set of control registers associated with the port. A special-purpose instruction is used to write these control registers:

```
mov !RA,W       ;move W to/from Port A
                ;control register
mov !RB,W       ;move W to/from Port B
                ;control register
mov !RC,W       ;move W to/from Port C
                ;control register
mov !RD,W       ;move W to/from Port D
                ;control register
mov !RE,W       ;move W to/from Port E
                ;control register
```

Each one of these instructions reads or writes a port control register for Port A, B, C, D, or E. There are multiple control registers for each port. To specify which one you want to access, you use another register called the MODE register.

3.3.1. MODE Register

The MODE register controls access to the port configuration registers and Timer T1/T2 control registers. Because the MODE register is not memory-mapped, it is accessed by the following special-purpose instructions:

```
mov M, #lit     ;move literal to lower 4-bits
                ;of MODE register
mov M,W         ;move W to lower 5-bits
                ;of MODE register
mov W,M         ;move MODE register to W
```

The value contained in the MODE register determines which port control register is accessed by the "mov !rx,W" instruction as indicated in Table 3-3. (The table also shows the timer control registers accessed according to the MODE register setting.) MODE register values not defined in the table are reserved for future expansion and should not be used. Upon power-up, the MODE register is initialized to 1Fh, which enables write access to the port direction control registers.

When bit 4 of the MODE register is 0 (the top half of Table 3-3), a "mov !rx,W" instruction moves the contents of the applicable control register into W. When bit 4 of the MODE register is 1 (the bottom half of Table 3-3), a "mov !rx,W" instruction moves the contents of W into the applicable control register. However, there are some exceptions to this. For the CMP_B and WKPND_B registers, the CPU does an exchange of data between W and the control register, regardless of the state of bit 4 in the MODE register. For the WKED_B and WKEN_B registers, the CPU moves the data from W to the control register, regardless of the state of bit 4 in the MODE register.

After a value is written to the MODE register, that setting remains in effect until it is changed by writing to the MODE register again. For example, you can write the value 1Eh to the MODE register just once, and then write to each of the five pullup configuration registers using the five "mov !rx,W" instructions.

Table 3-3: Mode Register Settings					
MODE Reg.	**Mov !RA,W**	**Mov !RB,W**	**Mov !RC,W**	**Mov !RD,W**	**Mov !RE,W**
00h		Read T1CPL	Read T2CPL		
01h		Read T1CPH	Read T2CPH		
02h		Read T1R2CML	Read T2R2CML		
03h		Read T1R2CMH	Read T2R2CMH		
04h		Read T1R1CML	Read T2R1CML		
05h		Read T1R1CMH	Read T2R1CMH		
06h		Read T1CNTB	Read T2CNTB		
07h		Read T1CNTA	Read T2CNTA		
08h		Exchange CMP_B with W			
09h		Exchange WKPND_B with W			
0Ah		Write WKED_B			
0Bh		Write $\overline{\text{WKEN\_B}}$			
0Ch		Read ST_B	Read ST_C	Read ST_D	Read ST_E
0Dh	Read LVL_A	Read LVL_B	Read LVL_C	Read LVL_D	Read LVL_E
0Eh	Read PLP_A	Read PLP_B	Read PLP_C	Read PLP_D	Read PLP_E
0Fh	Read RA Direction	Read RB Direction	Read RC Direction	Read RD Direction	Read RE Direction
10h		Clear Timer T1			
11h					
12h		Write T1R2CML	Write T2R2CML		
13h		Write T1R2CMH	Write T2R2CMH		
14h		Write T1R1CML	Write T2R1CML		
15h		Write T1R1CMH	Write T2R1CMH		
16h		Write T1CNTB	Write T2CNTB		
17h		Write T1CNTA	Write T2CNTA		
18h		Exchange CMP_B with W			
19h		Exchange WKPND_B with W			
1Ah		Write WKED_B			
1Bh		Write $\overline{\text{WKEN\_B}}$			
1Ch		Write ST_B	Write ST_C	Write ST_D	Write ST_E
1Dh	Write LVL_A	Write LVL_B	Write LVL_C	Write LVL_D	Write LVL_E
1Eh	Write PLP_A	Write PLP_B	Write PLP_C	Write PLP_D	Write PLP_E
1Fh	Write RA Direction	Write RB Direction	Write RC Direction	Write RD Direction	Write RE Direction

The following code example shows how to program the pullup control registers.

```
mov     W,#$1E  ;MODE=1Eh to write port pullup
mov     M, W    ;registers

mov     W,#$03  ;W = 0000 0011
mov     !RA,W   ;disable pullups for RA0 and RA1

mov     W,#$FF  ;W = 1111 1111
mov     !RB,W   ;disable all pullups for RB0-RB7

mov     W,#$00  ;W = 0000 0000
mov     !RC,W   ;enable all pullups for RC0-RC7
```

First the MODE register is loaded with 1Eh to select write access to the pullup control registers (PLP_A, PLP_B, and so on). Then the MOV !rx,W instructions are used to specify which port pins are to be connected to the internal pullup resistors. Setting a bit to 1 disconnects the corresponding pullup resistor, and clearing a bit to 0 connects the corresponding pullup resistor.

3.3.2. Port Configuration Registers

The port configuration registers that you control with the MOV !rx,W instruction operate as described below.

RA through RE Data Direction Registers (MODE=1Fh)

Each register bit sets the data direction for one port pin. Set the bit to 1 to make the pin operate as a high-impedance input. Clear the bit to 0 to make the pin operate as an output. Upon reset, the bit is set to 1.

PLP_A through PLP_E: Pullup Enable Registers (MODE=1Eh)

Each register bit determines whether an internal pullup resistor is connected to the pin. Set the bit to 1 to disconnect the pullup resistor or clear the bit to 0 to connect the pullup resistor. Upon reset, the bit is set to 1.

LVL_A through LVL_E: Input Level Registers (MODE=1Dh)

Each register bit determines the voltage levels sensed on the input port, either TTL or CMOS, when the Schmitt trigger option is disabled. Program each bit according to the type of device that is driving the port input pin. Set the bit to 1 for TTL or clear the bit to 0 for CMOS. Upon reset, the bit is set to 1. If $\overline{\text{SYNC}}$ is enabled in the FUSE register, port data must be read more than 2 cycles after a change to the input level mode or Schmitt Trigger mode (see Figure 3-2).

ST_B through ST_E: Schmitt Trigger Enable Registers (MODE=1Ch)

Each register bit determines whether the port input pin operates with a Schmitt trigger. Set the bit to 1 to disable Schmitt trigger operation and sense either TTL or CMOS voltage levels; or clear the bit to 0 to enable Schmitt trigger operation. Upon reset, the bit is set to 1. If $\overline{\text{SYNC}}$ is enabled in the FUSE register, port data must be read more than 2 cycles after a change to the input level mode or Schmitt Trigger mode (see Figure 3-2).

$\overline{\text{WKEN}}$_B: Wakeup Enable Register (MODE=1Bh)

Each register bit enables or disables the Multi-Input Wakeup/Interrupt (MIWU) function for the corresponding Port B input pin. Clear the bit to 0 to enable MIWU operation or set the bit to 1 to disable MIWU operation. Upon reset, the bit is set to 1. For more information on using the Multi-Input Wakeup/Interrupt function, see Section 8.0.

WKED_B: Wakeup Edge Register (MODE=1Ah)

Each register bit selects the edge sensitivity of the Port B input pin for MIWU operation. Clear the bit to 0 to sense rising (low-to-high) edges. Set the bit to 1 to sense falling (high-to-low) edges. Upon reset, the bit is set to 1.

WKPND_B: Wakeup Pending Flag Register (MODE=19h)

When you access the WKPND_B register using "mov !RB,W", the CPU exchanges the contents of W and WKPND_B. This feature lets you read the WKPND_B register contents while clearing the Wakeup Pending bits simultaneously. Each bit read from the WKPND_B register indicates the status of the corresponding MIWU pin. A bit set to 1 indicates that a valid edge has occurred on the corresponding MIWU pin, and has triggered a wakeup or interrupt. A bit cleared to 0 indicates that no valid edge has occurred on the MIWU pin.

CMP_B: Comparator Register (MODE=08h)

When you access the CMP_B register using MOV !RB,W, the CPU exchanges the contents of W and CMP_B. This feature lets you read the CMP_B register contents while writing a new value to the register. Clear bit 7 to enable operation of the comparator. Clear bit 6 to place the comparator result on the RB0 pin. Bit 0 is a result flag that is set to 1 when the voltage on RB2 (positive input) is greater than RB1 (negative input), or cleared to 0 otherwise. (For more information on using the comparator, see Section 12.0.)

3.3.3. Port Configuration Upon Power-Up

Upon power-up, all the port control registers are initialized to FFh. Thus, each port pin is configured to operate as a high-impedance input that senses TTL voltage levels, with no internal pullup resistor connected. The MODE register is initialized to 1Fh, which allows immediate write access to the data direction registers using the "MOV !rx,W" instruction.

4.0 SPECIAL-FUNCTION REGISTERS

The CPU uses a set of special-function registers to control operation of the device.

The CPU registers include an 8-bit working register (W), which serves as a pseudo accumulator. It holds the second operand of an instruction, receives the literal in immediate type instructions, and also can be program selected as the destination register.

A set of 31 file registers serves as the primary accumulator. One of these registers holds the first operand of an instruction and another can be program-selected as the destination register. The first 10 file registers include the Real-Time Clock/Counter register (RTCC), the lower eight bits of the 12-bit Program Counter (PC), the 8-bit STATUS register, five port control registers for Ports A through E, the 8-bit File Select Register (FSR), and INDF (used for indirect addressing).

The five low-order bits of the FSR register select one of the 31 file registers in the indirect addressing mode. Calling for the file register located at address 00h (INDF) in any of the file-oriented instructions selects indirect addressing, which uses the FSR register. It should be noted that the file register at address 00h is not a physically implemented register. The CPU also contains an 8 level, 12-bit hardware push/pop stack for subroutine linkage.

Table 4-1: Special-Function Register

Address	Name	Function
00h	INDF	Used for indirect addressing
01h	RTCC	Real Time Clock/Counter
02h	PC	Program Counter (low byte)
03h	STATUS	Holds status bits of ALU
04h	FSR	File Select Register
05h	RA	Port RA data register
06h	RB	Port RB data register
07h	RC	Port RC data register
08h	RD	Port RD data register
09h	RE	Port RE data register

4.1. PC Register (02h)

The PC register holds the lower eight bits of the program counter. It is accessible at run time to perform branch operations. The upper three bits are located in the STATUS register (PA2:0), bit 8 is not accessible.

4.2. STATUS Register (03h)

The STATUS register holds the arithmetic status of the ALU, the page select bits, and the reset state. The STATUS register is accessible during run time, except that bits PD and TO are read-only. It is recommended that only SETB and CLRB instructions be used on this register. Care should be exercised when writing to the STATUS register as the ALU status bits are updated upon completion of the write operation, possibly leaving the STATUS register with a result that is different than intended.

PA2	PA1	PA0	TO	PD	Z	DC	C
Bit 7							**Bit 0**

Bit 7-5: Program memory page select bits PA2:PA0

000 = Page 0 (000h - 1FFh)

001 = Page 1 (200h - 3FFh)

...

111 = Page 7 (E00h - FFFh)

Bit 4: Time Out bit, TO (Read Only)

1 = Set to 1 after power up and upon execution of CLRWDT or SLEEP instructions

0 = A watchdog time-out occurred

Bit 3: Power Down bit, PD (Read Only)

1= Set to a 1 after power up and upon execution of the CLR !WDT instruction

0 = Cleared to a '0' upon execution of SLEEP instruction

Bit 2: Zero bit, Z (affected by most logical, arithmetic, and data movement instructions

1 = Result of math operation is zero

0 = Result of math operation is non-zero Bit 1: Digit Carry bit, DC

After Addition:

1 = A carry from bit 3 occurred

0 = No carry from bit 3 occurred

After Subtraction:

1 = No borrow from bit 3 occurred

0 = A borrow from bit 3 occurred

Bit 0: Carry bit, C

After Addition:

1 = A carry from bit 7 of the result occurred

0 = No carry from bit 7 of the result occurred.

After Subtraction:

1 = No borrow from bit 7 of the result occurred

0 = A borrow from bit 7 of the result occurred

Rotate (RR or RL) Instructions:

The carry bit is loaded with the low or high order bit, respectively

When CF bit of the FUSEX register is cleared to 0, Carry bit works as input for ADD and SUB instructions.

4.3. OPTION Register

RTW	RTW_IE	RTS	RTE_ES	PSA	PS2	PS1	PS0
Bit 7							**Bit 0**

Bit 7: RTW RTCC/W register selection:
0 = Register 01h addresses W
1 = Register 01h addresses RTCC

Bit 6: RTE_IE RTCC interrupt enable:
0 = RTCC roll-over interrupt is enabled
1 = RTCC roll-over interrupt is disabled

Bit 5: RTS RTCC increment select:
0 = RTCC increments on internal instruction cycle
1 = RTCC increments upon transition on RTCC pin

Bit 4: RTE_ES RTCC edge select:
0 = RTCC increments on low-to-high transitions
1 = RTCC increments on high-to-low transitions

Bit 3: PSA Prescaler Assignment:
0 = Prescaler is assigned to RTCC, with divide rate determined by PS0 PS2 bits
1 = Prescaler is assigned to WDT, and divide rate on RTCC is 1:1

Bits 2-0: PS2-PS0 Prescaler divider (see Table 4-2)

Upon reset, all bits in the OPTION register are set to 1.

Table 4-2: Prescaler Divider Ratios

PS2, PS1, PS0	RTCC Divide Rate	Watchdog Timer Divide Rate
000	1:2	1:1
001	1:4	1:2
010	1:8	1:4
011	1:16	1:8
100	1:32	1:16
101	1:64	1:32
110	1:128	1:64
111	1:256	1:128

5.0 DEVICE CONFIGURATION AND ID REGISTERS

The SX device has two registers (FUSE, FUSEX) that control functions such as clock oscillator configuration. These registers are not programmable "on the fly" during normal device operation. Instead, the FUSE and FUSEX registers can only be accessed when the SX device is being programmed. The DEVICE ID register is a read only, hard-wired register, defined during the manufacturing process. Locations 1000h to 100Fh are allocated for user code ID.

5.1. FUSE Word (Read/program via Programming Command)

Unused	$\overline{\text{SYNC}}$	Unused	Unused	$\overline{\text{IRC}}$	DIV1/$\overline{\text{IFBD}}$	DIV0/FOSC2	XTLBUF_EN	$\overline{\text{CP}}$	WDTE	FOSC1	FOSC0
11	10	9	8	7	6	5	4	3	2	1	0

$\overline{\text{SYNC}}$ Synchronous input enable (this bit synchronizes the signal presented at the input pins to the internalclock through two internal flip-flops). Required to be enabled unless the transition on the input pin isnot close to the clock edge. If enabled, port data must be read more than 2 cycles after a change tothe input level mode or Schmitt Trigger mode (see Figure 3-2). SYNC is always enabled on RTCC.

0 = enabled

1 = disabled

$\overline{\text{IRC}}$ Internal RC oscillator enable

0 = enabled - OSC1 is pulled low by weak pulldown, OSC2 is pulled high by weak pullup

1 = disabled - OSC1 and OSC2 behave according to FOSC2:FOSC0

DIV1:DIV0 Internal RC oscillator divider (if $\overline{\text{IRC}}$ = 0)

00b = 4 MHz

01b = 1 MHz

10b = 128 KHz

11b = 32 KHz

$\overline{\text{IFBD}}$ Internal crystal/resonator oscillator feedback resistor (10MΩ)

0 = Internal feedback resistor disable (external feedback required for crystal/resonator oscillator)

1 = Internal feedback resistor enabled (valid only when $\overline{\text{IRC}}$ = 1, disabled when $\overline{\text{IRC}}$= 0)

XTLBUF_EN Crystal Buffer enable (disable when not using a crystal to reduce Idd)

0 = Crystal Buffer disabled (required if not using crystal/resonator oscillator)

1 = Crystal Buffer enabled

$\overline{\text{CP}}$ Code protect enable

0 = enabled (FUSE, code, and ID memories read back as scrambled data, programming disabled)

1 = disabled (FUSE, code, and ID memories can be read normally)

WDTE Watchdog timer enable

0 = disabled

1 = enabled

FOSC2:FOSC0 External oscillator configuration (valid when $\overline{\text{IRC}}$ = 1, lower settings are recommended for lower power consumption):

000b = LP1 - low power crystal (32KHz)

001b = LP2 - low power crystal/resonator (32KHz - 1MHz)

010b = XT1 - normal crystal/resonator (32KHz - 1MHz)

011b = XT2 - normal crystal/resonator (1MHz - 8MHz)

100b = HS1 - high speed crystal/resonator (1MHz - 20MHz)

101b = HS2 - high speed crystal/resonator (1MHz - 50MHz)

110b = HS3 - high speed crystal/resonator (1MHz - 75MHz)

111b = External RC network - OSC2 is pulled high by a weak pullup (no CLKOUT output)

5.2. FUSEX Word (Read/program via Programming Command)

IRCTRIM2	$\overline{\text{SLEEPCLK}}$	IRCTRIM1:IRCTRIM0	Unused	$\overline{\text{CF}}$	BOR1:BOR0	BORTR1:BORTR0	DRT1:DRT0
11	**10**	**9 8**	**7**	**6**	**5 4**	**3 2**	**1 0**

IRCTRIM2: IRCTRIM0 — Internal RC Oscillator Trim. This 3-bit field adjusts the operation of the internal RC oscillator to make it operate within the target frequency range of typically 4.0 MHz. Parts are shipped from the factory untrimmed. The device relies on the programming tool to provide trimming.

100b = maximum frequency

111b = typical

011b = minimum frequency

$\overline{\text{SLEEPCLK}}$ — Sleep Clock Disable.

0 = enable operation of the crystal/resonator clock during power down mode (to allow fast start up).

1 = disable crystal/resonator clock operation during power down mode (to reduce power consumption).

$\overline{\text{CF}}$ — Carry Flag ADD/SUB enable

0 = carry bit input to ADD and SUB instructions.

1 = ADD and SUB without carry

BOR1: BOR0 — Sets the Brown Out Reset threshold voltage

00b = 4.2 V

01b = 2.6 V

10b = 2.2 V

11b = BOR disabled

BORTR1: BORTR0 — Brown-Out trim bits (parts are shipped out of factory untrimmed).

01b = minimum threshold voltage

00b = LOW

11b = HIGH

10b = maximum threshold voltage

DRT1:DRT0 — Delay Reset Timer (DRT) timeout period. Specifies the time from de-assertion of reset to start code execution.

10b = 0.60 µsec

11b = 18 msec

00b = 60 msec

01b = 960 msec

5.3. DEVICE ID Word

(Hard-Wired Read-Only Via Programming Command) - Part ID Code

0	0	0	0	0	0	0	0	0	0	1	0
11	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

5.4. User Code ID

Locations 1000h to 100Fh are allocated for user code ID.

6.0 MEMORY ORGANIZATION

6.1. Program Memory

The program memory is organized as 4K, 12-bit wide words. The program memory words are addressed sequentially by a binary program counter. Upon reset, the program counter is initialized with 0FFFh. If there is no branch operation, it will increment to the maximum value possible for the device and roll over and begin again. Internally, the program memory has a semi-transparent page structure. A page is composed of 512 contiguous program memory words. The lower nine bits of the program counter are zeros at the first address of a page and ones at the last address of a page. This page structure has no effect on the program counter. The program counter will freely increment through the page boundaries.

6.1.1. Program Counter

The program counter contains the 12-bit address of the instruction to be executed. The lower eight bits of the program counter are contained in the PC register (02h), and the three upper bits are specified by the STATUS register (PA0, PA1, PA2). Bit 8 is not accessible. Changing the STATUS bits is necessary to cause jumps and subroutine calls across program memory page boundaries. Prior to the execution of a branch operation, the user program must initialize the upper bits of the STATUS register to cause a branch to the desired page. An alternative method is to use the PAGE instruction, which automatically causes subsequent branch instructions to vector to the desired page, based on the value specified in the operand field.

6.1.2. Subroutine Stack

The subroutine stack consists of eight 12-bit save registers. A physical transfer of register contents from the program counter to the stack or vice versa, and within the stack, occurs on all operations affecting the stack, primarily calls and returns. The stack is physically and logically separate from data RAM. The program cannot read or write the stack.

6.2. Data Memory

The data memory is a RAM-based register set consisting of 262 general-purpose registers and nine special-purpose registers. All of these registers are eight bits wide. The data memory is organized into 16 banks, designated Bank 0 through Bank F, each containing 16 registers, plus an additional bank of 16 “global” registers. Because the registers are organized into banks or “files,” these memory-mapped registers are called “file registers.”

6.2.1. Addressing Modes/FSR

Each SX instruction that accesses a data memory register contains a 5-bit field in the instruction opcode that specifies the register to be accessed. The abbreviation “fr” (file register) represents the 5-bit register address designator. For example, the instruction description “mov fr,W” means that a 5-bit value or label must be substituted for “fr” in the instruction, such as “mov $0F,W” (to move the contents of the working register W into file reg ister 0Fh).

There are three different addressing modes, called the indirect, direct, and semi-direct modes. The addressing mode used for register access depends on the 5-bit “fr” value used in the instruction:

- indirect mode: fr = 00h
- direct mode (fr bit 4 = 0): fr = 01h through 0Fh
- semi-direct mode (fr bit 4 = 1): fr = 10h through 1Fh

Figure 6-1 illustrates the data memory addressing scheme. For indirect addressing (fr=00), the File Select Register (FSR) specifies the register to be accessed. FSR is an 8bit, memory-mapped register (at address 04h) which serves as an 8-bit pointer into data memory for indirect addressing. In this mode, the global register bank and Bank 1 through Bank F are accessible. Bank 0 is not accessible.

For direct addressing (fr=01-0F), the value of “fr” itself specifies the register to be accessed, and the FSR register is ignored. For this addressing mode, only the global register bank is accessible. To gain access to any other bank, you must use either indirect or semi-direct addressing.

For semi-direct addressing (fr=10-1F), the bank number is selected by the four high-order bits of FSR, and the register within that bank is selected by the four low-order bits of “fr.” In other words, the register address is obtained by combining the four high-order bits of FSR with the four low-order bits of “fr”. In this addressing mode, the low-order bits of FSR are ignored. Bank 0 through Bank F are accessible, but the global register bank is not accessible.

Figure 6-1 shows how register addressing works in the indirect, direct, and semi-direct modes. The 16 global registers are always accessible by direct addressing, regardless of what is contained in the FSR register. The global registers are also accessible with indirect addressing, but they are not accessible with semi-direct addressing. Of the 16 global registers, nine are special-purpose registers (RTCC, PC, STATUS, and so on), and six are general-purpose registers. Location 00 is used for indirect addressing (INDF). All of the registers in Bank 0 though Bank F are general-purpose registers. To change the contents of the FSR register, the program can either write an eight-bit value to the FSR register or use the “bank” instruction. The “bank” instruction writes bits 4, 5, and 6 in the FSR register. Bit 7 of FSR is used to select the upper or lower “bank” of memory banks. Thus, to change from one upper bank to another, only a single “bank” instruction is required. To change from one upper bank to a lower bank, the “bank” instruction must be followed by “setb FSR.7”.

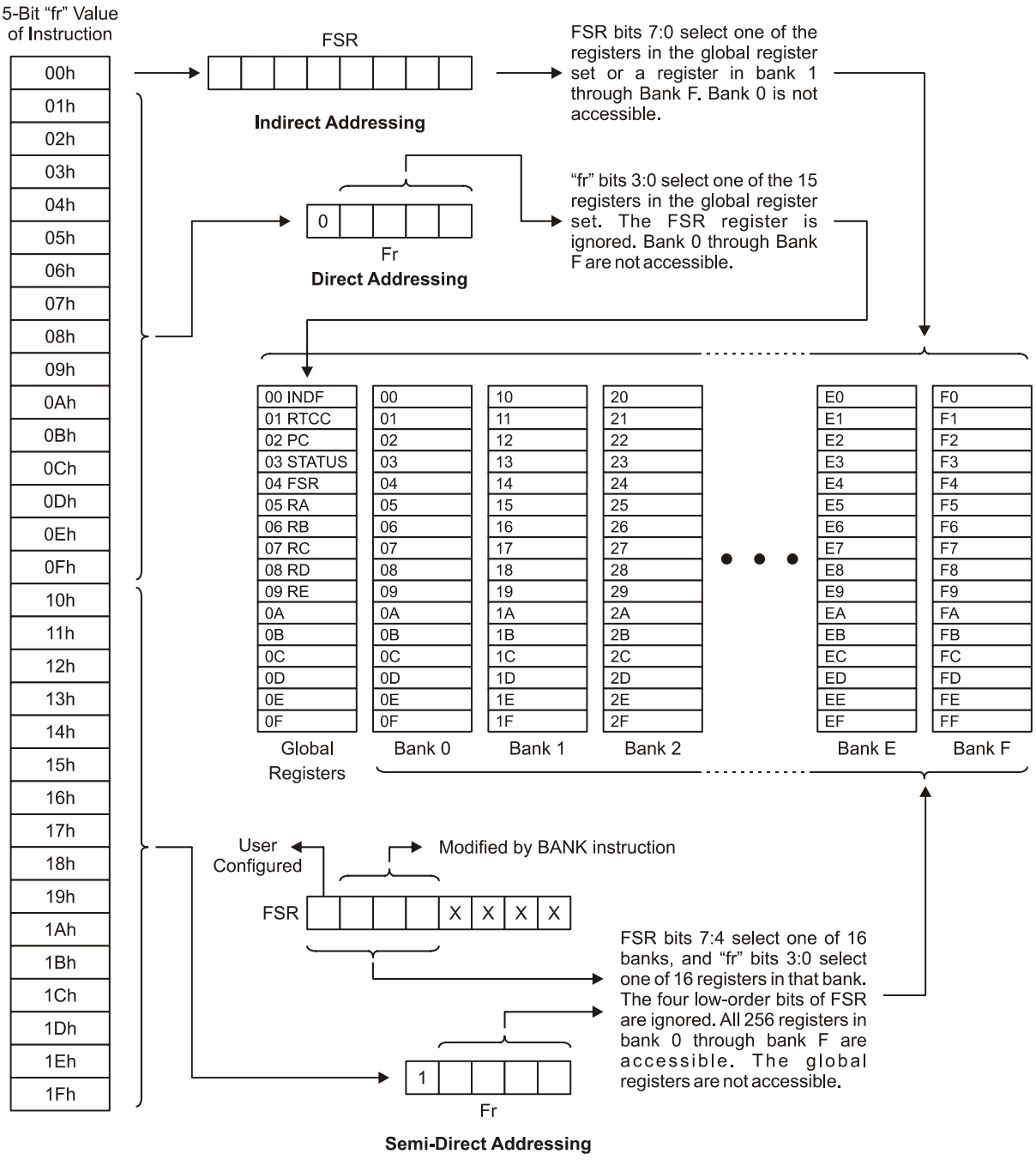


Figure 6-1: Register Access Modes

6.2.2. Register Access Examples

Here is an example of an instruction that uses direct addressing:

```
inc     $0F    ;increment file register 0Fh
```

This instruction increments the contents of file register 0Fh in the global register bank. It does not matter what is contained in the FSR register.

To gain access to any register outside of the global register bank, it is necessary to use semi-direct or indirect addressing. In that case, you need to make sure that the FSR register contains the correct value for accessing the desired bank. Here are 2 examples that use semi-direct addressing:

```
mov     W,#$F0  ;load W with F0h
mov     FSR,W   ;load W into FSR (Bank F)
inc     $1F     ;increment file register FFh
```

Or, to access bank 0,

```
mov     W,#$00  ;load W with 00h
mov     FSR,W   ;load W into FSR (Bank 0)
inc     $1F     ;increment file register 0Fh
```

In these examples, "FSR" is a label that represents the value 04h, which is the address of the FSR register in the global register bank. Note that the FSR register is itself a memory-mapped global register, which is always accessible using direct addressing.

The "banked" data memory is divided into upper and lower blocks, each consisting of 8 banks of data memory. The range for the lower block is from $00 to $7F, while the range for the upper block is from $80 to $FF. Bit 7 of the FSR is used to select the upper or lower block. The BANK instruction is used to select the bank within that block.

To use the "bank" instruction, in the syntax of the assembly language, you specify an 8-bit value that corresponds to the desired bank number. The assembler encodes bits 4, 5, and 6 of the specified value into the instruction opcode and ignores bit 7 and the low-order bits. For example, if another lower bank was being used to increment file register 2Fh, you could use the following instructions:

```
bank    $20     ;select Bank 2 in FSR
inc     $1F     ;increment register 2F
```

Note that the "bank" instruction only modifies bits 4, 5, and 6 the FSR register. Therefore, to change from a lower block to an upper block bank, the "bank" instruction will not work. Instead, you need to write the whole FSR register using code such as the following:

```
mov     W,#$80  ;load W with 80h
mov     FSR,W   ;select Bank 8 in FSR
```

Another approach is to set bit 7 of the FSR register individually after the "bank" instruction to address an upper block bank.

```
bank    $80     ;set bits in 4, 5, and 6 FSR
setb    FSR.7   ;select Bank 8 in FSR
```

To change from an upper block to a lower block bank, bit 7 of FSR must be cleared.

With indirect addressing, you specify the full 8-bit address of the register using FSR as a pointer. This addressing mode provides the flexibility to access different registers or multiple registers using the same instruction in the program.

You invoke indirect addressing by using fr=00h. For example:

```
mov     W,#$F5  ;load W with F5h
mov     $04,W   ;move value F5h into FSR
mov     W,#$01  ;load W with 01h
mov     $00,W   ;move value 01h into register F5h
```

In the second "mov" instruction, FSR is loaded with the desired 8-bit register address. In the fourth "mov" instruction, fr = 00, so the device looks at FSR and moves the result to the register addressed by FSR, which is the register at F5h (Bank F, register number 5).

A practical example that uses indirect addressing is the following program, which clears the upper eight registers in the global register bank and the upper 8 registers in all banks from Bank 1 through Bank F:

```
clr     FSR     ;clear FSR to 00h (at 04h)
:loop   setb    FSR.3   ;set FSR bit 3
clr     $00     ;clear register pointed to by FSR
incsz   FSR     ;increment FSR and test
                ;skip jmp if 00h
jmp     :loop   ;jump back and clear next reg.
```

This program initially clears FSR to 00h. At the beginning of the loop, it sets bit 3 of FSR so that it starts at 08h. The "clr $00" instruction clears the register pointed to by FSR (initially, the file register at 08h in the global register bank). Then the program increments FSR and clears consecutive file registers, always in the upper half of each bank: (08h, 09h, 0Ah... 0Fh, 18h, 19h... FFh). The loop ends when FSR wraps back to 00h.

For addresses from 01h through 0Fh, the global register bank is accessed. For higher addresses, Bank 1 through Bank F are accessed. This program does not affect Bank 0, which is not accessible in the indirect addressing mode.

7.0 POWER DOWN MODE

The power down mode is entered by executing the SLEEP instruction.

In power down mode, only the Watchdog Timer (WDT) and SLEEPCLOCK are active, if enabled. The operation clock can be enabled or disabled during this mode, by using the SLEEPCLK bit of the FUSEX register. If the Watchdog Timer is enabled, upon execution of the SLEEP instruction, the Watchdog Timer is cleared, the TO (time out) bit is set in the STATUS register, and the PD (power down) bit is cleared in the STATUS register.

There are three different ways to exit from the power down mode:

1. A timer overflow signal from the Watchdog Timer (WDT).
2. A valid transition on any of the Multi-Input Wakeup pins (Port B pins).
3. An external reset input on the MCLR pin.

The states of registers (upon wakeup) are described in Section 15.0.

To achieve the lowest possible power consumption, the Watchdog Timer should be disabled (the sleep clock should be disabled) and the device should exit the power down mode through the (Multi-Input Wakeup) MIWU pins or an external reset. In addition, the SLEEPCLOCK should be disabled during the power down mode.

Bit 11 of the FUSEX can be used to enable (clear bit to 0) the clock operation during the power down mode (to allow fast clock start-up upon exiting the power down mode).

7.1. Multi-Input Wakeup

Multi-Input Wakeup is one way of causing the device to exit the power down mode. Port B is used to support this feature. The WKEN_B register (Wakeup Enable Register) allows any Port B pin or combination of pins to cause the wakeup. Clearing a bit in the WKEN_B register enables the wakeup on the corresponding Port B pin. If multi-input wakeup is selected to cause a wakeup, the trigger condition on the selected pin can be either rising edge (low to high) or falling edge (high to low). The WKED_B register (Wakeup Edge Select) selects the desired transition edge. Setting a bit in the WKED_B register selects the falling edge on the corresponding Port B. Resetting the bit selects the rising edge. The WKEN_B and WKED_B registers are set to FFh upon reset.

Once a valid transition occurs on the selected pin, the WKPND_B register (Wakeup Pending Register) latches the transition in the corresponding bit position. A logic ‘1’ indicates the occurrence of the selected trigger edge on the corresponding Port B pin. The WKPND_B comes up with undefined value upon reset. The user program must clear the WKPND_B register prior to enabling the interrupt.

Upon exiting the power down mode, the Multi-Input Wakeup logic causes program counter to branch to the maximum program memory address (same as reset).

Figure 7-1 shows the Multi-Input Wakeup block diagram.

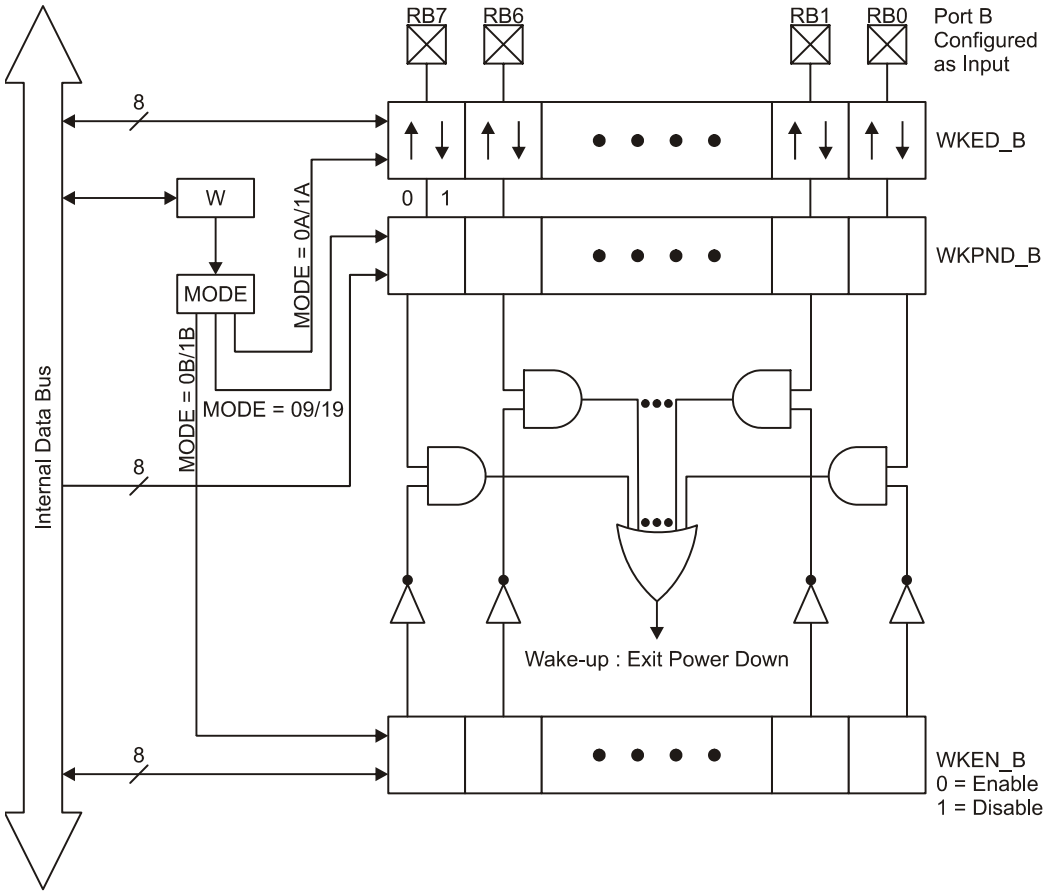


Figure 7-1: Multi-Input Wakeup Block Diagram

7.2. Port B MIWU/Interrupt Configuration

The WKPND_B register comes up with an unknown value upon reset. The user program must clear the register prior to enabling the wake-up condition or interrupts. The proper initialization sequence is:

Select the desired edge (through WKED_B register).

Clear the WKPND_B register.

Enable the Wakeup condition (through WKEN_B register).

Below is an example of how to read the WKPND_B register to determine which Port B pin caused the wakeup or interrupt, and to clear the WKPND_B register:

```
mov     W, #$19 ;prepare to exchange WKPND_B
                ;with W (can also use $09)
        mov M,W
        clr W
mov     !RB,W   ;W contains WKPND_B
                ;contents of W exchanged
                ;with contents of WKPND_B
```

The final "mov" instruction in this example performs an exchange of data between the working register (W) and the WKPND_B register. This exchange occurs only with accesses to the WKPND_B and CMP_B registers. Otherwise, the "mov" instruction does not perform an exchange, but only moves data from the source to the destination. Here is an example of a program segment that configures the RB0, RB1, and RB2 pins to operate as Multi-Input Wakeup/Interrupt pins, sensitive to falling edges:

```
mov     W,#$1F  ;prepare to write port data
mov     M,W     ;direction registers
mov     W,#$07  ;load W with the value 07h
mov     RB,W    ;configure RB0-RB2 to be inputs

mov     W,#$1A  ;prepare to write WKED_B
mov     M,W     ;(edge) register
mov     W,#$07  ;load W with the value 07h
mov     RB,W    ;configure RB0-RB2 to sense
                ;falling edges

mov     W,#$19  ;prepare to access WKPND_B
mov     M,W     ;(pending) register
mov     W,#$00  ;clear W
mov     !RB,W   ;clear all wakeup pending flags

mov     W,#$1B  ;prepare to write WKEN_B (enable)
mov     M,W     ;register
mov     W,#$F8  ;load W with the value F8h
mov     !RB,W   ;enable RB0-RB2 to operate as
                ;wakeup inputs
```

To prevent false interrupts, the enabling step (clearing bits in WKEN_B) should be done as the last step in a sequence of Port B configuration steps.

After this program segment is executed, the device can receive interrupts on the RB0, RB1, and RB2 pins. If the device is put into the power down mode (by executing a SLEEP instruction), the device can then receive wakeup signals on those same pins.

8.0 INTERRUPT SUPPORT

The device supports both internal and external maskable interrupts. The internal interrupt is generated as a result of the RTCC rolling over from FFh to 00h. This interrupt source has an associated enable bit located in the OPTION register and pending flag bit in the Timer T1 Control B register. In addition, timers T1 and T2 each have three interrupt sources associated with counter overflow, compare match, and input capture.

Port B provides the source for eight external software selectable, edge sensitive interrupts, when the device is not in the power down mode. These interrupt sources share logic with the Multi-Input Wakeup circuitry. The WKEN_B register allows interrupt from Port B to be individually enabled or disabled.

Clearing a bit in the WKEN_B register enables the interrupt on the corresponding Port B pin. The WKED_B selects the transition edge to be either positive or negative. The WKEN_B and WKED_B registers are set to FFh upon reset. Setting a bit in the WKED_B register selects the falling edge while clearing the bit selects the rising edge on the corresponding Port B pin.

The WKPND_B register serves as the external interrupt pending register.

The WKPND_B register comes up with a random value upon reset. The user program must clear the WKPND_B register prior to enabling the interrupt.

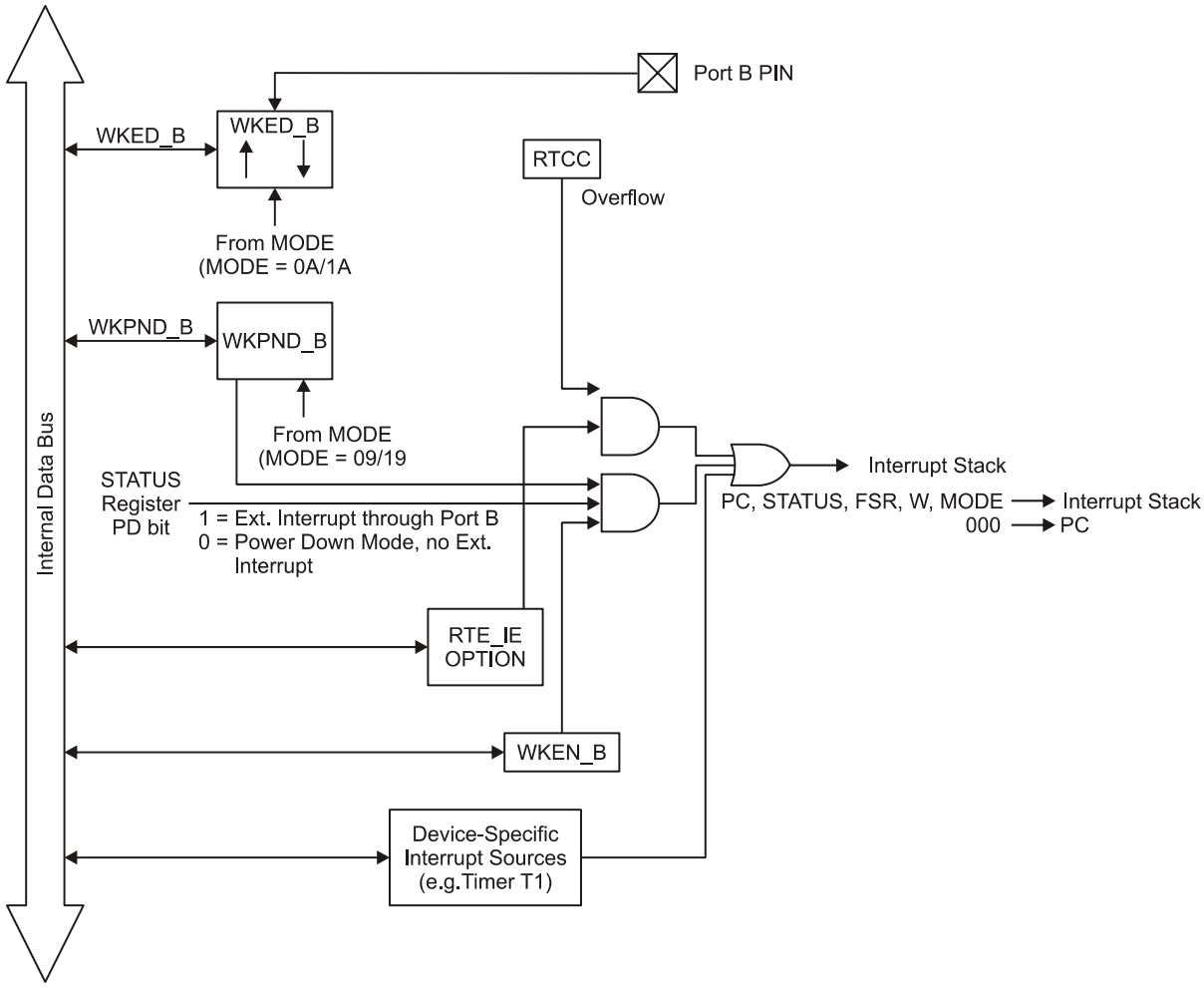


Figure 8-1: Interrupt Structure

All interrupts are global in nature; that is, no interrupt has priority over another. Interrupts are handled sequentially. Figure 8-2 shows the interrupt processing sequence. Once an interrupt is acknowledged, all subsequent interrupts are disabled until return from servicing the current interrupt. The PC is pushed onto the single level interrupt stack, and the contents of the FSR, STATUS, MODE, and W registers are saved in their corresponding shadow registers. The status bits PA2, PA1, and PA0 are cleared after STATUS has been saved in its shadow register. The interrupt logic has its own single-level stack and is not part of the CALL subroutine stack. The vector for the interrupt service routine is address 0.

Once in the interrupt service routine, the user program must poll all interrupt pending bits to determine the source of the interrupt. The interrupt service routine should clear the corresponding interrupt pending flag. Normally it is a requirement for the user program to process every interrupt without missing any. To ensure this, the longest path through the interrupt routine must take less time than the shortest possible delay between interrupts.

Using more than one interrupt, such as multiple external interrupts or both RTCC and external interrupts, can result in missed or, at best, jittery interrupt handling should one occur during the processing of another. When handling external interrupts, the interrupt routine should clear at least one pending register bit. The bit that is cleared should represent the interrupt being handled in order for the next interrupt to trigger.

Upon return from the interrupt service routine, the contents of PC, FSR, STATUS, MODE, and W registers are restored from their corresponding shadow registers. The interrupt service routine should end with instructions such as RETI or RETIW. RETI pops the interrupt stack and the special shadow registers used for storing W, STATUS, MODE, and FSR (preserved during interrupt handling). RETIW behaves like RETI but also adds W to RTCC. The interrupt return instruction enables the global interrupts.

If a MIWU interrupt occurs during a pre-existing interrupt service routine, the MIWU interrupt flag is set immediately, and the MIWU interrupt is serviced upon completion of the pre-existing interrupt service routine.

Timer interrupt will occur only if not in the ISR when the interrupt occurs.

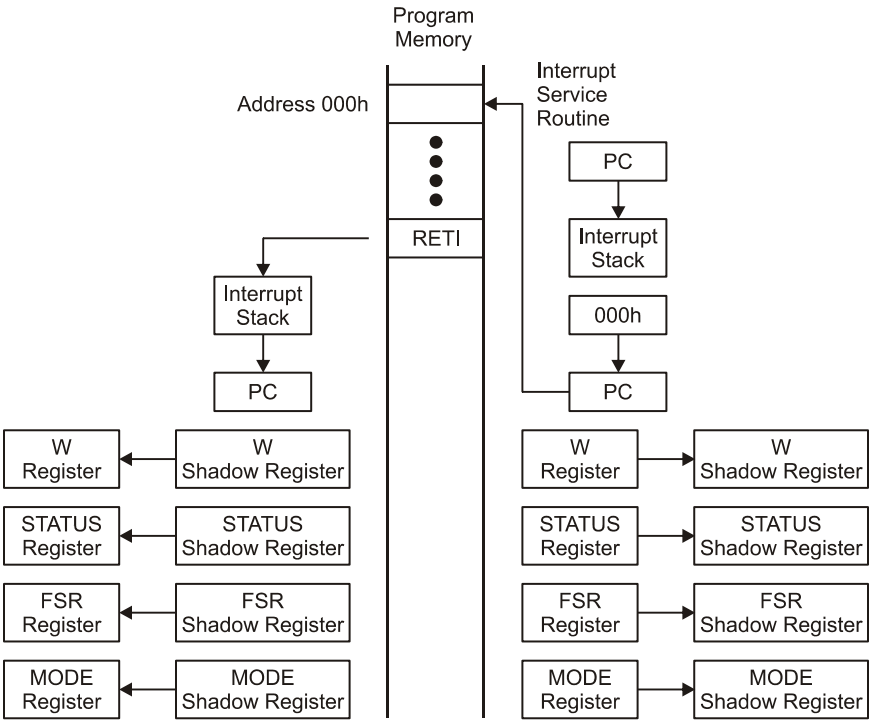


Figure 8-2: Interrupt Processing

Note: The interrupt logic has its own single-level stack and is not part of the CALL subroutine stack.

9.0 OSCILLATOR CIRCUITS

The device supports several user-selectable oscillator modes. The oscillator modes are selected by programming the appropriate values into the FUSE Word register. These are the different oscillator modes offered:

- LP: Low Power Crystal
- XT: Crystal/Resonator
- HS: High Speed Crystal/Resonator/Clock Oscillator
- RC: External Resistor/Capacitor
  Internal Resistor/Capacitor

9.1. XT, LP or HS Modes

In XT, LP or HS, modes, you can use either an external resonator circuit or an external clock source as the device clock.

To use an external resonator circuit, connect a crystal or ceramic resonator to the OSC1/CLKIN and OSC2/CLKOUT pins according to the circuit configuration shown in Figure 9-1. A parallel resonant crystal type is recommended. Use of a series resonant crystal is not recommended. Table 9-1 shows the recommended external components associated with a crystal-based oscillator. Table 9-2 shows the recommended external component values for a resonator-based oscillator.

Bits 5, 1 and 0 of the FUSE register (FOSC2:FOSC0) are used to configure the different external resonator/crystal oscillator modes. These bits allow the selection of the appropriate gain setting for the internal driver to match the desired operating frequency. If the XT, LP, or HS mode is selected, the OSC1/CLKIN pin can be driven by an external clock source rather than a resonator network, as long as the clock signal meets the specified duty cycle, rise and fall times, and input levels (Figure 9-2). In this case, the OSC2/CLKOUT pin should be left open.

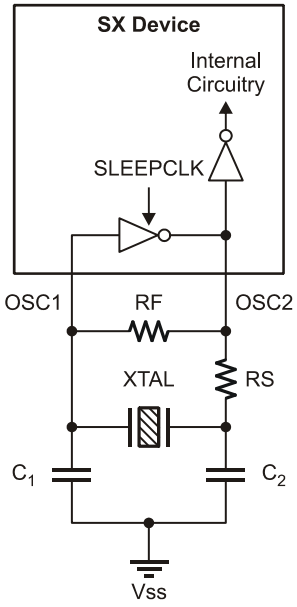


Figure 9-1: Crystal Operation
(or Ceramic Resonator)
(HS, XT or LP OSC Configuration)

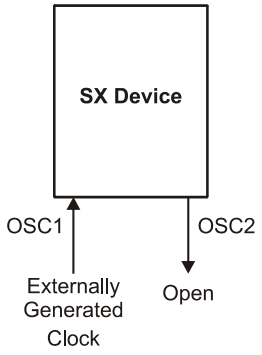


Figure 9-2: External Clock Input Operation
(HS, XT or LP OSC Configuration)

Table 9-1: External Component Selection for Crystal Oscillator (V_{dd} = 5.0 V)						
FOSC2:FOSC0	**Crystal Frequency**	**Setting Symbol**	**C1 (pF)**	**C2 (pF)**	**R_F (M)**	**Rs (Ω)**
011	4 MHz	XT2	33	56	1	0
011	8 MHz	XT2	22	56	1	0
100	20 MHz	HS1	22	33	1	0
101	32 MHz	HS2	15	47	1	0
101	50 MHz	HS2	15	33	1	0

Table 9-2: External Component Selection for Murata Ceramic Resonators (V_{dd} = 5.0 V)						
FOSC2:FOSC0	**Frequency**	**Resonator Part Number**	**C1**	**C2**	**R_F**	**R_S**
011	4 MHz	CSA4.00MG	30 pF		1 MΩ	0 Ω
011	4 MHz	CST4.00MGW	Internal (30 pF)	Internal (30 pF)	1 MΩ	0 Ω
011	4 MHz	CSTCC4.00G0H6	Internal (47 pF)	Internal (47 pF)	1 MΩ	0 Ω
011	8 MHz	CSA8.00MTZ	30 pF	30 pF	1 MΩ	0 Ω
011	8 MHz	CST8.00MTW	Internal (30 pF)	Internal (30 pF)	1 MΩ	0 Ω
011	8 MHz	CSTCC8.00MG0H6	Internal (47 pF)	Internal (47 pF)	1 MΩ	0 Ω
011	20 MHz	CSA20.00MXZ040	5 pF	5 pF	1 MΩ	0 Ω
011	20 MHz	CST20.00MXW0H1	Internal (5 pF)	Internal (5 pF)	1 MΩ	0 Ω
011	20 MHz	CSACV20.00MXJ040	5 pF	5 pF	22 kΩ	0 Ω
011	20 MHz	CSTCV20.00MXJ0H1	Internal (5 pF)	Internal (5 pF)	22 kΩ	0 Ω
100	33 MHz	CSA33.00MXJ040	5 pF	5 pF	1 MΩ	0 Ω
100	33 MHz	CST33.00MXW040	Internal (5 pF)	Internal (5 pF)	1 MΩ	0 Ω
100	33 MHz	CSACV33.00MXJ040	5 pF	5 pF	1 MΩ	0 Ω
100	33 MHz	CSTCV33.00MXJ040	Internal (15 pF)	Internal (15 pF)	1 MΩ	0 Ω
101	50 MHz	CSA50.00MXZ040	15 pF	15 pF	10 kΩ	0 Ω
101	50 MHz	CST50.00MXW0H3	Internal (15 pF)	Internal (15 pF)	10 kΩ	0 Ω
101	50 MHz	CSACV50.00MXJ040	15 pF	15 pF	10 kΩ	0 Ω
101	50 MHz	CSTCV50.00MXJ0H3	Internal (15 pF)	Internal (15 pF)	10 kΩ	0 Ω

Table 9-3: Clock Devices Available through Parallax Inc.				
Parallax Stock#	**Frequency**	**Device Type**	**Package**	**Manufacturer/Part #**
250-04050	4 MHz	Ceramic resonator	3-pin SIP	Murata CSTS0400MG03
250-14050	4 MHz	Ceramic resonator	SMT	Jiankang ZZTTC4.0MG
250-02060	20 Mhz	Ceramic resonator	3-pin SIP	Murata CST20.00MXW040
250-12060	20 MHz	Ceramic resonator	SMT	Transko CR3731M-20.000MHz
250-05060	50 MHz	Ceramic resonator	3-pin SIP	Murata CSTLS50M0X51-B0
250-15060	50 MHz	Ceramic resonator	SMT	Murata CSTCV50.00MXJ040-TC20
252-00005	75 MHz	TTL Oscillator	8-pin DIP half-size	Transko SX0550HT00ET

9.2. 75 MHz Operation

It is a good engineering practice to design your system to operate as conservatively as possible; that is as slowly as possible. However, some applications require a high clock rate and there is no way around it. Consider also that since the physical size of the elements within ceramic resonators vary inversely with the operational frequency, you will not find a great selection of resonators designed to operate over 50MHz.

To aide our customers who need to exploit the full-speed capabilities of the SX, we have specified a custom TTL oscillator that performs well throughout the industrial temperature range. Figure 9-3 depicts how the SX is used with the Transko 75MHz TTL oscillator.

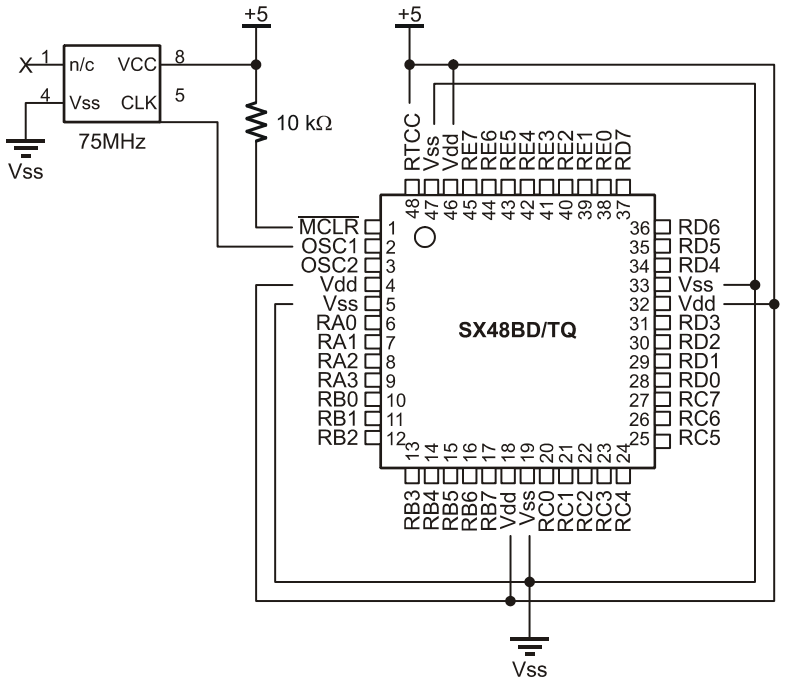


Figure 9-3: SX48BD with 75 MHz TTL Oscillator

9.3. External RC Mode

The external RC oscillator mode provides a cost-effective approach for applications that do not require a precise operating frequency. In this mode, the RC oscillator frequency is a function of the supply voltage, the resistor (R) and capacitor (C) values, and the operating temperature. In addition, the oscillator frequency will vary from unit to unit due to normal manufacturing process variations. Furthermore, the difference in lead frame capacitance between package types also affects the oscillation frequency, especially for low C values. The external R and C component tolerances contribute to oscillator frequency variation as well.

Figure 9-4 shows the external RC connection diagram. The recommended R value is from 3 kΩ to 100 kΩ. For R values below 2.2 kΩ, the oscillator may become unstable, or may stop completely. For very high R values (such as 1 MΩ), the oscillator becomes sensitive to noise, humidity, and leakage.

Although the oscillator will operate with no external capacitor (C = 0 pF), it is recommended that you use values above 20 pF for noise immunity and stability. With no or small external capacitance, the oscillation frequency can vary significantly due to variation in PCB trace or package lead frame capacitances.

9.4. Internal RC Mode

The internal RC mode uses an internal oscillator, so the device does not need any external components. At 4 MHz, the internal oscillator provides typically +/-8% accuracy over the allowed temperature range. The internal clock frequency can be divided down to provide one of eight lower-frequency choices by selecting the desired value in the FUSE Word register. The frequency range is from 31.25 KHz to 4 MHz. The default operating frequency of the internal RC oscillator may not be 4 MHz. This is due to the fact that the SX device requires trimming to obtain 4 MHz operation. The parts shipped out of the factory are not trimmed. The device relies on the programming tool provided by the third party vendors to support trimming.

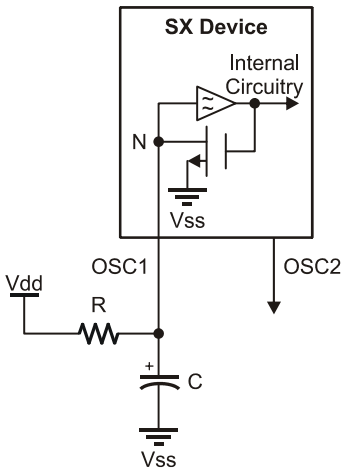


Figure 9-4: RC Oscillator Mode

10.0 REAL TIME CLOCK/COUNTER (RTCC)/WATCHDOG TIMER

The device contains an 8-bit Real Time Clock/Counter (RTCC) and an 8-bit Watchdog Timer (WDT). An 8-bit programmable prescaler extends the RTCC to 16 bits. If the prescaler is not used for the RTCC, it can serve as a postscaler for the Watchdog Timer. Figure 10-1 shows the RTCC and WDT block diagram.

10.1. RTCC

RTCC is an 8-bit real-time timer that is incremented once by the internal instruction cycle clock or from a transition on the RTCC pin. The on-board prescaler can be used to extend the RTCC counter to 16 bits.

To select the internal clock source, bit 5 of the OPTION register should be cleared. In this mode, RTCC is incremented at each instruction cycle unless the prescaler is selected to increment the counter.

To select the external clock source, bit 5 of the OPTION register must be set. In this mode, the RTCC pin is sampled on each rising edge of the OSC1 pin (the signal frequency at the RTCC pin must be less half the frequency on OSC1). By using bit 4 of the OPTION register, the transition can be programmed to be either a falling edge or rising edge. Setting the control bit selects the falling edge to increment the counter. Clearing the bit selects the rising edge.

The RTCC generates an interrupt (if enabled) as a result of an RTCC rollover from FFh to 00h. Bit 7 of the Timer T1 Control B register is an interrupt pending flag (RTCCOV) associated with this event. The program should read this flag to determine any rollover occurrence. Writing to the RTCC also clears the prescaler if it is assigned to the RTCC (bit 3 at OPTION register is cleared). Using the “TEST fr” with RTCC (with fr being the RTCC and RTCC clock internally or externally) will not allow the RTCC to increment. The workaround is to use the “MOV W, RTCC” instruction instead.

10.2. Watchdog Timer

The watchdog logic consists of a Watchdog Timer which shares the same 8-bit programmable prescaler with the RTCC. The prescaler actually serves as a postscaler if used in conjunction with the WDT, in contrast to its use as a prescaler with the RTCC. The WDT is clocked by it’s own internal RC oscillator.

The Watchdog oscillator has a nominal operating frequency of 16 kHz, or a period of 62.5 microseconds. At this rate, the 8-bit counter counts from 00h to FFh in 16 milliseconds. In the default configuration (prescaler assigned to WDT, with divide rate set to 1:128), the application program needs to execute a “CLR !WDT” instruction at least once every 2 seconds to prevent a Watchdog reset (if the WDTE bit in the FUSE register is set to 1).

10.3. The Prescaler

The 8-bit prescaler may be assigned to either the RTCC or the WDT through the PSA bit (bit 3 of the OPTION register). Setting the PSA bit assigns the prescaler to the WDT. If assigned to the WDT, the WDT clocks the prescaler and the prescaler divide rate is selected by the PS0, PS1, and PS2 bits located in the OPTION register. Clearing the PSA bit assigns the prescaler to the RTCC. Once assigned to the RTCC, the prescaler clocks the RTCC and the divide rate is selected by the PS0, PS1, and PS2 bits in the OPTION register. The prescaler is not mapped into the data memory, so run-time access is not possible.

The prescaler cannot be assigned to both the RTCC and WDT simultaneously.

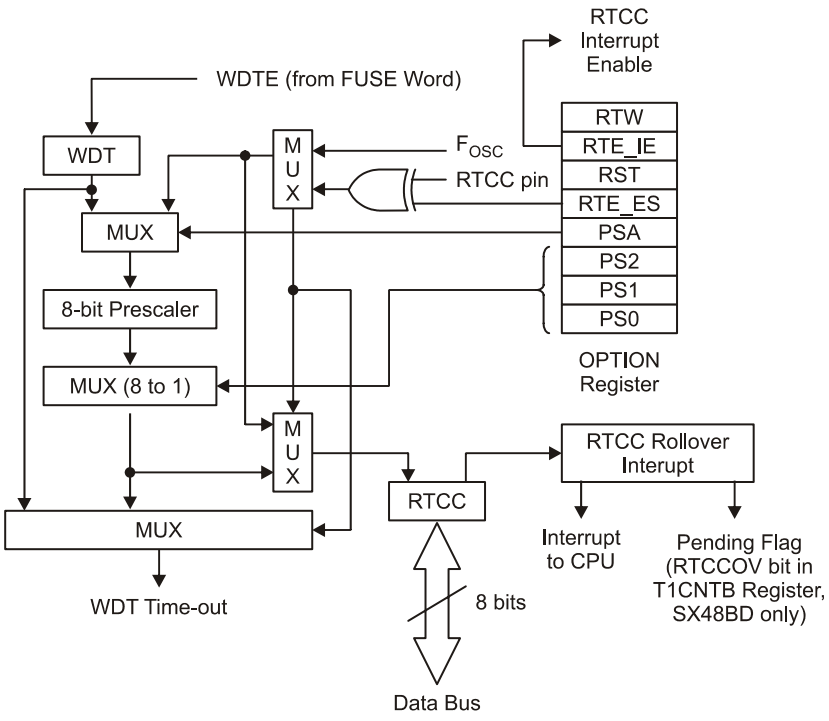


Figure 10-1: RTCC and WDT Block Diagram

11.0 MULTI-FUNCTION TIMERS

The device contains two independent 16-bit multi-func tion timers, designated T1 and T2. These versatile, programmable timers reduce the software burden on the CPU in real-time control applications such as PWM generation, motor control, triac control, variable-brightness display control, sine wave generation, and data acquisition.

Each timer consists of a 16-bit counter register supported by a dedicated 16-bit capture register and two 16-bit comparison registers. The second compare register can also serve as capture register. Each timer uses up to four I/O pins: one clocking input, two capture inputs, and one timer output. The timer I/O pins are alternate functions of Port B pins for timer T1 and Port C pins for Timer T2.

Figure 11-1 is a block diagram showing the registers and I/O pins of one timer. The 16-bit free-running timer/counter register is initialized to 0000h upon reset and counts upward continuously. It is clocked either by an external signal provided on an I/O pin or by the on-chip system clock divided by a 3-bit divide-by factor.

The CPU can access the Compare and Capture registers by using the "mov !RB,W" instruction for T1 or the "mov !RC,W" instruction for T2. The other timer registers are not directly accessible.

You can configure the timer to generate an interrupt upon overflow from FFFFh to 0000h, upon a match between the counter value and a programmed comparison value, or upon the occurrence of a valid capture signal on either of two capture inputs.

The timers can be cleared to 0000h by writing to the registers accessed via MODE address $10. Clearing the timer forces it to begin compare with R1.

The MODE register controls access to the timer registers. Because the MODE register is not memory mapped, it is accessed by the following special purpose instructions:

- mov M, #lit (move literal to lower 4-bits of MODE register)
- mov M,W (move W to lower 5-bits of MODE register)
- mov W,M (move MODE register to W)

The value contained in the MODE register determines which timer register is accessed by the "mov !rx,W" instruction as indicated in Table 11-1.

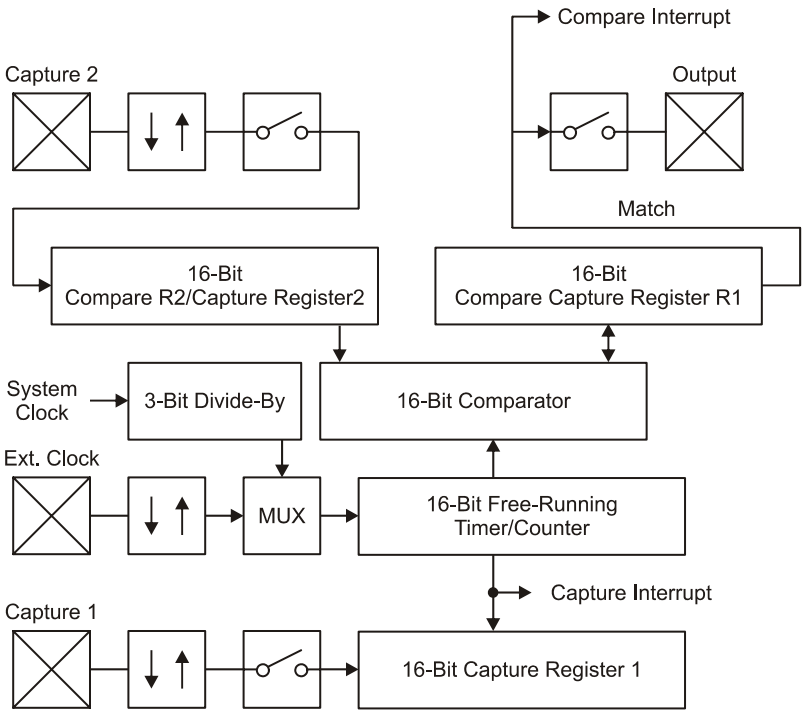


Figure 11-1: Multi-Function Timer Block Diagram

Table 11-1:Mode Register Settings for T1/T2 Registers

MODE Reg.	mov !RB,W	mov !RB,W
00h	Read T1CPL	Read T2CPL
01h	Read T1CPH	Read T2CPH
02h	Read T1R2CML	Read T2R2CML
03h	Read T1R2CMH	Read T2R2CMH
04h	Read T1R1CML	Read T2R1CML
05h	Read T1R1CMH	Read T2R1CMH
06h	Read T1CNTB	Read T2CNTB
07h	Read T1CNTA	Read T2CNTA
12h	Write T1R2CML	Write T2R2CML
13h	Write T1R2CMH	Write T2R2CMH
14h	Write T1R1CML	Write T2R1CML
15h	Write T1R1CMH	Write T2R1CMH
16h	Write T1CNTB	Write T2CNTB
17h	Write T1CNTA	Write T2CNTA

11.1. Timer Registers

Each timer consists of several registers.

Timer T1 registers:

T1CPL - Lower byte of Timer T1 capture register

T1CPH - Higher byte of Timer T1 capture register

T1R1CML - Lower byte of Timer T1 compare register 1

T1R1CMH - Higher byte of Timer T1 compare register 1

T1R2CML - Lower byte of Timer T1 compare register 2

T1R2CMH - Higher byte of Timer T1 compare register 2

T1CNTA - Timer T1 control register A

T1CNTB - Timer T1 control register B

Timer T2 registers:

T2CPL - Lower byte of Timer T2 capture register

T2CPH - Higher byte of Timer T2 capture register

T2R1CML - Lower byte of Timer T2 compare register 1

T2R1CMH - Higher byte of Timer T2 compare register 1

T2R2CML - Lower byte of Timer T2 compare register 2

T2R2CMH - Higher byte of Timer T2 compare register 2

T2CNTA - Timer T1 control register A

T2CNTB - Timer T1 control register B

11.2. Timer Operating Modes

Each timer can be configured to operate in one of the following modes:

- Pulse Width Modulation (PWM) mode
- Software Timer mode
- External Event mode
- Capture/Compare mode

11.2.1. PWM Mode

In the Pulse Width Modulation (PWM) mode, the timer generates an output signal having a programmable frequency and duty cycle. To use this mode, you load two 16-bit comparison registers, R1 and R2, with the number of timer clock cycles that you want the output signal to be high and low. The contents of R1 define the PWM low time while the contents of R2 define the PWM high time.

After the "Clear Timer" command is initiated through the MODE register, the timer starts from zero and counts up until it reaches the value in R1. At that point, it generates an interrupt (if enabled), toggles the output signal to a logic high level, and starts counting from zero again. The second time, it counts up until it reaches the value in R2. At that point, it again generates an interrupt (if enabled), toggles the output signal to a logic low level, and starts counting from zero again. This process is repeated continuously, alternating between R1 and R2 to obtain the value at which to toggle the output signal and return the counter to zero. The values of R1 and R2 establish the duty cycle and frequency of the output signal. If R1 and R2 contain the same value, the resulting output signal is a square wave. If R1 is changed to a value less than the timer count while the timer is counting to match R1, the timer will continue to count through FFFFh, and back up to the R1 value, while the output is low. Same is true for R2, except the output signal will be high.

Upon reset, the timer/counter is initialized to 0000.

In the PWM mode, the timer is clocked by the on-chip system clock divided by an 8-bit prescaler value. The divide-by factor can be set to any power-of-2 from 1 to 256. Thus, the period of the timer clock can be set from 1 to 256 times the system clock period.

Upon entering the PWM mode, the internally generated PWM signal is connected to the designated PWM output pin. The PWM mode bypasses the port data register (does not affect the contents of the data register). For the PWM output signal to appear on the pin (RB6 for T1, RC2 for T2), the corresponding port pin direction register must be configured for output.

11.2.2. Software Timer Mode

The Software Timer mode is the same as the PWM mode, except that the timer does not toggle the output signal. Instead, the application program takes action in response to the internally generated PWM signal upon each match between the counter and the contents of the active comparison value in either R1 or R2. The software can determine the cause of each interrupt by checking the timer interrupt pending flags. There are different flag bits associated with each type of event (R1 match, R2 match, and overflow).

11.2.3. External Event Mode

The External Event mode is the same as the PWM mode, except that the counter register is clocked by an external signal provided on an input pin (RB7 for T1 and RC3 for T2) rather than by the system clock. This mode can be used to count the occurrences of external events. The input pin can be configured to sense either rising or falling edges.

11.2.4. Capture/Compare Mode

In the Capture/Compare mode, the counter counts upward continuously without interruption. A valid transition received on either of two input pins causes the current value of the counter to be captured in an associated capture register. This capture feature can be used to keep track of the elapsed time between successive external events. In addition, the timer continuously compares the counter value against the value programmed into the R1 register. Each time a match occurs, it toggles the timer output pin, generates an interrupt (if enabled) and sets an associated interrupt pending flag. The timer continues to

count upward after a match occurs (unlike the PWM mode, which resets the counter to zero when a match occurs).

In the Capture/Compare mode, the timer is clocked by the on-chip system clock divided by a value defined by a 3-bit divide-by factor. The divide-by factor can be set to any power-of-2 from 1 to 128. The two input capture pins are designated Capture 1 and Capture 2. They can be configured to sense either rising or falling edges. The Capture 1 pin captures the counter value in a dedicated 16-bit capture register, a read-only register. The Capture 2 pin captures the counter value in the R2 register. The occurrence of a capture event also generates an interrupt (if enabled) and sets an associated interrupt pending flag.

Overflow of the counter from FFFFh to 0000h also generates an interrupt (if enabled) and sets an associated interrupt pending flag. Because the counter is free-running, an overflow can occur at any time. In cases where the time between successive capture events might exceed 65,536 counts of the timer, the software should keep track of the number of overflows between successive events in order to determine the true amount of time between such events.

11.3. Timer Pin Assignments

The following table lists the I/O port pins associated with the Timer T1 and Timer T2 I/O functions.

Table 11-2: Timer T1/T2 Pin Assignments

I/O Pin	Timer T1/T2 Function
RB4	Timer T1 Capture Input 1
RB5	Timer T1 Capture Input 2
RB6	Timer T1 PWM/Compare Output
RB7	Timer T1 External Event Clock Source
RC0	Timer T2 Capture Input 1
RC1	Timer T2 Capture Input 2
RC2	Timer T2 PWM/Compare Output
RC3	Timer T2 External Event Clock Source

11.4. Timer Control Registers

There are two 8-bit control registers associated with each timer, called the Control A and Control B registers. The Control A register contains the interrupt enable bits and interrupt flag bits associated with the timer. (Interrupts are caused by comparison, capture, and overflow events.) The Control B register contains bits for setting the timer operating mode, the clock prescaler divide-by factor, and the input signal edge sensitivity. Each Control B register also contains one device configuration bit not related to operation of the multi-function timers.

The register formats are shown in the following diagrams.

Timer T1 Control A Register (T1CNTA)

T1CPF2	T1CPF1	T1CPIE	T1CMF2	T1CMF1	T1CMIE	T1OVF	T1OVIE
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

T1CPF2 — Timer T1 Capture Flag 2. In Capture/Compare mode, this flag is automatically set to 1 when a capture event occurs on the Capture 2 pin of Timer T1 (pin RB5). It stays set until cleared by the software.

T1CPF1 — Timer T1 Capture Flag 1. In Capture/Compare mode, this flag is automatically set to 1 when a capture event occurs on the Capture 1 pin of Timer T1 (pin RB4). It stays set until cleared by the software.

T1CPIE — Timer T1 Capture Interrupt Enable. Set this bit to 1 to enable capture interrupts for Timer T1 in Capture/Compare mode. In that case, an interrupt will occur each time a valid edge is received on the Capture 1 or Capture 2 pin of Timer T1. Clear this bit to 0 to disable capture interrupts.

T1CMF2 — Timer T1 Comparison Flag 2. This flag is automatically set to 1 when the contents of the timer counter match the contents of R2, when R2 is the active comparison register. The flag stays set until it is cleared by the software.

T1CMF1 — Timer T1 Comparison Flag 1. This flag is automatically set to 1 when the contents of the timer counter match the contents of R1, when R1 is the active comparison register. The flag stays set until it is cleared by the software.

T1CMIE — Timer T1 Comparison Interrupt Enable. Set this bit to 1 to enable comparison interrupts for Timer T1. In that case, an interrupt will occur each time the contents of the timer counter match the contents of the active comparison register (R1 or R2) of Timer T1. Clear this bit to 0 to disable comparison interrupts.

T1OVF — Timer T1 Overflow Flag. This flag is automatically set to 1 when the timer counter overflows from FFFFh to 0000h. The flag stays set until it is cleared by the software.

T1OVIE — Timer T1 Overflow Interrupt Enable. Set this bit to 1 to enable overflow interrupts for Timer T1. In that case, an interrupt will occur each time Timer T1 overflows. Clear this bit to 0 to disable overflow interrupts.

Timer T1 Control B Register (T1CNTB)

RTCCOV	T1CPEDG	T1EXEDG	T1PS2 – T1PSO			T1MC1 – T1MC0	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

RTCCOV RTCC Overflow Flag. This flag is automatically set to 1 when the Real-Time Clock/Counter (RTCC) overflows from FFh to 00h. This flag stays set until it is cleared by the software. Note that this flag is not related to Multi-function timers T1 and T2.

T1CPEDG Timer T1 Capture Edge. This bit sets the edge sensitivity of the Timer T1 input capture pins, Capture 1 and Capture 2 (RB4 and RB5). Set this bit to 1 to sense positive-going (low-to-high) edges. Clear this bit to 0 to sense negative-going (high-to-low) edges.

T1EXEDG Timer T1 External Event Clock Edge. This bit sets the edge sensitivity of the Timer T1 input used to count external events (RB7). Set this bit to 1 to sense positive-going (low-to-high) edges. Clear this bit to 0 to sense negative-going (high-to-low) edges.

T1PS2-T1PSO Timer T1 Prescaler Divider field. This 3-bit field specifies the divide-by factor for generating the timer T1PS0clock from the on-chip system clock:

000 = divide by 1

001 = divide by 2

010 = divide by 4

011 = divide by 8

100 = divide by 16

101= divide by 32

110 = divide by 64

111 = divide by 128

For example, setting this field to 010 sets the divide-by factor to 4, which means that the T1 counter register is incremented once every four system clock cycles.

T1MC1-T1MCO Timer T1 Mode Control field. This 2-bit field specifies the Timer T1 operating mode as follows:

00 = Software Timer mode

01 = PWM mode

10 = Capture/Compare mode

11 = External Event mode

Timer T2 Control A Register (T2CNTA)

T2CPF2	T2CPF1	T2CPIE	T2CMF2	T2CMF1	T2CMIE	T2OVF	T2OVIE
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

T2CPF2 Timer T2 Capture Flag 2. In Capture/Compare mode, this flag is automatically set to 1 when a capture event occurs on the Capture 2 pin of Timer T2 (pin RC1). It stays set until cleared by the software.

T2CPF1 Timer T2 Capture Flag 1. In Capture/Compare mode, this flag is automatically set to 1 when a capture event occurs on the Capture 1 pin of Timer T2 (pin RC1). It stays set until cleared by the software.

T2CPIE Timer T2 Capture Interrupt Enable. Set this bit to 1 to enable capture interrupts for Timer T2 in Capture/Compare mode. In that case, an interrupt will occur each time a valid edge is received on the Capture 1 or Capture 2 pin of Timer T2. Clear this bit to 0 to disable capture interrupts.

T2CMF2 Timer T2 Comparison Flag 2. This flag is automatically set to 1 when the contents of the timer counter match the contents of R2, when R2 is the active comparison register. The flag stays set until it is cleared by the software.

T2CMF1 Timer T2 Comparison Flag 1. This flag is automatically set to 1 when the contents of the timer counter match the contents of R1, when R1 is the active comparison register. The flag stays set until it is cleared by the software.

T2CMIE Timer T2 Comparison Interrupt Enable. Set this bit to 1 to enable comparison interrupts for Timer T2. In that case, an interrupt will occur each time the contents of the timer counter match the contents of the active comparison register (R1 or R2) of Timer T2. Clear this bit to 0 to disable comparison interrupts.

T2OVF Timer T2 Overflow Flag. This flag is automatically set to 1 when the timer counter overflows from FFFFh to 0000h. The flag stays set until it is cleared by the software.

T2OVIE Timer T2 Overflow Interrupt Enable. Set this bit to 1 to enable overflow interrupts for Timer T2. In that case, an interrupt will occur each time Timer T2 overflows. Clear this bit to 0 to disable overflow interrupts.

Timer T2 Control b Register (T2CNTb)

PRTRD	T2CPEDG	T2EXEDG	T2PS2 – T2PS0			T2MC1 – T2MC0	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

PORTRD Port Read mode. This bit determines how the device reads data from its I/O ports (Port A through Port E). Clear this bit to 0 to have the device read data from the port I/O pins directly. Set this bit to 1 to have the device read data from the port data registers. Under normal (output mode) conditions, it should not matter which method you use to read the port data. However, if a port pin is configured as an output and an external circuit forces the pin to the wrong value, the value read from the port will depend on the reading mode used. Note that this control bit is not related to multi-function timers T1 and T2.

T2CPEDG Timer T2 Capture Edge. This bit sets the edge sensitivity of the Timer T2 input capture pins, Capture 1 and Capture 2 (RC0 and RC1). Set this bit to 1 to sense positive-going (low-to-high) edges. Clear this bit to 0 to sense negative-going (high-to-low) edges.

T2EXEDG Timer T2 External Event Clock Edge. This bit sets the edge sensitivity of the Timer T2 input used to count external events (RC3). Set this bit to 1 to sense positive-going (low-to-high) edges. Clear this bit to 0 to sense negative-going (high-to-low) edges.

T2PS2-T2PS0 Timer T2 Prescaler Divider field. This 3-bit field specifies the divide-by factor for generating the timer clock from the on-chip system clock:

000 = divide by 1

001 = divide by 2

010 = divide by 4

011 = divide by 8

100 = divide by 16

101 = divide by 32

110 = divide by 64

111 = divide by 128

For example, setting this field to 010 sets the divide-by factor to 4, which means that the T2 counter register is incremented once every four system clock cycles.

T2MC1- T2MC0 Timer T2 Mode Control field. This 2-bit field specifies the Timer T1 operating mode as follows:

00 = Software Timer mode

01 = PWM mode

10 = Capture/Compare mode

11 = External Event mode

12.0 COMPARATOR

The device contains an on-chip differential comparator. Ports RB0-RB2 support the comparator. Pins RB1 and RB2 are the comparator negative and positive inputs, respectively, while RB0 serves as the comparator output pin. To use these pins in conjunction with the comparator, the user program must configure RB1 and RB2 as inputs and RB0 as an output. The CMP_B register is used to enable the comparator, to read the output of the comparator internally, and to enable the output of the comparator to the comparator output pin.

The comparator enable bits are set to "1" upon reset, thus disabling the comparator. To avoid drawing additional current during the power down mode, the comparator should be disabled before entering the power down mode. Here is an example of how to set up the comparator and read the CMP_B register.

```
                ;enable RB0 as output
mov     W,#$18  ;set MODE register to access
mov     M,W     ;CMP_B
mov     W,#$00  ;clear W
mov     !RB,W   ;enable comparator and its
                ;output
                ;delay after enabling
                ;comparator for response
mov     W,#$18  ;set MODE register to access
mov     M,W     ;CMP_B
mov     W,#$00  ;clear W
mov     !RB,W   ;enable comparator and its
                ;output and also read CMP_B
                ;(exchange W and CMP_B)
and     W,#$01  ;set/clear Z flag based on
                ;comparator result
snb      $03.2  ;test Z flag in STATUS reg
                ;(0 => RB2<RB1)
jmp rb2_hi      ;jump only if RB2>RB1
```

The final "mov" instruction in this example performs an exchange of data between the working register (W) and the CMP_B register. This exchange occurs only with accesses to CMP_B and WKPEND_B. Otherwise, the "mov" instruction does not perform an exchange, but only moves data from the source to the destination.

The following figure shows the format of the CMP_B register.

$\overline{\text{CMP\_EN}}$	$\overline{\text{CMP\_OE}}$	Reserved	CMP_RES
Bit 7	Bit 6	Bits 5 – 1	Bit 0

$\overline{\text{CMP\_EN}}$ — When cleared to 0, enables the comparator.

$\overline{\text{CMP\_OE}}$ — When cleared to 0, enables the comparator output to the RB0 pin if RB0 is configured as an output.

CMP_RES — Comparator result (Read Only): 1 for RB2>RB1 or 0 for RB2<RB1. Comparator must be enabled (CMP_EN = 0) to read the result. The result can be read whether or not the CMP_OE bit is cleared.

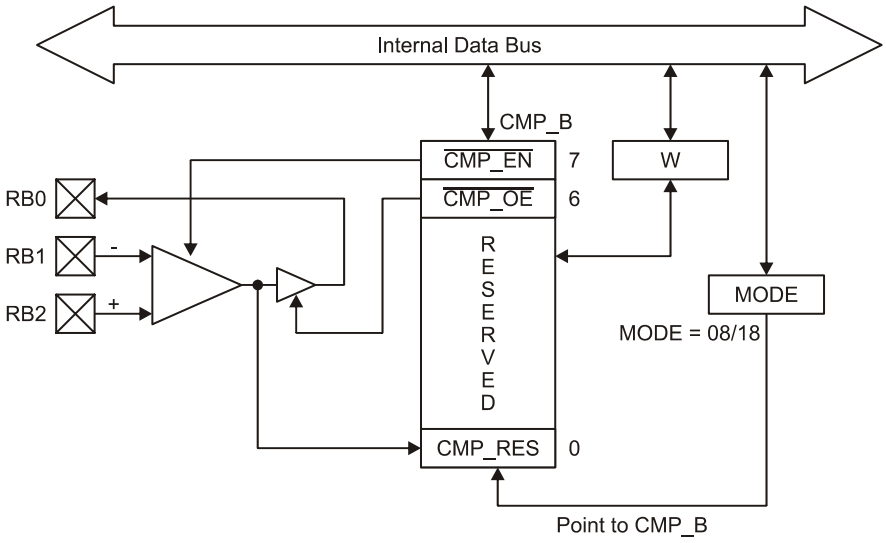


Figure 12-1: Comparator Block Diagram

13.0 RESET

Power-On Reset, Brown-Out Reset, Watchdog Reset, or External Reset initializes the device. Each one of these reset conditions causes the program counter to branch to the top of the program memory. For example, on the device with 4096 K ($1000 hex) words of program memory, the program counter is initialized to 0FFF upon a valid reset condition.

The device incorporates an on-chip Power-On Reset (POR) circuit that generates an internal reset as V_{DD} rises during power-up. Figure 13-2 is a block diagram of the circuit. The circuit contains a 10-bit Delay Reset Timer (DRT) and a reset latch. The DRT controls the reset timeout delay. The reset latch controls the internal reset signal.

Upon power-up, the reset latch is set (device held in reset), and the DRT starts counting once it detects a valid logic high signal at the $\overline{MCLR}$ pin. Once DRT reaches the end of the timeout period (typically 72 msec), the reset latch is cleared, releasing the device from reset state. Figure 13-1 shows a power-up sequence where $\overline{MCLR}$ is not tied to the V_{DD} pin and V_{DD} signal is allowed to rise and stabilize before $\overline{MCLR}$ pin is brought high. The device will actually come out of reset T_{DRT} msec after $\overline{MCLR}$ goes high.

The brown-out circuitry resets the chip when device power (V_{DD}) dips below its minimum allowed value, but not to zero, and then recovers to the normal value.

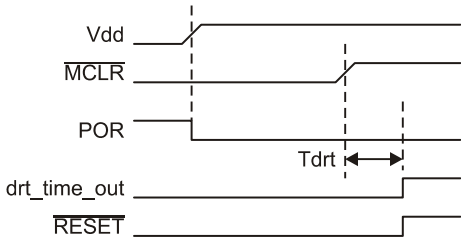


Figure 13-1: Time-Out Sequence on Power-Up ($\overline{MCLR}$ not tied to V_{dd})

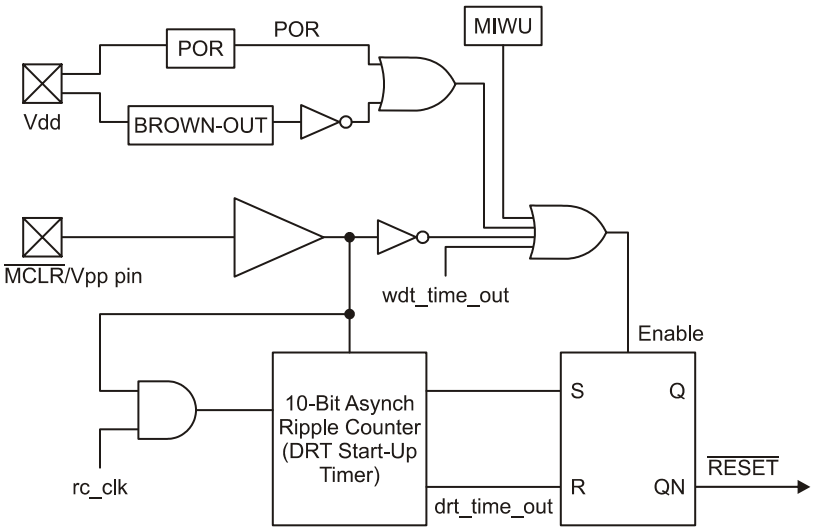


Figure 13-2
Block Diagram
of On-Chip Reset Circuit

Note: Ripple counter is 10 bits for Power on Reset (POR) only.

Figure 13-3 shows the on-chip Power-On Reset sequence where the MCLR pin is tied to V_{DD} via a 10K resistor. Note: connecting the MCLR pin directly to the V_{DD} supply is **not** recommended. If the V_{DD} signal is stable before the DRT timeout period expires, the device will receive a proper reset. However, Figure 13-4 depicts a situation where V_{DD} rises too slowly. In this scenario, the DRT will time-out prior to V_{DD} reaching a valid operating voltage level (V_{DD} min). This means the device will come out of reset and start operating with the supply voltage not at a valid level. In this situation, it is recommended that you use the external RC circuit shown in Figure 13-5. The RC delay should exceed the time period it takes V_{DD} to reach a valid operating voltage.

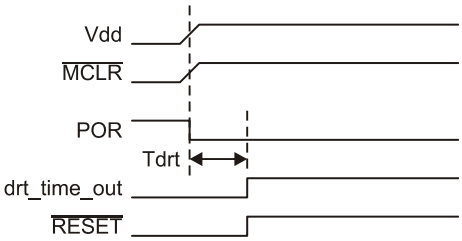


Figure 13-3: Time-Out Sequence on Power-Up (MCLR not tied to Vdd): Fast Vdd Rise Time

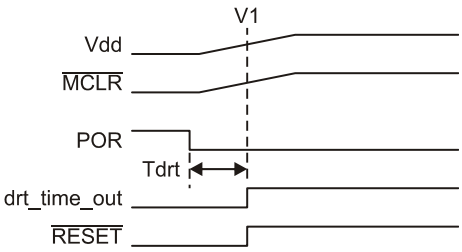


Figure 13-4: Time-Out Sequence on Power-Up (MCLR not tied to Vdd): Slow Vdd Rise Time

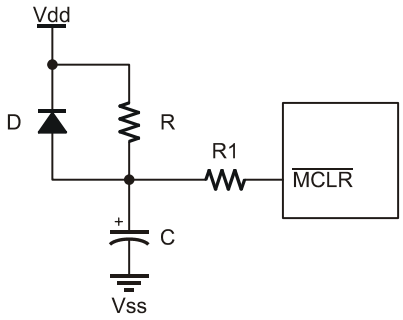


Figure 13-5: External Power-On Reset Circuit (For Slow Vdd Power-Up)

A 2-bit field in the FUSEX register can be used to specify the Delay Reset Timer (DRT) timeout period that results in an automatic wake-up from the power down mode.

10 = 0.25 msec
11 = 18 msec (default)
00 = 60 msec
01 = 1 sec

Note 1: The external Power-On Reset circuit is required only if V_{DD} power-up is too slow. The diode D helps discharge the capacitor quickly when V_{DD} powers down.

Note 2: R < 40 kΩ is recommended to make sure that voltage drop across R does not violate the device electrical specifications.

Note 3: R1 = 100 Ω to 1 kΩ will limit any current flowing into MCLR from external capacitor C. This helps prevent MCLR pin breakdown due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS).

14.0 BROWN-OUT DETECTOR

The on-chip brown-out detection circuitry resets the device when V_{dd} dips below the specified brown-out voltage. The device is held in reset as long as V_{dd} stays below the brown-out voltage. The device will come out of reset when V_{dd} rises above the brown-out voltage. The brown-out level can be set to 2.2 V, 2.4 V, or 4.1 V levels through BOR1:BOR0 bits in the FUSEX register.

15.0 REGISTER STATES UPON DIFFERENT RESET CONDITIONS

The effect of different reset operations on a register depends on the register and the type of reset operation. Some registers are initialized to specific values, some are left unchanged, some are undefined, and some are initialized to an unknown value.

A register that starts with an unknown value should be initialized by the software to a known value; you cannot simply test the initial state and rely on it starting in that state consistently. Table 15-1 lists the SX registers and shows the state of each register upon reset, with a different column for each type of reset.

Table 15-1: Register States upon Different Resets

Register	Power-On	Wakeup	Brown-Out	Watchdog Timeout	MCLR
W	Undefined	Unchanged	Undefined	Unchanged	Unchanged
OPTION	FFh	FFh	FFh	FFh	FFh
MODE (Note 3)	1Fh	1Fh	1Fh	1Fh	1Fh
RTCC (01h)	Undefined	Unchanged	Undefined	Unchanged	Unchanged
PC (02h)	FFh	FFh	FFh	FFh	FFh
STATUS (03h) (Note 3)	Bits 0-2: Undefined Bits 3-4: 11 Bits 5-7: 000	Bits 0-2: Unchanged Bits 3-4: Unchanged Bits 5-7: 000	Bits 0-4: Undefined Bits 5-7: 000	Bits 0-2: Unchanged Bits 3-4: (Note 1) Bits 5-7: 000	Bits 0-2: Unchanged Bits 3-4: (Note 2) Bits 5-7: 1
FSR (04h)	Undefined	Bits 0-6: Unchanged Bit 7: 1	Bits 0-6: Undefined Bit 7: 1	Bits 0-6: Unchanged Bit 7: 1	Bits 0-6: Unchanged Bit 7: 1
RA through RE Direction	FFh	FFh	FFh	FFh	FFh
RA through RE Data	Undefined	Unchanged	Undefined	Unchanged	Unchanged
Other File Registers - SRAM	Undefined	Unchanged	Undefined	Unchanged	Unchanged
CMP_B	Bits 0, 6-7: 1 Bits 1-5: Undefined	Bits 0, 6-7: 1 Bits 1-5: Undefined	Bits 0, 6-7: 1 Bits 1-5: Undefined	Bits 0, 6-7: 1 Bits 1-5: Undefined	Bits 0, 6-7: 1 Bits 1-5: Undefined
WKPND_B	Undefined	Unchanged	Undefined	Unchanged	Unchanged
WKED_B	FFh	FFh	FFh	FFh	FFh
WKEN_B	FFh	FFh	FFh	FFh	FFh
ST_B – ST_E	FFh	FFh	FFh	FFh	FFh
LVL_A through LVL_E	FFh	FFh	FFh	FFh	FFh
PLP_A through PLP_E	FFh	FFh	FFh	FFh	FFh
Watchdog Counter	Undefined	Unchanged	Undefined	Unchanged	Unchanged
Timers T1 and T2 Free-Running Timer/Counter	0001	0001	0001	0001	0001
Timers T1 and T2 Compare/Capture Registers	0000	0000	0000	0000	0000
Timers T1 and T2 Control Registers (Note 3)	00	00	00	00	00

Note 1: Watchdog reset during power down mode: 00 (Bits TO, PD); Watchdog reset during Active mode: 01 (Bits TO, PD).
Note 2: External reset during power down mode: 10 (Bits TO, PD); External reset during Active mode: Unchanged (Bits TO, PD).
Note 3: MODE, STATUS and Timer registers are not initialized properly by the development system in Debug mode.

16.0 INSTRUCTION SET

As mentioned earlier, the SX family of devices uses a modified Harvard architecture with memory-mapped input/output. The device also has a RISC type architecture in that there are 43 single-word basic instructions. The instruction set contains byte-oriented file register, bit-oriented file register, and literal/control instructions.

Working register W is one of the CPU registers, which serves as a pseudo accumulator. It is a pseudo accumulator in a sense that it holds the second operand, receives the literal in the immediate type instructions, and also can be program-selected as the destination register. The bank of 31 file registers can also serve as the primary accumulators, but they represent the first operand and may be program-selected as the destination registers.

16.1. Instruction Set Features

- All single-word (12-bit) instructions for compact code efficiency.
- All instructions are single cycle except the jump type instructions (JMP, CALL) and failed test instructions (DECSZ fr, INCSZ fr, SB bit, SNB bit), which are two-cycle.
- A set of file registers can be addressed directly or indirectly, and serve as accumulators to provide first oper and; W register provides the second operand.
- Many instructions include a destination bit which selects either the register file or the accumulator as the destination for the result.
- Bit manipulation instructions (Set, Clear, Test and Skip if Set, Test and Skip if Clear).
- STATUS Word register memory-mapped as a register file, allowing testing of status bits (carry, digit carry, zero, power down, and timeout).
- Program Counter (PC) memory-mapped as register file allows W to be used as offset register for indirect ad dressing of program memory.
- Indirect addressing data pointer FSR (file select register) memory-mapped as a register file.
- IREAD instruction allows reading the instruction from the program memory addressed by W and upper four bits of MODE register.
- Eight-level, 12-bit push/pop hardware stack for subroutine linkage using the Call and Return instructions.
- Seven addressing mode provide great flexibility.

16.2. Instruction Execution

An instruction goes through a four-stage pipeline to be executed (Figure 16-1). The first instruction is fetched from the program memory on the first clock cycle. On the second clock cycle, the first instruction is decoded and the second instruction is fetched. On the third clock cycle, the first instruction is executed, the second instruction is decoded, and the third instruction is fetched. On the fourth clock cycle, the first instruction's results are written to its destination, the second instruction is executed, the third instruction is decoded, and the fourth instruction is fetched. Once the pipeline is full, instructions are executed at the rate of one per clock cycle.

Instructions that directly affect the contents of the program counter (such as jumps and calls) require that the pipeline be cleared and subsequently refilled. Therefore, these instructions take more than one clock cycle.

The instruction execution time is derived by dividing the oscillator frequency by one (bit 11 of the FUSE Word register must be initialized to 0).

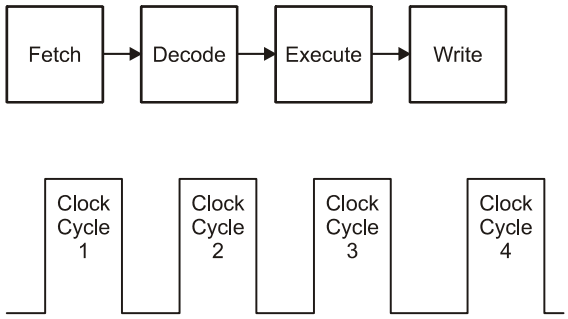


Figure 16-1: Pipeline and Clock Scheme

16.3. Addressing Modes

The device support the following addressing modes:

- Data Direct
- Data Indirect
- Data Semidirect
- Immediate
- Program Direct
- Program Indirect
- Relative

Both direct and indirect addressing modes are available. The INDF register, though physically not implemented, is used in conjunction with the indirect data pointer (FSR) to perform indirect addressing. An instruction using INDF as its operand field actually performs the operation on the register pointed by the contents of the FSR. Consequently, processing two multiple-byte operands requires alternate loading of the operand addresses into the FSR pointer as the multiple byte data fields are processed.

Examples:

Direct addressing:

```
mov     W,#1
mov     RA,W            ;move "1" to RA
```

Indirect Addressing:

```
mov     W,#RA
mov     FSR,W           ;FSR = address of RA
mov     INDF,#$01       ;move "1" to RA
```

Semidirect Addressing:

```
mov     W,#$00
mov     FSR,W           ;FSR = bank 0 address
inc     $1F             ;increment file
                        ;register 0Fh
```

16.4. The Bank Instruction

Often it is desirable to set the bank select bits of the FSR register in one instruction cycle. The Bank instruction provides this capability. This instruction sets the upper 3 bits 4, 5 and 6 of the FSR to point to a specific RAM bank without affecting the lower 4 FSR bits, in preparation for using direct or semidirect addressing. Bit 7 of the FSR register is used to select the lower or upper block of banks.

Example:

```
bank    $E0     ;Select Bank E in FSR
inc     $1F     ;increment file register
                ;EFh using semidirect addressing
```

16.5. Bit Manipulation

The instruction set contains instructions to set, reset, and test individual bits in data memory. The device is capable of bit addressing anywhere in data memory.

16.6. Input/Output Operation

The device contains three registers associated with each I/O port. The first register (Data Direction Register), configures each port pin as a Hi-Z input or output. The second register (TTL/CMOS Register), selects the desired input level for the input. The third register (Pull-Up Register), enables a weak pull-up resistor on the pin configured as a input. To read or write these registers, you must first write an appropriate value into the MODE register to select the desired register set, and then use the "mov !rx,W" instruction to read or write the register.

16.6.1. Read-Modify-Write Considerations

When two successive instructions are used on the same I/O port (except "mov Rx, W") with a very high clock rate, the "write" part of one instruction might not occur soon enough before the "read" part of the very next instruction, resulting in getting "old" data for the second instruction. To ensure predictable results, avoid using two successive read-modify-write instructions that access the same port data register if the clock rate is high or, insert 3 NOP instructions between the successive read-modify-write instructions (if SYNC bit in the FUSE register is enabled, 5 NOP instructions are required). For operating frequencies of 50 Mhz or lower, if bit 7 of the T2CNTB (PORTRD) is set, the port reads data from the data register instead of port pins. In this case, the NOP instructions are not required.

In the default device configuration, when a read is performed from a port bit position, the operation is actually reading the voltage level on the pin itself, not necessarily the bit value stored in the port data register. This is true whether the pin is configured to operate as an input or an output. Therefore, with the pin configured to operate as an input, the data register contents have no effect on the value that you read. With the pin configured to operate as an output, what is read generally matches what has been written to the register. PORTRD of the T2CNTB register determines how the device reads data from its I/O ports (Port A through Port E). Clear this bit to 0 to have the device read data from the port I/O pins directly. Set this bit to 1 to have the device read data from the port data registers. Under normal output mode conditions, it should not matter which method you use to read the port data. However, if a port pin is configured as an output and an external circuit forces the pin to the opposite value, the value read from the port will depend on the reading mode used. Note that this control bit is not related to multi-function timers T1 and T2.

16.7. Increment/Decrement

The current selected bank of 31 registers serves as a set of accumulators. The instruction set contains instructions to increment and decrement the register file. The device also includes both INCSZ fr (increment file register and skip if zero) and DECSZ fr (decrement file register and skip if zero) instructions.

16.8. Loop Counting and Data Pointing Testing

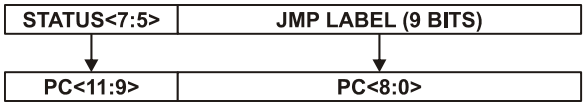
The device has specific instructions to facilitate loop counting. The DECSZ fr (decrement file register and skip if zero) tests any one of the file registers and skips the next instruction (which can be a branch back to loop) if the result is zero.

16.9. Branch and Loop Call Instructions

The device contains an 8-level hardware stack where the return address is stored with a subroutine call. Multiple stack levels allow subroutine nesting. The instruction set supports absolute address branching.

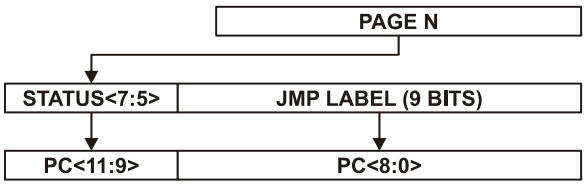
16.9.1. Jump Operation

When a JMP instruction is executed, the lower nine bits of the program counter are loaded with the address of the specified label. The upper three bits of the program counter are loaded with the page select bits, PA2:PA0, contained in the STATUS register. Therefore, care must be exercised to ensure the page select bits are pointing to the correct page before the jump occurs.



16.9.2. Page Jump Operation

When a JMP instruction is executed and the intended destination is on a different page, the page select bits must be initialized with appropriate values to point to the desired page before the jump occurs. This can be done easily with SETB and CLRB instructions or by writing a value to the STATUS register. The device also has the PAGE instruction, which automatically selects the page in a single-cycle execution.



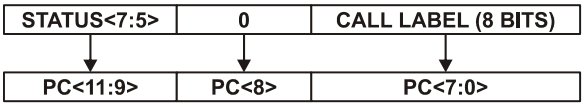
Note:"N" must be 0, 1, 2, or 3.

16.9.3. Call Operation

The following happens when a CALL instruction is executed:

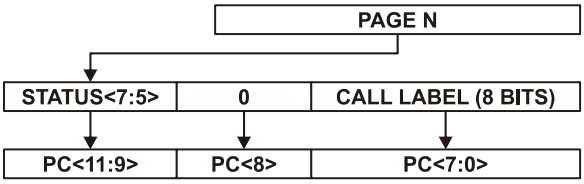
- The current value of the program counter is incremented and pushed onto the top of the stack.
- The lower eight bits of the label address are copied into the lower eight bits of the program counter.
- The ninth bit of the Program Counter is cleared to zero.
- The page select bits (in STATUS register) are copied into the upper three bits of the 12-bit program counter.

This means that the call destination must start in the lower half of any page. For example, 00h-0FFh, 200h-2FFh, 400h-4FFh, etc.



16.9.4. Page Call Operation

When a subroutine that resides on a different page is called, the page select bits must contain the proper values to point to the desired page before the call instruction is executed. This can be done easily using SETB and CLRB instructions or writing a value to the STATUS register. The device also has the PAGE instruction, which automatically selects the page in a single-cycle execution.



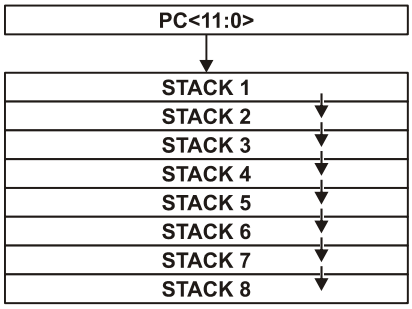
16.10. Return Instructions

The device has several instructions for returning from subroutines and interrupt service routines. The return from subroutine instructions are RET (return without affecting W), RETP (same as RET but affects PA2:PA0), RETI (return from interrupt), RETIW (return and add W to RTCC), and RETW #literal (return and place literal in W). The literal serves as an immediate data value from memory. This instruction can be used for table lookup operations. To do table lookup, the table must contain a string of RETW #literal instructions. The first instruction just in front of the table calculates the offset into the table. The table can be used as a result of a CALL.

16.11. Subroutine Operation

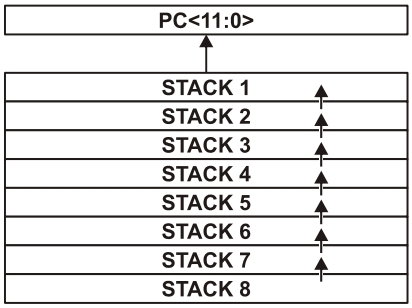
16.11.1. Push Operation

When a subroutine is called, the return address is pushed onto the subroutine stack. Specifically, each address in the stack is moved to the next lower level in order to make room for the new address to be stored. Stack 1 receives the contents of the program counter. Stack 8 is overwritten with what was in Stack 7. The contents of stack 8 are lost.



16.11.2. Pop Operation

When a return instruction is executed the subroutine stack is popped. Specifically, the contents of Stack 1 are copied into the program counter and the contents of each stack level are moved to the next higher level. For example, Stack 1 receives the contents of Stack 2, etc., until Stack 7 is overwritten with the contents of Stack 8. Stack 8 is left unchanged, so the contents of Stack 8 are duplicated in Stack 7.



16.12. Comparison and Conditional Branch Instructions

The instruction set includes instructions such as DECSZ fr (decrement file register and skip if zero), INCSZ fr (increment file register and skip if zero), SNB bit (bit test file register and skip if bit clear), and SB bit (bit test file register and skip if bit set). These instructions will cause the next instruction to be skipped if the tested condition is true. If a skip instruction is immediately followed by a PAGE or BANK instruction (and the tested condition is true) then two instructions are skipped and the operation consumes three cycles. This is useful for conditional branching to another page where a PAGE instruction precedes a JMP. If several PAGE and BANK instructions immediately follow a skip instruction then they are all skipped plus the next instruction and a cycle is consumed for each.

16.13. Logical Instruction

The instruction set contain a full complement of the logical instructions (AND, OR, Exclusive OR), with the W register and a selected memory location (using either direct or indirect addressing) serving as the two operands.

16.14. Shift and Rotate Instructions

The instruction set includes instructions for left or right rotate-through-carry.

16.15. Complement and SWAP

The device can perform one's complement operation on the file register (fr) and W register. The MOV W,<>fr instruction performs nibble-swap on the fr and puts the value into the W register.

16.16. Key to Abbreviations and Symbols

Table 16-1: Key to Abbreviations and Symbols			
Symbol	**Description**	**Symbol**	**Description**
W	Working register	n	Numerical value bit in opcode
fr	File register (memory mapped in range of 00h to FFh)	b	Bit position selector bit in opcode
PC	Lower eight bits of program counter (file register 02h)	.	File register / bit selector separator in assembly instruct.
STATUS	STATUS register (file register 03h)	#	Immediate literal designator in assembly instruction
FSR	File Select Register (file register 04h)	lit	Literal value in assembly language instruction
C	Carry bit in STATUS register (Bit 0)	addr8	8-bit address in assembly language instruction
DC	Digit Carry bit in STATUS register (Bit 1)	addr9	9-bit address in assembly language instruction
Z	Zero bit in STATUS register (Bit 2)	addr12	12-bit address in assembly language instruction
PD	Power Down bit in STATUS register (Bit 3)	/	Logical 1's complement
TO	Watchdog Timeout bit in STATUS register (Bit 4)	|	Logical OR
PA2:PA0	Page select bits in STATUS register (Bits7:5)	^	Logical exclusive OR
OPTION	OPTION register (not memory mapped)	&	Logical AND
WDT	Watchdog Timer register (not memory-mapped)	<>	Swap high and low nibbles (4-bit segments)
MODE	MODE register (not memory mapped)	<<	Rotate left through carry bit
rx	Port control register pointer (RA, RB, RC)	>>	Rotate right through carry bit
!	Non-memory-mapped register designator	--	Cecrement file register
f	File register address bit in opcode	++	Increment file register
k	Constant value bit in opcode		

17.0 NATIVE INSTRUCTION SET SUMMARY TABLES

Table 17-1 through Table 17-6 list all of the native assembly instructions, organized by category. For each instruction, the table shows the instruction mnemonic (as written in assembly language), a brief description of what the instruction does, the number of clock cycles required for execution, the binary opcode, and the status flags affected by the instruction.

The "Clock Cycles" column typically shows a value of 1, which means that the overall throughput for the instruction is one per clock cycle. Exceptions include program control branch instructions, which take 3 clock cycles, and the system control instruction IREAD, which takes 4.

In some cases, the exact number of cycles depends on the outcome of the instruction (such as the test-and-skip instructions). In those cases, all possible numbers of cycles are shown in the table.

The instruction cycle time is derived by multiplying the oscillator period by the number of clock cycles required by the instruction.

A superset of these instructions are available in the SASM assembler of the SX-Key IDE, and supported by the *SX Key User's Manual*. Both are available for free download from www.parallax.com/sx.

Table 17-1: Native SX Instruction Set: Logical Operands

Mnemonic, Operands	Description	Clock Cycles	Opcode	Flags Affected
AND fr,W	AND of fr and W into fr (fr = fr & W)	1	0001 011f ffff	Z
AND W,fr	AND of W and fr into W (W = W & fr)	1	0001 010f ffff	Z
AND W,#lit	AND of W and Literal into W (W = W &b lit)	1	1110 kkkk kkkk	Z
NOT fr	Complement of fr into fr (fr = fr ^ FFh)	1	0010 011f ffff	Z
OR fr,W	OR of fr and W into fr (fr = fr | W)	1	0001 001f ffff	Z
OR W,fr	OR of W and fr in to fr (W – W | fr)	1	0001 000f ffff	Z
OR w,#lit	Or of W and Literal into W (W = W | lit)	1	1101 kkkk kkkk	Z
XOR fr,W	XOR of fr and W into fr (fr = fr ^ W)	1	0001 101f ffff	Z
XOR W,fr	XOR of W and fr into W (W = W ^ fr)	1	0001 100f ffff	Z
XOR W,#lit	XOR of W and Literal into W (W = W ^ lit)	1	1111 kkkk kkkk	Z

Table 17-2: Native SX Instruction Set: Arithmetic and Shift Operations				
Mnemonic, Operands	**Description**	**Clock Cycles**	**Opcode**	**Flags Affected**
ADD fr,W	Add W to fr (fr = fr + W); carry bit is added if $\overline{CF}$ bit in FUSEX register is cleared to 0	1	0001 111f ffff	C, DC, Z
ADD W,fr	Add fr to W (W = W +fr); carry bit is added if $\overline{CF}$ bit in FUSEX register is cleared to 0	1	0001 110f ffff	C, DC, Z
CLR fr	Clear fr (fr = 0)	1	0000 011f ffff	Z
CLR W	Clear W (W = 0)	1	0000 0100 0000	Z
CLR !WDT	Clear Watchdog Timer, clear prescaler if assigned to the Watchdog (TO = 1, PD = 1)	1	0000 0000 0100	TO, PD
DEC fr	Decrement fr (fr = fr – 1)	1	0000 111f ffff	Z
DECSZ fr	Decrement fr and Skip if Zero (fr = fr – 1 and skip next instruction if result is zero)	1 or 2 (skip)	0010 111f ffff	none
INC fr	Increment fr (fr = fr +1)	1	0010 101f ffff	Z
INCSZ fr	Increment fr and Skip if Zero (fr = fr +1 and skip next instruction if result is zero)	1 or 2 (skip)	0011 111f ffff	none
RL fr	Rotate fr Left through Carry (fr = << fr)	1	0011 011f ffff	C
RR fr	Rotate fr Right through Carry (fr = >> fr)	1	0011 001f ffff	C
SUB fr,W	Subtract W from fr (fr = fr – $\underline{W}$); complement of the carry bit is subtracted if $\overline{CF}$ bit in FUSEX register is cleared to 0	1	0000 101f ffff	C, DC, Z
SWAP fr	Swap high/low nibbles so fr xy = yx (fr = <> fr)	1	0011 101f ffff	none

Table 17-3: Native SX Instruction Set: Bitwise Operations				
Mnemonic, Operands	**Description**	**Clock Cycles**	**Opcode**	**Flags Affected**
CLRB fr.bit	Clear Bit in fr (fr.bit = 0)	1	0100 bbbf ffff	none
SB fr.bit	Test Bit in fr and Skip if set (test fr.bit and skip next instruction if bit is 1)	1 or 2 (skip)	0111 bbbf ffff	none
SETB fr.bit	Set Bit in fr (fr.bit = 1)	1	0101 bbbf ffff	none
SNB fr.bit	Test Bit in fr and Skip if clear (test fr.bit and skip next instruction if bit is 0)	1 or 2 (skip)	0110 bbbf ffff	none

Table 17-4: Native SX Instruction Set: Data Movement Instructions				
Mnemonic, Operands	**Description**	**Clock Cycles**	**Opcode**	**Flags Affected**
MOV fr,W	Move W to fr (fr = W)	1	0000 001f ffff	none
MOV W,fr	Move fr to W (W = fr)	1	0010 000f ffff	Z
MOV W,fr-W	Move (fr-W) to W (W = fr-W); complement of carry bit is subtracted if CF bit in FUSEX register is cleared to 0	1	0000 100f ffff	C, DC, Z
MOV W,#lit	Move Literal to W (W = lit)	1	1100 kkkk kkkk	none
MOV W,/fr	Move Complement of fr to W (W = fr ^ FFh)	1	0010 010f ffff	Z
MOV W,--fr	Move (fr – 1) to W (W = fr – 1)	1	0000 110f ffff	Z
MOV W,++fr	Move (fr = 1) to W (W = fr = 1)	1	0010 100f ffff	Z
MOV W,<<fr	Rotate fr Left through Carry and Move to W (W = >> ffr)	1	0011 010f ffff	C
MOV W,>>fr	Rotate fr Right through Carry and Move to W (W = <> fr)	1	0011 000f ffff	C
MOV W,<>fr	Swap High/Low nibbles of fr and move to W (W = <>fr)	1	0011 100f ffff	none
MOV W,M	Move MODE register to W (W = MODE), high nibble is cleared	1	0000 0100 0010	none
MOVSZ W,--fr	Move (fr – 1) to W and Skip if Zero (w = fr – 1 and skip next instruction if result is zero)	1 or 2 (skip)	0010 110f ffff	none
MOVSZ W,++fr	Move (fr + 1) to W and Skip of Zero (W = fr +1 and skip next instruction if result is zero)	1 or 2 (skip)	0011 110f ffff	none
MOV M,W	Move W to MODE register (MODE = W)	1	0000 0100 0011	none
MOV M,#lit	Move Literal to MODE register (MODE = lit)	1	0000 0101 kkkk	none
MOV !rx,W	Move W to Port Rx control register: rx <=> W (exchange W and WKPND_B or CMP_B ***or*** rx = w (move W to rx for all other port control registers)	1	0000 0000 0fff	none
MOV !OPTION,W	Move W to OPTION register (OPTION = W)	1	0000 0000 0010	none
TEST fr	Test fr for zero (fr = fr to set or clear Z bit)	1	0010 001f ffff	Z

Table 17-5: Native SX Instruction Set: Program Control Instructions				
Mnemonic, Operands	**Description**	**Clock Cycles**	**Opcode**	**Flags Affected**
CALL addr8	Call Subroutine: Top-of-stack = program counter + 1 PC(7:0) = addr8 Program counter (8) = 0 Program counter (10:9) = PA1:PA0	3	1001 kkkk kkkk	none
JMP addr9	Jump to Address: PC(7:0) = addr9(7:0) Program counter = (8) = addr9(8) Program counter (10:9) = PA1:PA0	3	101k kkkk kkkk	none
NOP	No Operation	1	0000 0000 0000	none
RET	Return from subroutine (program counter = top-of-stack)	3	0000 0000 1100	none
RETP	Return from subroutine across Page boundary (PA1:PA0 = top-of-stack (10:19) and program counter = top-of-stack)	3	0000 0000 1101	PA1, PA0
RETI	Return from Interrupt (restore W, STATUS, FSR, and program counter from shadow registers)	3	0000 0000 1110	All STATUS except TO, PD
RETIW	Return from Interrupt and add W to RTCC (restore W, STATUS FSR, and program counter from shadow registers; and add W to RTCC)	3	0000 0000 1111	All STATUS except TO, PD
RETW lit	Return from Subroutine with Liter in W (W = lit and program counter = top-of-stack)	3	1000 kkkk kkkk	none

Table 17-6: Native SX Instruction Set: System Control Instructions				
Mnemonic, Operands	**Description**	**Clock Cycles**	**Opcode**	**Flags Affected**
BANK addr8	Load Bank number into FSR(7:5) FSR(7:5) = addr8(7:5)	1	0000 0001 1nnn	none
IREAD	Read word from Instruction memory MODE:W = data at (MODE:W)	4	0000 0100 0001	none
PAGE addr12	Load Page number into STATUS(7:5) STATUS(7:5) = addr12(11:9)	1	0000 0001 0nnn	PA1, PA0
SLEEP	Power down mode WDT = 00h, TO = 1, stop oscillator (PD = 0, clears prescaler if assigned)	1	0000 0000 0011	TO, PD

17.1. Equivalent Assembler Mnemonics

Some assemblers support additional instruction mnemonics that are special cases of existing instructions or alternative mnemonics for standard ones. For example, an assembler might support the mnemonic "CLC" (clear carry), which is interpreted the same as the instruction "clrb $03.0" (clear bit 0 in the STATUS register). Some of the commonly supported equivalent assembler mnemonics are described in Table 17-7.

Table 17-7: SX Equivalent Assembler Mnemonics

Syntax	Description	Equivalent	Cycles
CLC	Clear Carry bit	CLRB $03.0	1
CLZ	Clear Zero bit	CLRB $03.2	1
JMP W	Jump Indirect W	MOV $02,W	3
JMP PC+W	Jump Indirect W Relative	ADD $02,W	3
MODE imm4	Move Immediate to MODE Register	MOV M,#lit	1
NOT W	Complement of W	XOR W,#$FF	1
SC	Skip if Carry bit set	SB $03.0	1 or 2 (see Note 1)
SKIP	Skip Next Instruction	SNB $02.0 or SB $02.0	1 or 2 (see Note 1)

Note 1: The SC instruction takes 1 clock cycle if the tested condition is false or 2 cycles if the tested condition is true.

18.0 ELECTRICAL CHARACTERISTICS

18.1. Absolute Maximum Ratings

Stresses in excess of the absolute maximum ratings can cause permanent damage to the device. These are absolute stress ratings only. Functional operation of the device is not implied at these or any other conditions in excess of those given in the remainder of Section 18.0. Exposure to absolute maximum ratings for extended periods can adversely affect device reliability.

Table 18-1: Absolute Maximum Ratings (beyond which permanent damage may occur)	
Ambient temperature under bias	-40 °C to +85 °C
Storage temperature	-65 °C to +150 °C
Voltage on V_{dd} with respect to V_{ss}	0 V to +7.0 V
Voltage on OSC1 with respect to V_{ss}	0 V to +13.5 V
Voltage on $\overline{MCLR}$ with respect to V_{ss}	0 V to +13.5 V
Voltage on all other pins with respect to V_{ss}	-0.6 V to (V_{dd} + 0.6V)V
Total power dissipation	1 W at 70 °C; 1.5 W at 25 °C
Maximum current out of V_{ss} pins	180 mA at 70 °C; 300 mA at 25 °C
Maximum current into V_{dd} pins	180 mA at 70 °C; 300 mA at 25 °C
Maximum DC current into an input pin with internal protection diode forward biased	±500 µA
Input clamp current, I_{ik} (V_i < 0 or Vi > V_{dd})	±20 mA
Output clamp current, I_{ok} (VO < 0 or VO > V_{dd})	±20 mA
Maximum allowable sink current per I/O pin	45 mA
Maximum allowable source current per I/O pin	45 mA
Maximum allowable sink current per group of I/O pins between V_{dd} pins	50 mA
Maximum allowable source current per group of I/O pins between V_{dd} pins	50 mA
Latchup	200 mA
θ_{JA}, 48-pin Package	85 °C/W
Number of EEPROM Write Cycles	10,000
ESD Human Body Model – all pins	2000 V
ESD Machine Model – all pins	200 V

18.2. DC Characteristics

SX48BD running at 50 MHz: Operating Temperature - 40 °C <= Ta <= +85 °C (Industrial)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{dd}	Supply Voltage	F_{osc} = 0-75 MHz F_{osc} = 0-50 MHz F_{osc} = 0-32 MHz (Note 1)	4.5 3.0 2.2	- - -	5.5 5.5 5.5	V V V
S_{Vdd}	V_{dd} rise rate to ensure Power-On Reset (Note 2)		0.05	-	-	V/ms
I_{dd}	Supply Current, active Crystal oscillator (Note 3)	V_{dd} = 5.0 V, F_{osc} = 75 MHz V_{dd} = 5.0 V, F_{osc} = 50 MHz V_{dd} = 5.0 V, F_{osc} = 20 MHz V_{dd} = 5.0 V, F_{osc} = 4 MHz	- - - -	106 82 16 7.6	150 110 20 10	mA mA mA mA
I_{pd}	Supply Current, power down	V_{dd} = 5.0 V, WDT disabled and SLEEPCLK disabled V_{dd} = 5.0 V, WDT disabled WDT enabled and SLEEPCLK disabled V_{dd} = 3.0 V, WDT enabled WDT disabled and SLEEPCLK disabled V_{dd} = 3.0 V, WDT disabled WDT enabled and SLEEPCLK disabled	-	1 50 <1 10	300 400 20 50	µA µA µA µA
V_{ih}, V_{il}	Input Levels MCLR, RTCC Logic High Logic Low OSC1 Logic High Logic Low All Other Inputs CMOS Logic High Logic Low TTL Logic High Logic Low		 0.9 V_{dd} V_{ss} 0.7 V_{dd} V_{ss} 0.7 V_{dd} V_{ss} 2.0 V_{ss}	 V_{dd} 0.1 V_{dd} V_{dd} 0.3 V_{dd} V_{dd} 0.3 V_{dd} V_{dd} 0.8		 V V V V V V V V
I_{il}	Input Leakage Current	V_{in} = V_{dd} or V_{ss} (Note 4)	-3.0		+3.0	µA
I_{ip} I_{pup}	Weak Pullup Current	V_{dd} = 5.5 V, V_{in} = 0 V V_{dd} = 3.0 V, V_{in} = 0 V	300 80	430 20	600 200	µA µA
V_{oh}	Output High Voltage Ports B, C, D, E Port A	 I_{oh} = 16 mA, V_{dd} = 4.5 V I_{oh} = 12 mA, V_{dd} = 3.0 V I_{oh} = 30 mA, V_{dd} = 4.5 V I_{oh} = 18 mA, V_{dd} = 3.0 V	 V_{dd}-0.7 V_{dd}-0.7 V_{dd}-0.7 V_{dd}-0.7			 V V V V
V_{ol}	Output Low Voltage All Ports	I_{ol} = 30 mA, V_{dd} = 4.5 V I_{ol} = 18 mA, V_{dd} = 3.0 V			0.6 0.6	V V

Note 1: In-system programming is guaranteed for V_{dd} of 2.7 V to 5.5 V.

Note 2: V_{dd} must start rising from V_{ss} to ensure proper Power-On-Reset when relying on the internal Power-On-Reset Circuitry.

Note 3: Not floating inputs. Ports RA4 – RA7 are programmed as outputs.

Note 4: The FUSE register contains 0AAh.

18.3. AC Characteristics

SX48BD running at 50 MHz: Operating Temperature - 40 °C <= Ta <= +85 °C (Industrial)

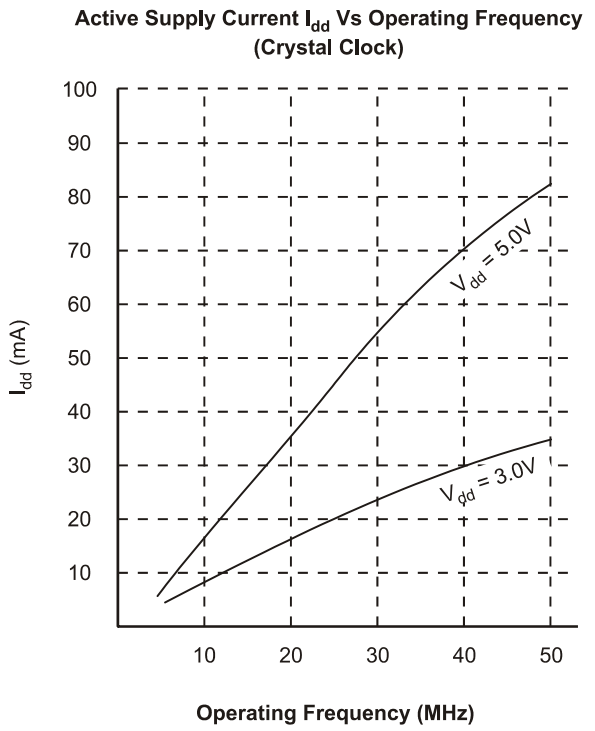
Symbol	Parameter	Min	Typ	Max	Units	Condition
F_{osc}	External CLKIN Frequency	DC	-	32	kHz	LP1
				1.0	MHz	LP2
				4.0	MHz	RC
				1.0	MHz	XT1
				8.0	MHz	XT2
				50.0	MHz	HS1/HS2/HS3
				75.0	MHz	HS3
	Oscillator Frequency (Crystal/Resonator)	DC	-	32	kHz	LP1
		0.032		1.0	MHz	LP2
		DC		4.0	MHz	RC
		0.032		1.0	MHz	XT1
		1.0		8.0	MHz	XT2
		1.0		50.0	MHz	HS1/HS2/HS3
		1.0		75.0	MHz	HS3
T_{osc}	External CLKIN Period	31.25	-	-	µs	LP1
		1.0			µs	LP2
		250.0			ns	RC
		1.0			µs	XT1
		125.0			ns	XT2
		20.0			ns	HS1/HS2/HS3
		13.3			ns	HS3
	Oscillator Period (Crystal/Resonator)	31.25	-	-	µs	LP1
		1.0		31.25	µs	LP2
		250.0		-	ns	RC
		1.0		31.25	µs	XT1
		125.0		1000.0	ns	XT2
		20.0		1000.0	ns	HS1/HS2/HS3
		13.3		-	ns	HS3
T_{osL}, T_{osH}	Clock in (OSC1) Low or High Time	400	-	-	µs	LP1/LP2
		50			µs	XT1/XT2
		8.0			µs	HS1/HS2/HS3
		5.3			µs	HS3

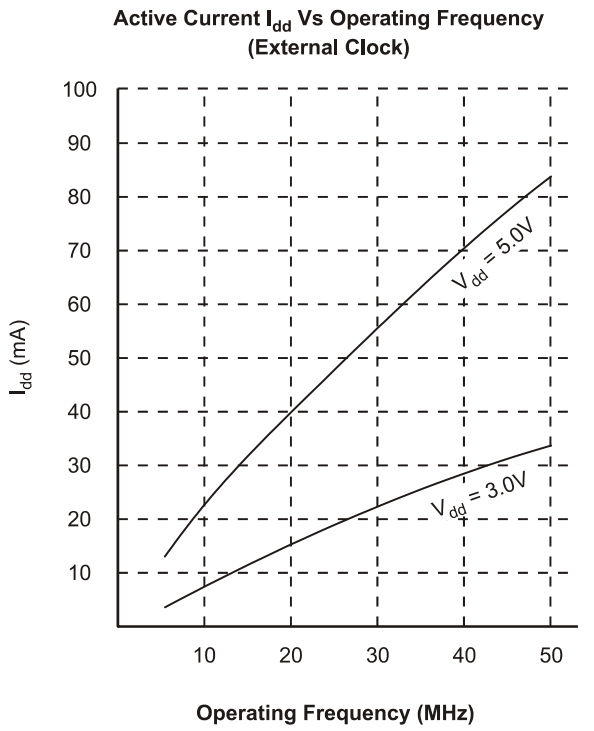
Note: Data in the Typical ("TYP") column is at 5 V, 25 °C unless otherwise stated.

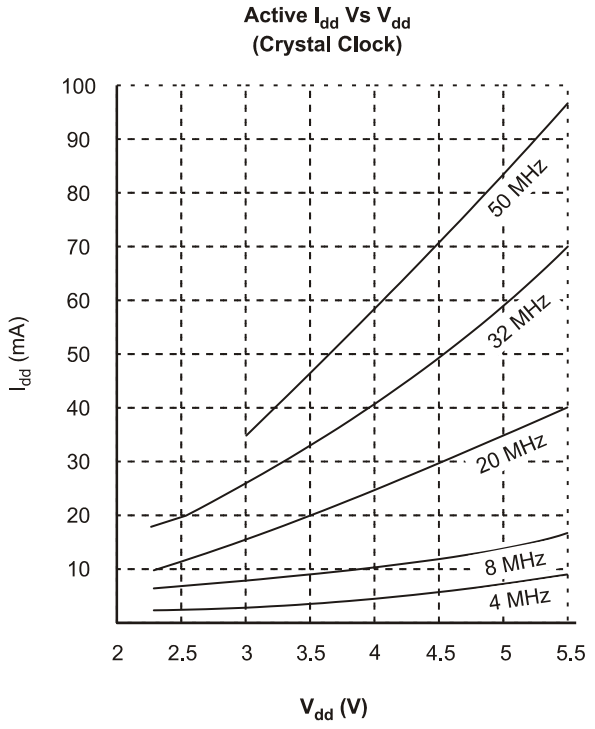
18.4. Comparator DC and AC Specifications

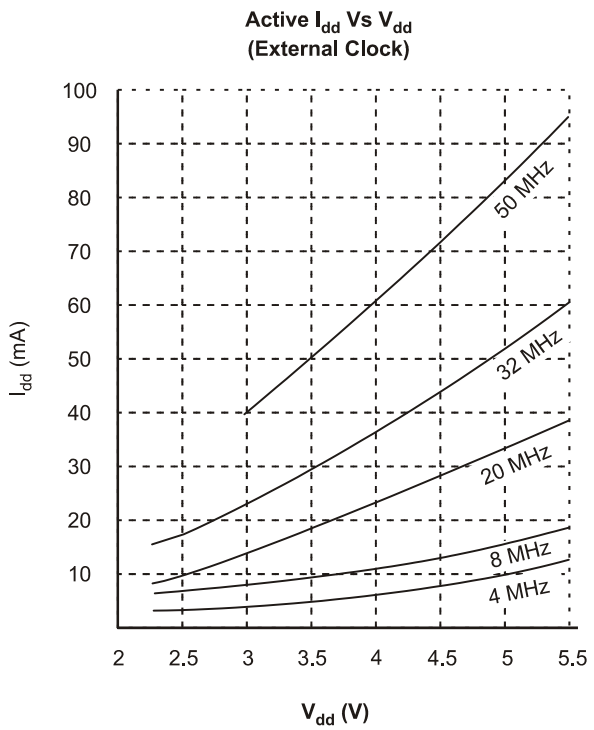
Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage	0 V < V_{in} < V_{dd} – 1.5 V		+/- 10	+/- 25	mV
Input Common Mode Voltage Range		0		V_{dd}	V
Voltage Gain			300 k		V/V
DC Supply Current (enabled)	V_{dd} = 5.5 V			120	µA
Response Time	$V_{overdrive}$ = 25 mV			250	ns

18.5. Typical Performance Characteristics (25 °C)

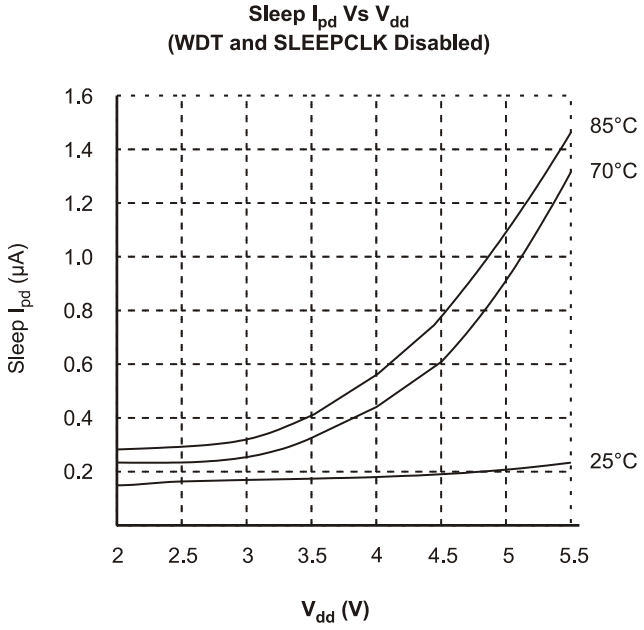


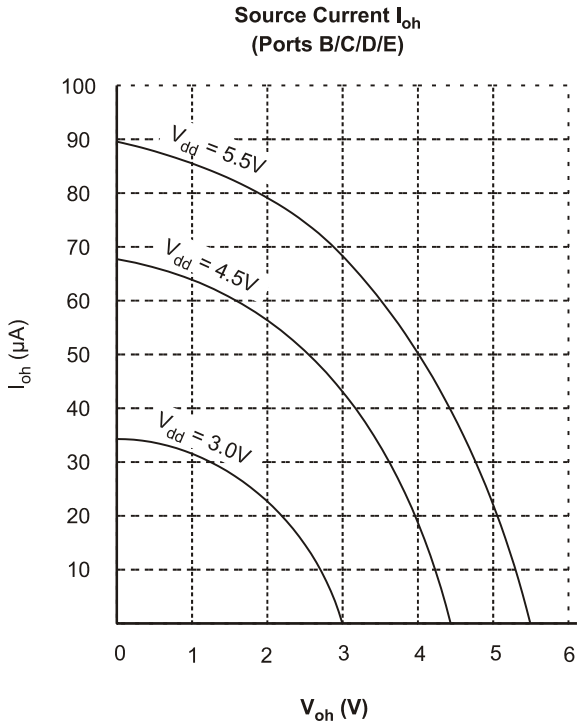


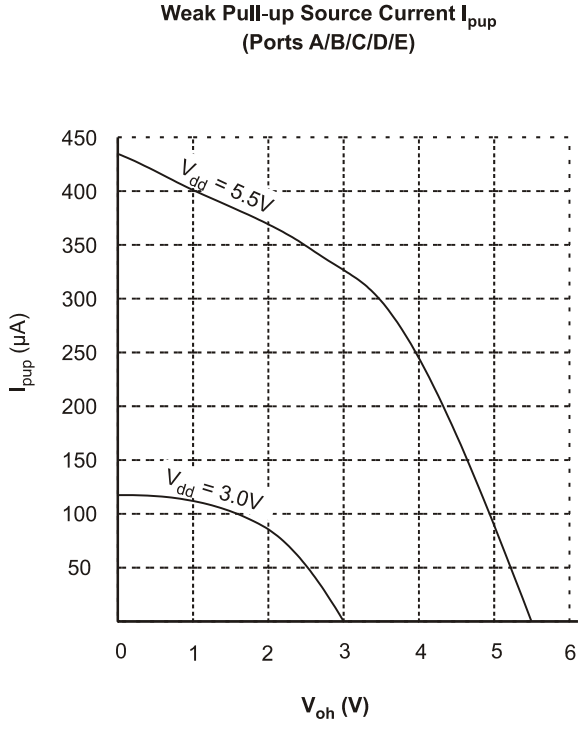


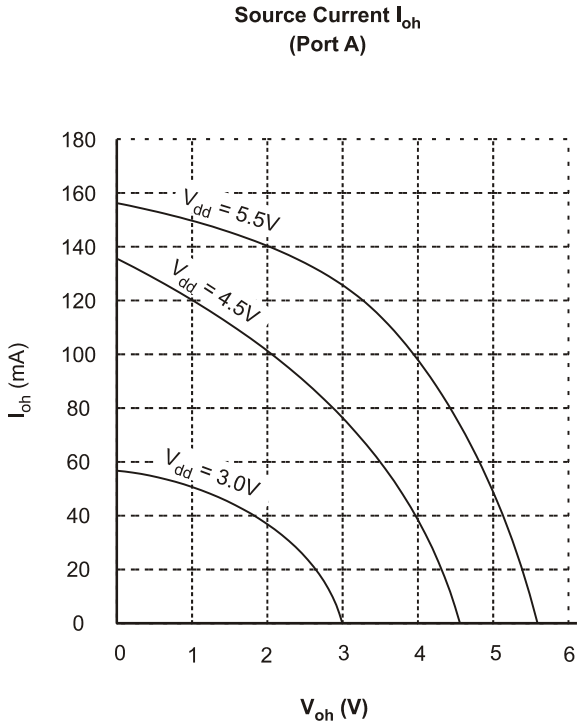


17.5. Typical Performance Characteristics (25 °C) (Continued)



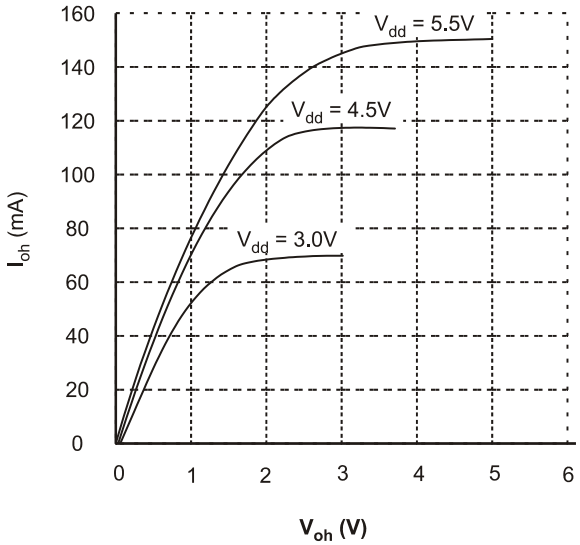




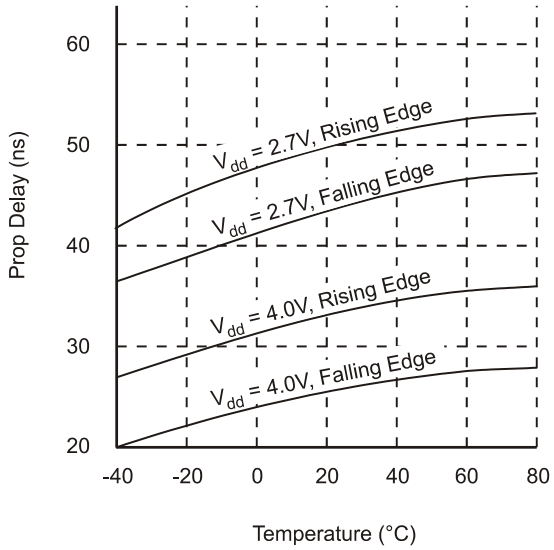


17.5. Typical Performance Characteristics (25 °C) (Continued)

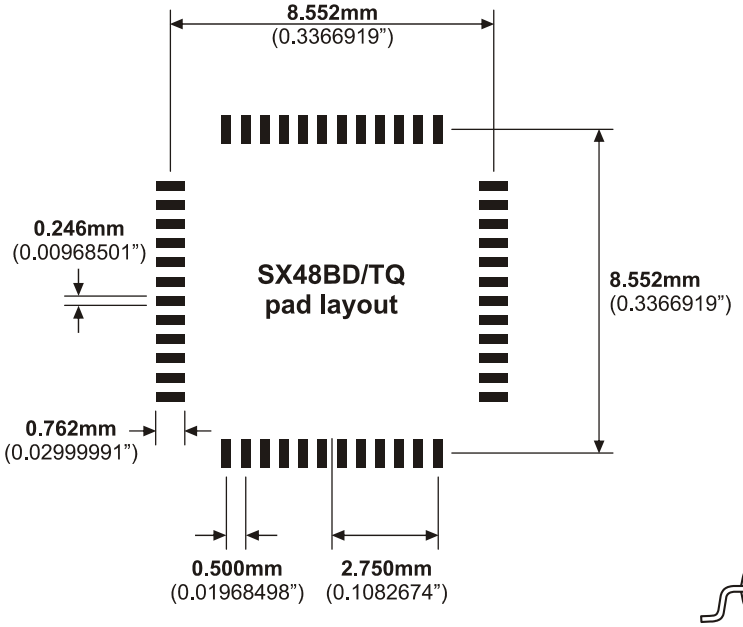
Sink Current I_{ol} (Ports A/B/C/D/E)

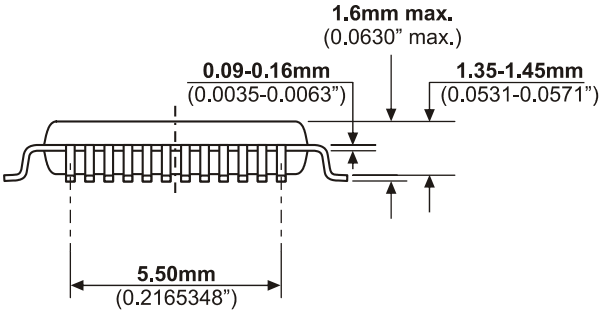


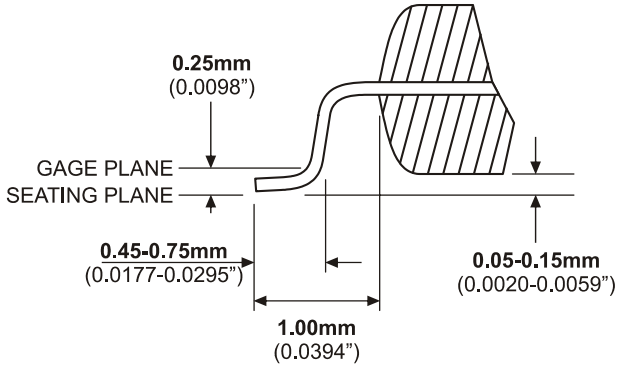
Maximum Propagation Delay Vs Temperature and V_{dd}

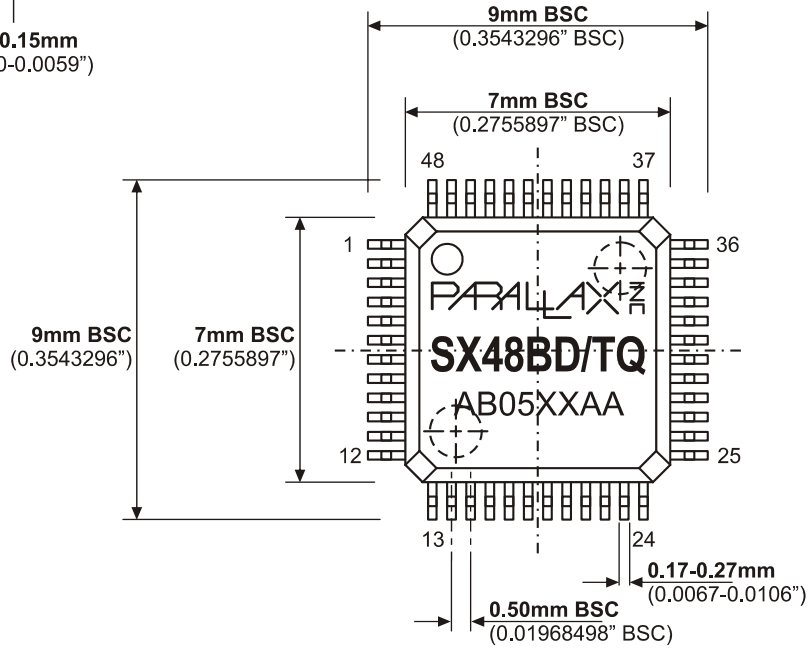


19.0 PACKAGE DIMENSIONS









20.0 MANUFACTURING INFORMATION

20.1. Reflow Peak Temperature

Package Type	Reflow Peak Temp.
Leaded	235+5/-0 °C
Green/RoHS	255+5/-0 °C

20.2. MSL3 Compliance

Chips shipped in production quantities are MSL3 compliant. For chips shipped in sample quantities or stored in compromised packaging, you may want to remove excess moisture before assembly by baking at 93 °C for 12 hours immediately before commencing soldering production.

20.3. Green/RoHS Compliance

All SX part numbers ending in "-G" are certified Green/RoHS Compliant. The Certificates of Compliance are appended to this datasheet; full test reports can be obtained by contacting the Parallax Sales Team.

20.4. Stress Testing Data Summary

The Parallax SX chip is packaged by a different supplier than previously used for the Ubicom SX. For this reason Parallax engaged in a series of stress tests useful for a change of packaging suppliers. Parallax SX chips packaged at Greatek in Taiwan were exposed to these stresses between June and September 2006. The stresses were provided by a separate stress testing reliability qualification services firm (Nano Measurements in Santa Clara, California).

The stress tests chosen are common for a change of packaging suppliers. The stress testing firm conducted the tests in accordance with the applicable Joint Electron Device Engineering Council (www.jedec.com) guidelines.

The following stresses were chosen:

- Preconditioning, Level 3, 192 hours 30° C, 60% RH, preflow peak leaded and unleaded (depending on the package)
- JESD22-A118, unbiased HAST 192 hours, 110°-130° C, 85% RH, 10-20 PSIG
- JESD22-A102, autoclave, 121° C, 100% RH, 15 PSIG, non-biased, 96 hours

RoHS-compliant and standard chips were exposed to the stresses separately to further identify if failures would be associated with a particular package type. Between each stress, the parts were sent back to Parallax for our standard production test which exercises the I/O, RAM and EE/Flash memory, operating voltage and current draw.

The results are that 100% of the chips passed the Parallax standard production test, in between and after the stresses were applied to the chips.

Stock Code	Tested	Passed
SX20AC/SS-G	20	20
SX28AC/SS-G	20	20
SX28AC/DP-G	17	17
SX48BD-G	20	20
Subtotal	**77**	**77**
SX20AC/SS-G	20	20
SX28AC/SS-G	20	20
SX28AC/DP-G	17	17
SX48BD-G	20	20
Subtotal	**77**	**77**

Other stress test data are available upon request from the Parallax Research and Development team, contact Parallax Sales or Tech Support.

Parallax Sales and Tech Support Contact Information

For the latest information on SX programming tools, development boards, compilers, instructional materials, and application examples, please visit www.parallax.com/sx.

Parallax, Inc.
599 Menlo Drive, Suite 100
Rocklin, CA 95765

Sales/Tech Support: (916) 624-8333
Toll Free in the US: 1-888-512-1024
Sales: sales@parallax.com
Tech Support: support@parallax.com

GREATEK ELECTRONICS INC
136,Gung-Yi-Rd.,Chunan,Cheng,Miaoli
Hsien,350-Taiwan,R.O.C.
TEL:-037-638568-Ext:5104
Fax:-037-628323

To：Parallax Inc.
599 Menlo Drive, Suite 100 Rocklin, CA 95765, USA

Supplier Company：GREATEK ELECTRONICS INC (seal)：

Printed Name：Steven Su Signature：STEVEN SU

Title：QA Manager Date：2005/12/14

Warranty for Non-Inclusion of Hazardous Substances in Products

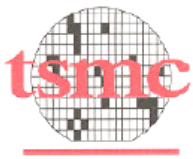
Our company (including subsidiaries, affiliates and suppliers) hereby warrants and guarantees that the below Part Number of **Green** products, parts, and packing materials, made for or delivered to your company directly or indirectly by our company and the manufacture process are free from the restricted substances and/or in compliance with the requirements listed as below. (Package only focus on Lead-frame base).

Parallax Part Number:	Package type：
SX20AC/SS-G	SSOP 20 209 mil
SX28AC/SS-G	SSOP 28 209mil
SX28AC/DP-G	P-DIP 28 300mil
SX48BD/TQU-G	LQFP 48 (7x7mm)

a. Cadmium (Cd) and cadmium compounds...	< 5	ppm
b. Lead (Pb) and lead compounds...	< 100	ppm
c. Mercury (Hg) and mercury compounds... ...	0	ppm
d. Hexavalent chromium (Cr^{6+}) compounds...	0	ppm
e. Polybrominated biphenyls (PBB)...	0	ppm
f. Polybrominated diphenylethers (PBDE)...	0	ppm

Packing Materials : Cadmium,Lead, Chuomium(VI), Hg and its compounds<100ppm

1/1



Taiwan Semiconductor Manufacturing Company, Ltd.
121, Park Ave. 3, Science-Based Industrial Park, Hsin-Chu, Taiwan 300, R.O.C.
Tel: 886-3-5780221 Fax:886-3-5781546

Date: 01/21/2005
To: Whom it may concern
Fm: Chung Yi Huang

Manager of Environment, Safety & Hygiene Strategic Planning Department

Subj: Certificate of non-use of RoHS prohibited substances

We hereby certify that all materials, used for products and packaging materials in TSMC manufacturing process, do not contain hazardous substances which are prohibited by the Directive of the European Parliament and of the Council on the Restriction of the use of certain Hazardous Substances in electrical and electronic equipment (RoHS), including **Lead, Cadmium, Mercury, Hexavalent Chromium, Polybrominated Biphenyls (PBB) and Polybrominated Diphenyl Ethers (PBDE)**.

Contact Person: Minglien Lo
E-mail: mlloa@tsmc.com
Tel: +886-3-5780221 ext 2242