

1 PRODUCT OVERVIEW

OVERVIEW

The S3C7544 single-chip CMOS microcontroller is designed for high-performance using Samsung's newest 4-bit CPU core, SAM47 (Samsung Arrangeable Microcontrollers).

With a versatile 8-bit timer/counter and a D/A converter, the S3C7544 offers an excellent design solution for a wide variety of telecommunication applications.

Up to 17 pins of the 24-pin SDIP package can be dedicated to I/O. Four vectored interrupts provide fast response to internal and external events. In addition, the S3C7544's advanced CMOS technology has realized substantially lower power consumption with a wide operating voltage range — all at a substantially lower cost.

OTP

The S3C7544 microcontroller is also available in OTP (One Time Programmable) version, S3P7544. S3P7544 microcontroller has an on-chip 4-Kbyte one-time-programmable EPROM instead of masked ROM. The S3P7544 is comparable to S3C7544, both in function and in pin configuration.

FEATURES SUMMARY

Memory

- 512 × 4-bit RAM
- 4096 × 8-bit ROM

I/O Pins

- 17 pins I/O
- N-channel open-drain I/O: 8 pins

8-Bit Basic Timer

- Programmable interval timer
- Watchdog timer

Interval 8-Bit Timer/Counter

- Programmable interval timer
- External event counter function
- Timer/counter clock output to TCLO0 pin

Buzzer Output

- Four frequency output to BUZ pin

D/A Converter

- 8-bit D/A converter

Interrupts

- Two external interrupt vectors
- Two internal interrupt vectors
- One quasi-interrupt

Memory-Mapped I/O Structure

- Data memory bank 15

Bit Sequential Carrier

- Supports 16-bit serial data transfer in arbitrary format

Power-Down Modes

- Idle mode (only CPU clock stops)
- Stop mode (system clock stops)

Oscillation Sources

- Crystal, or ceramic for system clock
- Crystal, ceramic: 0.4–6.0 MHz
- CPU clock divider circuit (by 4, 8, or 64)

Instruction Execution Times

- 0.95, 1.91, and 15.3 μ s at 4.19 MHz
- 0.67, 1.33, 10.7 μ s at 6.0 MHz

Operating Temperature

- –40 °C to 85 °C

Operating Voltage Range

- 1.8 V to 5.5 V (at 3 MHz)
- 2.7 V to 5.5 V (at 6 MHz)

Package Types

- 24-pin SOP-375
- 24-pin SDIP-300

BLOCK DIAGRAM

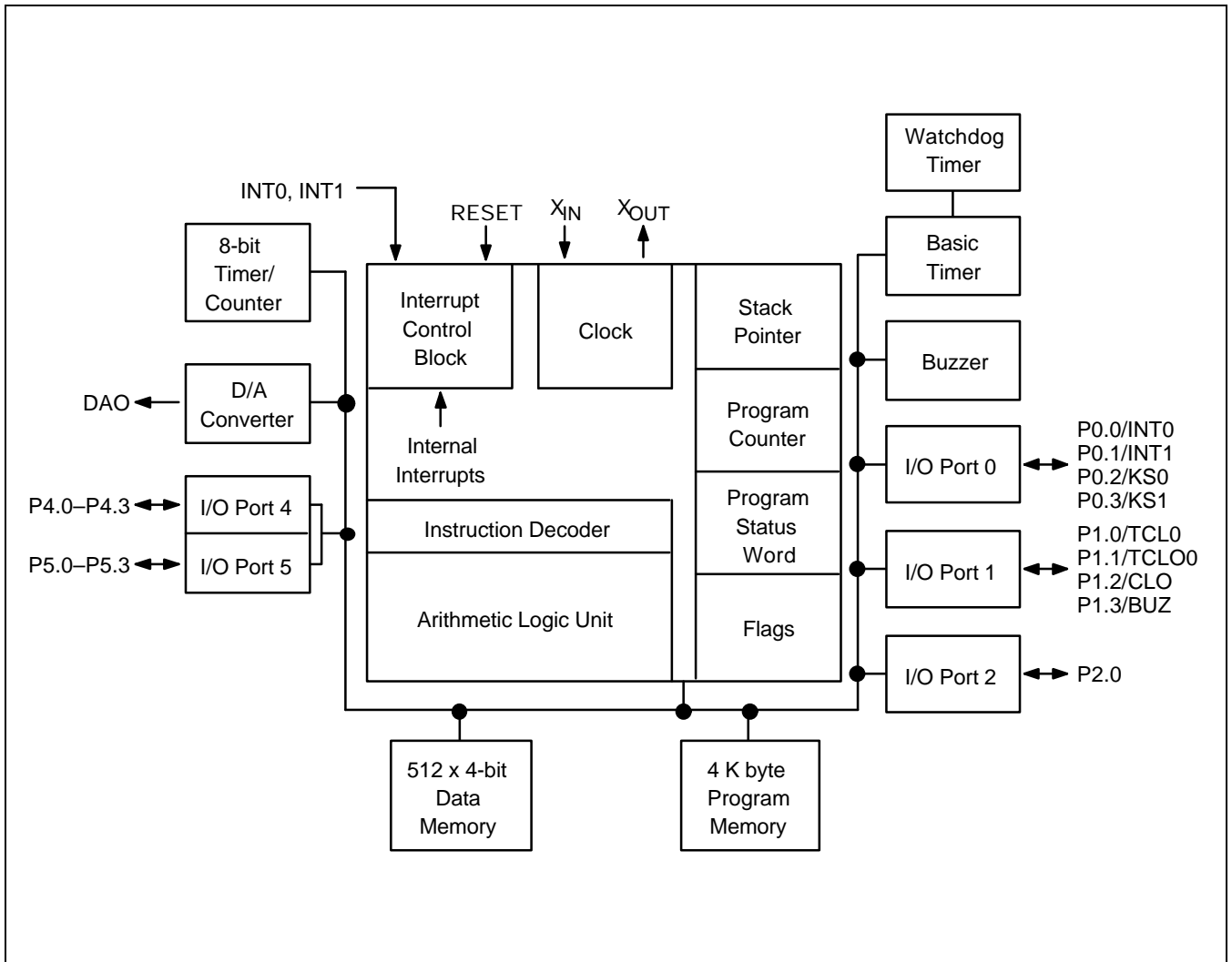


Figure 1-1. S3C7544 Simplified Block Diagram

PIN ASSIGNMENTS

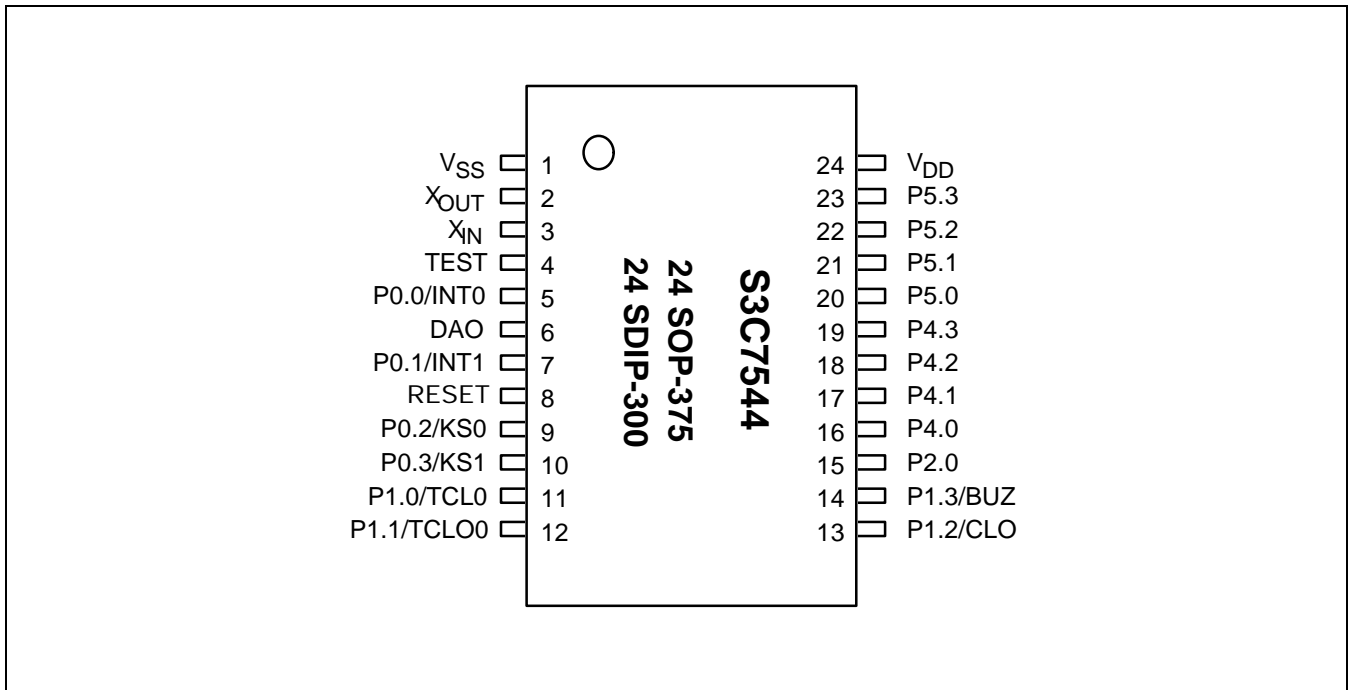


Figure 1-2. S3C7544 Pin Assignment Diagrams

PIN DESCRIPTIONS

Table 1-1. S3C7544 Pin Descriptions

Pin Name	Pin Type	Description	Share Pin
P0.0 P0.1 P0.2 P0.3	I	4-bit I/O port. 1- or 4-bit read/write and test is possible. Pull-up resistors are assignable to input pins by software and are automatically disabled for output pins. Pins are individually configurable as input or output.	INT0 INT1 KS0 KS1
P1.0 P1.1 P1.2 P1.3	I/O	4-bit I/O port. 1- or 4-bit read/write and test is possible. Pull-up resistors are assignable to input pins by software and are automatically disabled for output pins. Pins are individually configurable as input or output.	TCL0 TCLO0 CLO BUZ
P2.0	I/O	1-bit I/O port. 1- or 4-bit read/write and test is possible. Pull-up resistors are assignable to input pins by software and are automatically disabled for output pins.	–
P4.0–P4.3 P5.0–P5.3	I/O	4-bit I/O port. 1- or 4-bit read/write and test is possible. Pins are individually configurable as input or output. Pull-up resistors are assignable to input pins by software and are automatically disabled for output pins. The N-channel open drain or push-pull output can be selected by software (1-bit unit).	–
INT0	I/O	External interrupts with rising/falling edge detection	P0.0
INT1	I/O	External interrupts with rising/falling edge detection	P0.1
KS0 KS1	I/O	Quasi-interrupt input with falling edge detection	P0.2 P0.3
TCL0	I/O	External clock input for timer/counter	P1.0
TCLO0	I/O	Timer/counter clock output	P1.1
CLO	I/O	CPU clock output	P1.2
BUZ	I/O	0.5, 1, 2, or 4 kHz frequency output at 4.19 MHz for buzzer sound	P1.3
DAO	O	8-bit D/A converter output	–
V _{DD}	–	Main power supply	–
V _{SS}	–	Ground	–
RESET	I	Reset signal	–
TEST	I	Chip test input pin. Hold GND when the device is operating.	–
X _{IN} , X _{OUT}	–	Crystal, ceramic oscillator signal for system clock	–

Table 1-2. Overview of S3C7544 Pin Data

SDIP Pin Numbers	Share Pins	I/O Type	Reset Value	Circuit Type
V _{SS}	–	–	–	–
X _{OUT} , X _{IN}	–	–	–	–
TEST	–	I	–	–
P0.0, P0.1	INT0, INT1	I/O	Input	D-4
RESET	–	I	–	B
P0.2 P0.3	KS0 KS1	I/O	Input	D-4
P1.0 P1.1 P1.2 P1.3	TCL0 TCLO0 CLO BUZ	I/O	Input	D-2
P2.0	–	I/O	Input	D-2
DAO	–	O	Output	–
P4.0–P4.3	–	I/O	Input	E-2
P5.0–P5.3	–	I/O	Input	E-2
V _{DD}	–	–	–	–

PIN CIRCUIT DIAGRAMS

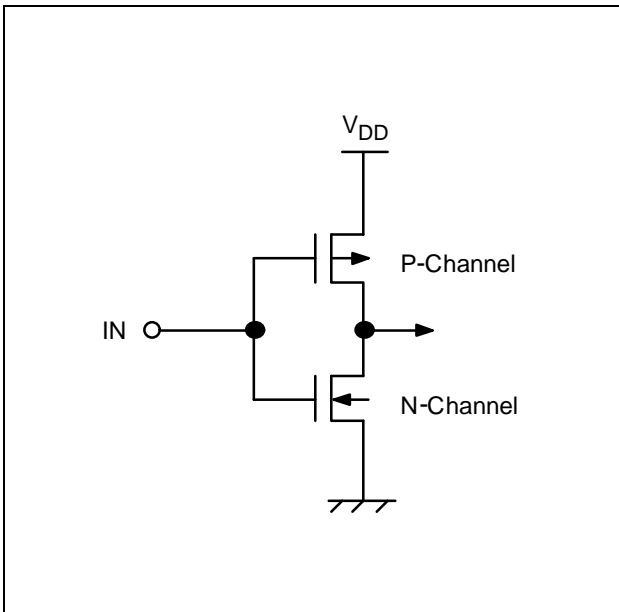


Figure 1-3. Pin Circuit Type A

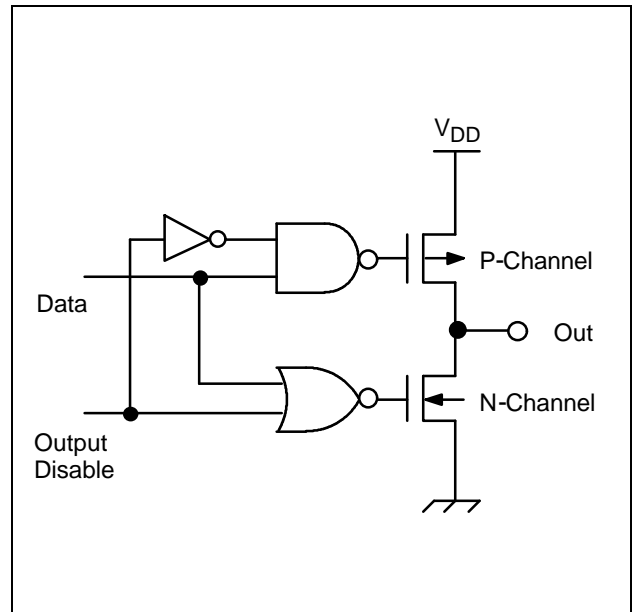


Figure 1-5. Pin Circuit Type C

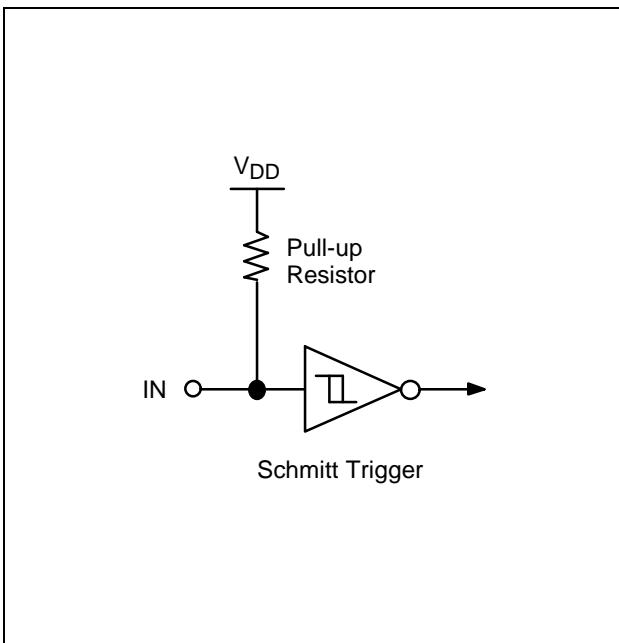


Figure 1-4. Pin Circuit Type B

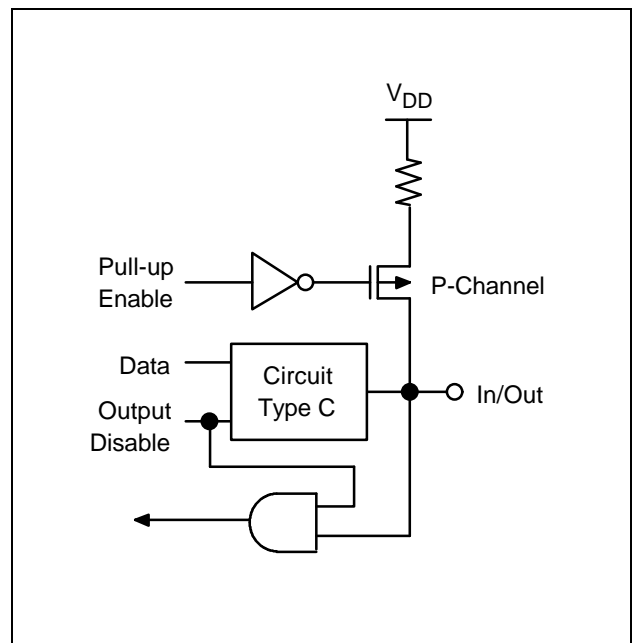


Figure 1-6. Pin Circuit Type D-2

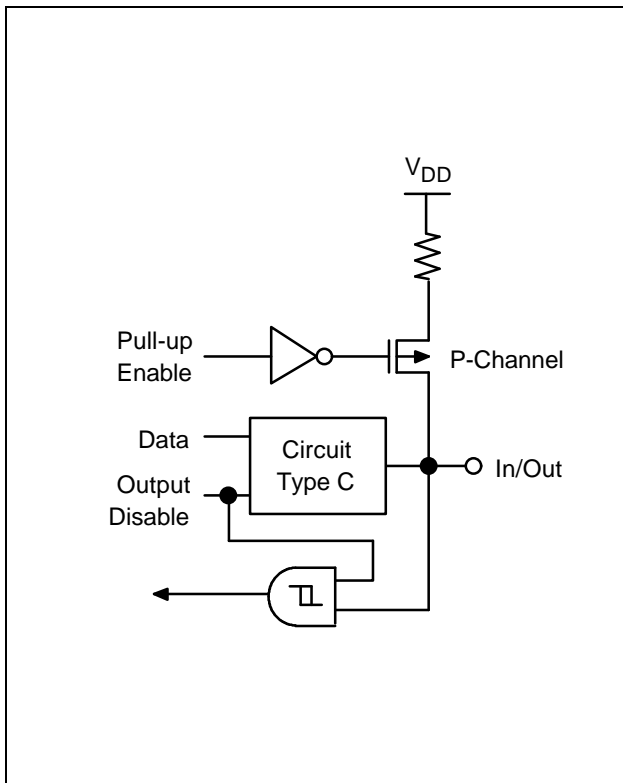


Figure 1-7. Pin Circuit Type D-4

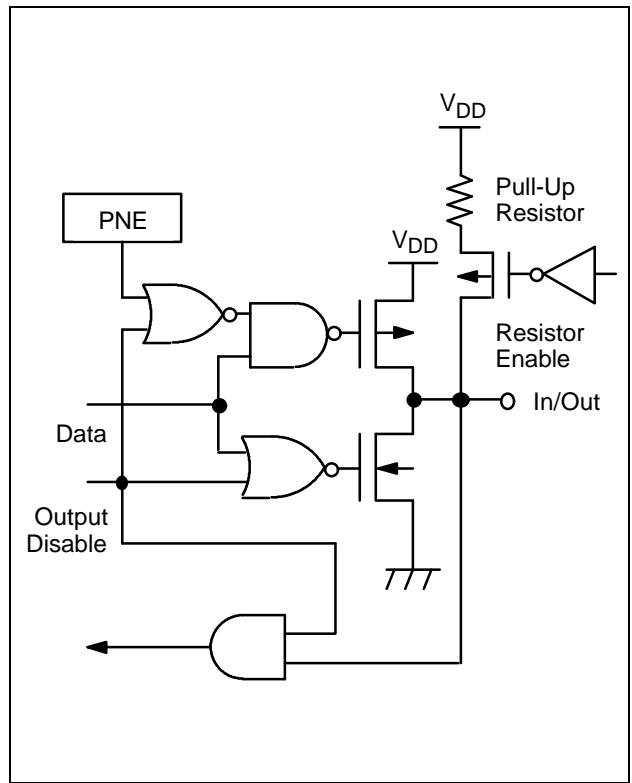


Figure 1-8. Pin Circuit Type E-2

2 ADDRESS SPACES

PROGRAM MEMORY (ROM)

OVERVIEW

ROM maps for the S3C7544 device are mask programmable at the factory. In its standard configuration, the device's 4096×8 -bit program memory has three areas that are directly addressable by the program counter (PC):

- 16-byte area for vector addresses
- 16-byte general-purpose area
- 96-byte instruction reference area
- 3968-byte general-purpose area

General-Purpose Memory

Two program memory areas are allocated for general-purpose use: One area is 16 bytes in size and the other is 3968 bytes.

Vector Addresses

You use the 16-byte vector address area to store the vector addresses required to execute system RESET and interrupts. Start addresses for interrupt service routines are stored in this area, along with the values of the enable memory bank (EMB) and enable register bank (ERB) flags that are used to set their initial value for the corresponding service routines. The 16-byte area can be used alternately as general-purpose ROM.

REF Instructions

Locations 0020H–007FH are used as a reference area (look-up table) for 1-byte REF instructions. Using REF instructions, you can reduce the byte size of instruction operands. REF can reference one 2-byte instruction, two 1-byte instructions, and three-byte instruction which are stored in the look-up table. Unused look-up table addresses can be used as general-purpose ROM.

Table 2-1. Program Memory Address Ranges

ROM Area Function	Address Ranges	Area Size (in Bytes)
Vector address area	0000H–000FH	16
General-purpose program memory	0010H–001FH	16
REF instruction look-up table area	0020H–007FH	96
General-purpose program memory	0080H–0FFFH	3968

GENERAL-PURPOSE MEMORY AREAS

The 16-byte area at the ROM locations 0010H–001FH and the 3968-byte area at the ROM locations 0080H–0FFFH are used as general-purpose program memory.

You can also use vacant locations in the vector address area and REF instruction look-up table areas as general-purpose program memory. But please be careful not to overwrite live data when writing programs that use special-purpose areas of the ROM.

VECTOR ADDRESS AREA

Use the 16-byte vector address area of the ROM to store the vector addresses for executing system resets and interrupts. The starting addresses of interrupt service routines are stored in this area, along with the enable memory bank (EMB) and enable register bank (ERB) flag values that are needed to set EMB and ERB's initial values for the service routines. A 16-byte vector address is organized as follows:

EMB	ERB	PC13 (note)	PC12 (note)	PC11	PC10	PC9	PC8
PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

NOTE: PC13, 12 are always logic 0.

To set up the vector address area for specific programs, you should use the instruction VENTn. The programming tips on the next page explain how to do this.

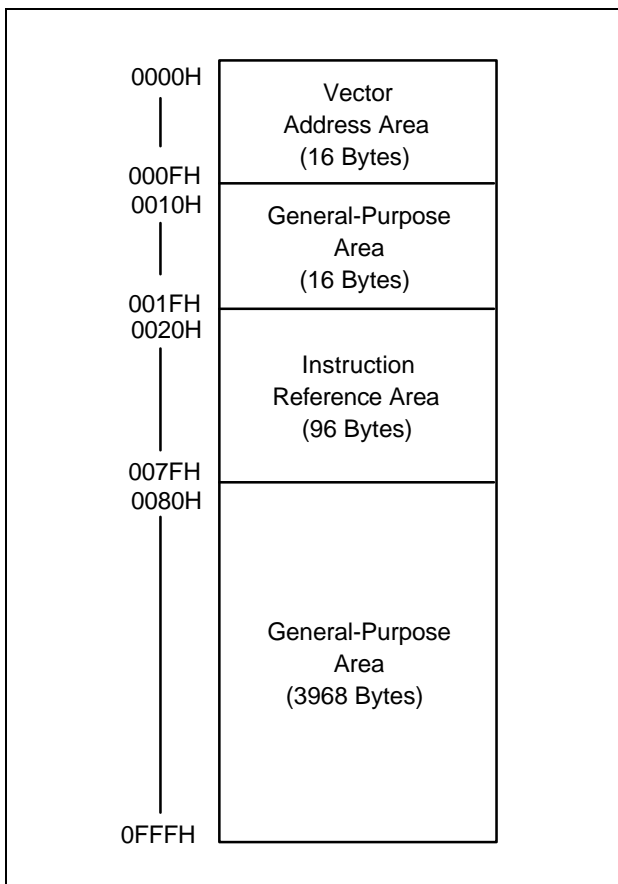


Figure 2-1. ROM Structure

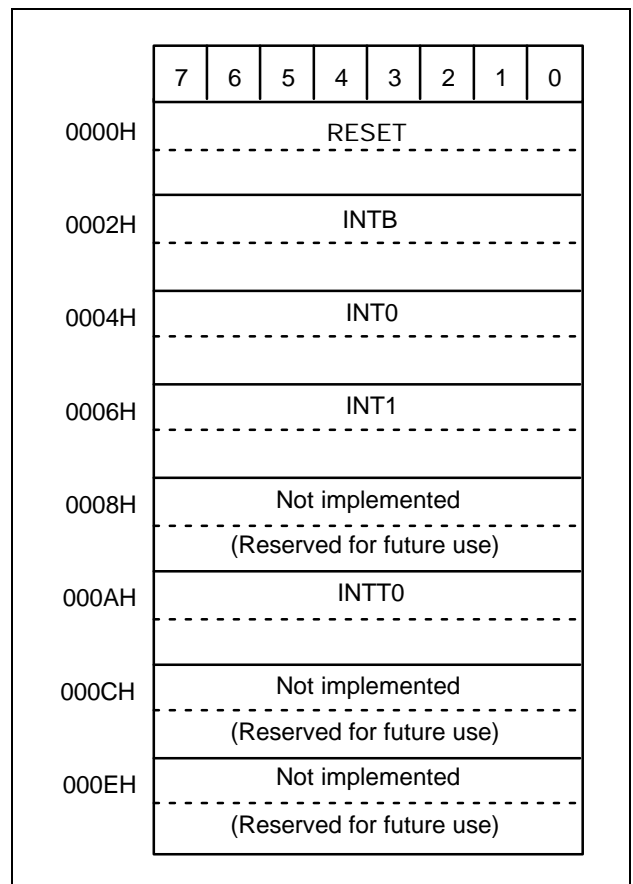


Figure 2-2. Vector Address Map

PROGRAMMING TIP — Defining Vectored Interrupt Areas

The following examples show you several ways you can define the vectored interrupt area in program memory:

1. When all vector interrupts are used:

```

;
;
;   ORG      0000H
;
;   VENT0    1,0,RESET      ; EMB ← 1, ERB ← 0; Jump to RESET address
;   VENT1    0,0,INTB      ; EMB ← 0, ERB ← 0; Jump to INTB address
;   VENT2    0,0,INT0      ; EMB ← 0, ERB ← 0; Jump to INT0 address
;   VENT3    0,0,INT1      ; EMB ← 0, ERB ← 0; Jump to INT1 address
;   NOP
;   NOP
;   VENT5    0,0,INTT0     ; EMB ← 0, ERB ← 0; Jump to INTT0 address

```

2. When a specific vectored interrupt such as INT0, and INTT0 is not used, the unused vector interrupt locations must be skipped with the assembly instruction ORG so that jumps will address the correct locations:

```

;
;
;   ORG      0000H
;
;   VENT0    1,0,RESET      ; EMB ← 1, ERB ← 0; Jump to RESET address
;   VENT1    0,0,INTB      ; EMB ← 0, ERB ← 0; Jump to INTB address
;   ORG      0006H          ; INT0 interrupt not used
;   VENT3    0,0,INT1      ; EMB ← 0, ERB ← 0; Jump to INT1 address
;
;
;   ORG      0010H          ; INTT0 interrupt not used

```

3. If an INT0 interrupt is not used and its corresponding vector interrupt area is not fully utilized, or if it is not written by an ORG instruction as in Example 2, a CPU malfunction will occur:

```

;
;
;   ORG      0000H
;
;   VENT0    1,0,RESET      ; EMB ← 1, ERB ← 0; Jump to RESET address
;   VENT1    0,0,INTB      ; EMB ← 0, ERB ← 0; Jump to INTB address
;   VENT3    0,0,INT1      ; EMB ← 0, ERB ← 0; Jump to INT1 address by INT0
;   VENT5    0,0,INTT0     ; EMB ← 0, ERB ← 0; Jump to INTT0 address by INT1
;
;
;   ORG      0010H
;
;   General-purpose ROM area
;
;

```

In this example, when an INT0 interrupt is generated, the corresponding vector area is not VENT2 INT0, but VENT3 INT1. This causes an INT0 interrupt to jump incorrectly to the INT1 address, causing a CPU malfunction.

INSTRUCTION REFERENCE AREA

Using 1-byte REF instructions, you can easily reference instructions with larger byte sizes that are stored in the addresses 0020H–007FH of program memory. This 96-byte area is called the REF instruction reference area, or look-up table. Locations in the REF look-up table may contain two one-byte instructions, a single two-byte instruction, or three-byte instructions such as a JP or CALL. The starting address of the instruction you are referencing must always be an even number. To reference a JP or CALL instruction, it must be written to the reference area in a two-byte format: for JP, this format is TJP; for CALL, it is TCALL. In summary, there are three ways to the REF instruction:

- Using the 1-byte REF instruction to execute one 2-byte or two 1-byte instructions,
- Branching to any location by referencing a branch instruction stored in the look-up table,
- Calling subroutines at any location by referencing a call instruction stored in the look-up table.

PROGRAMMING TIP — Using the REF Look-Up Table

Here is one example of how to use the REF instruction look-up table:

```

;          ORG      0020H
;
JMAIN     TJP      MAIN          ; 0, MAIN
KEYCK     BTSF    KEYFG         ; 1, KEYFG CHECK
WATCH    TCALL   CLOCK         ; 2, CALL CLOCK
INCHL    LD      @HL,A         ; 3, (HL) ← A
          INCS    HL
          .
          .
          .
ABC      LD      EA,#00H        ; 47, EA ← #00H
          ORG      0080
;
MAIN     NOP
          NOP
          .
          .
          .
          REF    KEYCK         ; BTSF KEYFG (1-byte instruction)
          REF    JMAIN         ; KEYFG = 1, jump to MAIN (1-byte instruction)
          REF    WATCH        ; KEYFG = 0, CALL CLOCK (1-byte instruction)
          REF    INCHL        ; LD @HL,A
          REF    INCS HL
          REF    ABC          ; LD EA,#00H (1-byte instruction)
          .
          .
          .

```

DATA MEMORY (RAM)

OVERVIEW

In its standard configuration, the 512×4 -bit data memory has five areas:

- 32×4 -bit working register area
- 224×4 -bit general-purpose area in bank 0 (also used as stack area)
- 256×4 -bit general-purpose area in bank 1
- 128×4 -bit area for memory-mapped I/O addresses

To simplify referencing, the data memory area has three memory banks — bank 0, bank 1 and bank 15. You should use the select memory bank instruction (SMB) to select the bank you want to use as working data memory. Data stored in RAM locations are 1-, 4-, and 8-bit addressable. Initialization values for the data memory area are not defined by hardware and must therefore be initialized by program software following a RESET. When a RESET signal is generated in power-down mode, the data memory contents are maintained.

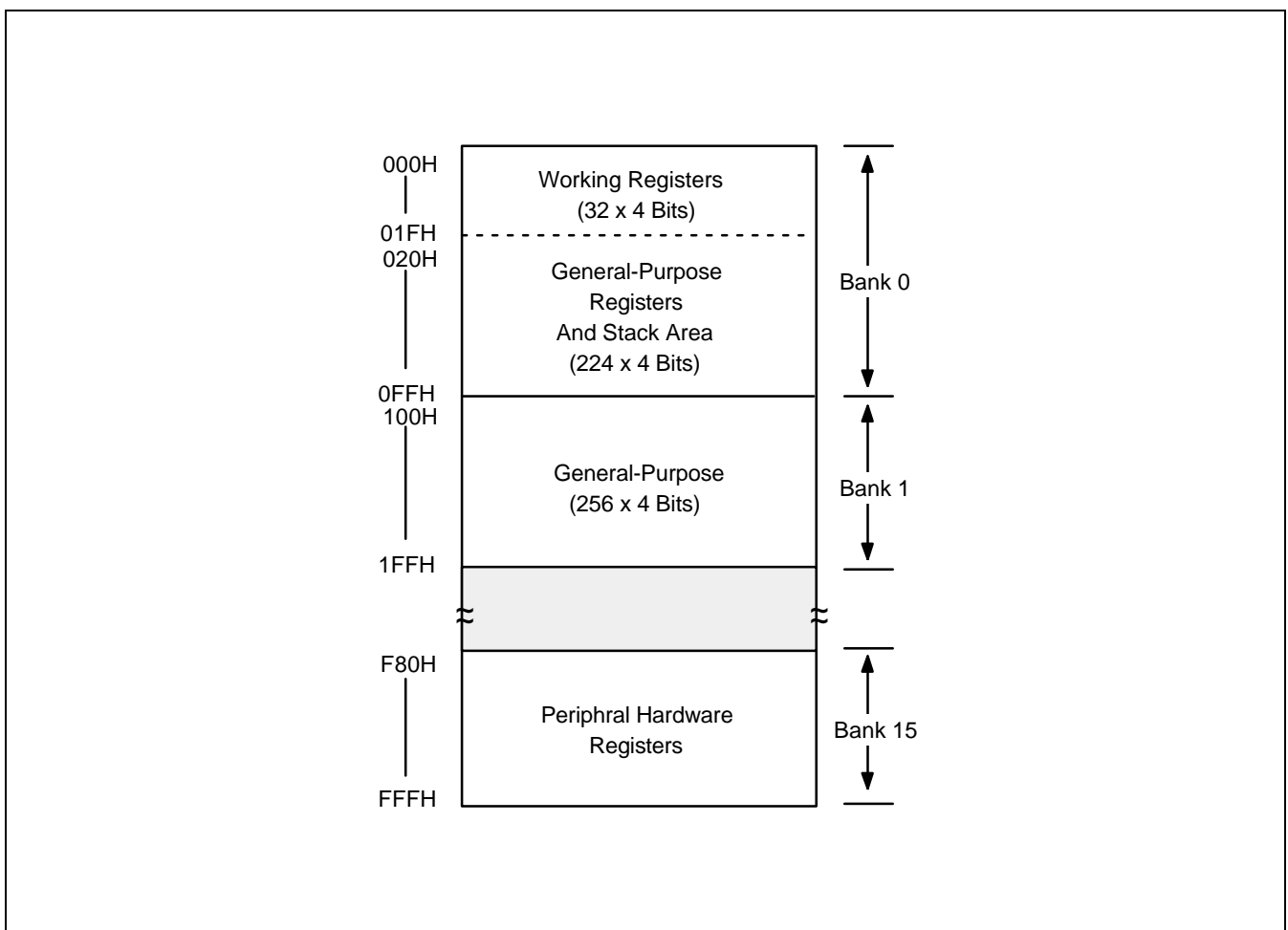


Figure 2-3. Data Memory (RAM) Map

Memory Banks 0, 1 and 15

Bank 0	(000H–0FFH)	The lowest 32 nibbles of bank 0 (000H–01FH) are used as working registers; the next 224 nibbles (020H–0FFH) can be used both as stack area and as general-purpose data memory. Use the stack area for implementing subroutine calls and returns, and for interrupt processing.
Bank 1	(100H–1FFH)	This area is used as general-purpose data memory.
Bank 15	(F80H–FFFH)	The microcontroller uses bank 15 for memory-mapped peripheral I/O. Fixed RAM locations for each peripheral hardware address are mapped into this area.

Data Memory Addressing Modes

The enable memory bank (EMB) flag controls the addressing mode for data memory bank 0 or 15. When the EMB flag is logic zero, the addressable area is restricted to specific locations, depending on whether direct or indirect addressing is used. With direct addressing, you can access the locations 000H–07FH of bank 0, bank 1 and bank 15. With indirect addressing, only bank 0 (000H–0FFH) can be accessed. When the EMB flag is set to logic one, three data memory banks can be accessed according to the current SMB value.

For 8-bit addressing, two 4-bit registers are addressed as a register pair. When using 8-bit instructions to address RAM locations, remember to use the even-numbered register address as the instruction operand.

Working Registers

The RAM working register area in data memory bank 0 is further divided into four *register* banks (bank 0, 1, 2, and 3). Each register bank has eight 4-bit registers and paired 4-bit registers are 8-bit addressable.

Register A is used as a 4-bit accumulator and the register pair EA is an 8-bit extended accumulator. The carry flag bit can also be used as a 1-bit accumulator. The register pairs WX, WL, and HL are used as address pointers for indirect addressing. To limit the possibility of data corruption caused by incorrect register addressing, it is advisable to use register bank 0 for the main program and banks 1, 2, and 3 for interrupt service routines.

Bit Sequential Carrier (BSC)

The bit sequential carrier (BSC) is a 16-bit general register mapped to the RAM addresses FC0H–FC3H that can be manipulated by 1-, 4-, and 8-bit RAM control instructions. A RESET clears all bit values to logic zero.

You can specify addresses and bit locations sequentially using a 1-bit indirect addressing instruction. In this way, a program can process 16-bit data by moving the bit location sequentially, incrementing or decrementing the value of the L register. BSC data can also be manipulated by direct addressing. For 8-bit manipulations, you must address the upper and lower 8 bits separately.

Table 2-2. Data Memory Organization and Addressing

Addresses	Register Areas	Bank	EMB Value	SMB Value
000H–01FH	Working registers	0	0, 1	0
020H–0FFH	Stack and general-purpose registers			
100H–1FFH	General-purpose registers	1	1	1
F80H–FFFH	I/O-mapped hardware registers	15	0, 1	15

 **PROGRAMMING TIP — Clearing Data Memory Banks 0 and 1**

Clear bank 0 of the data memory area:

```

RAMCLR  BITS      EMB
         SMB      0
         LD      HL,#10H
         LD      A,#0H
RMCL0   LD      @HL,A          ; RAM (010H–0FFH) clear
         INCS   HL
         JR     RMCL0
;

```

WORKING REGISTERS

Working registers, mapped to the RAM address 000H-01FH in data memory bank 0 are used to temporarily store intermediate results during program execution, as well as pointer values used for indirect addressing. Unused registers may be used as general-purpose memory. Working register data can be manipulated as 1-bit units, 4-bit units or, using paired registers, as 8-bit units.

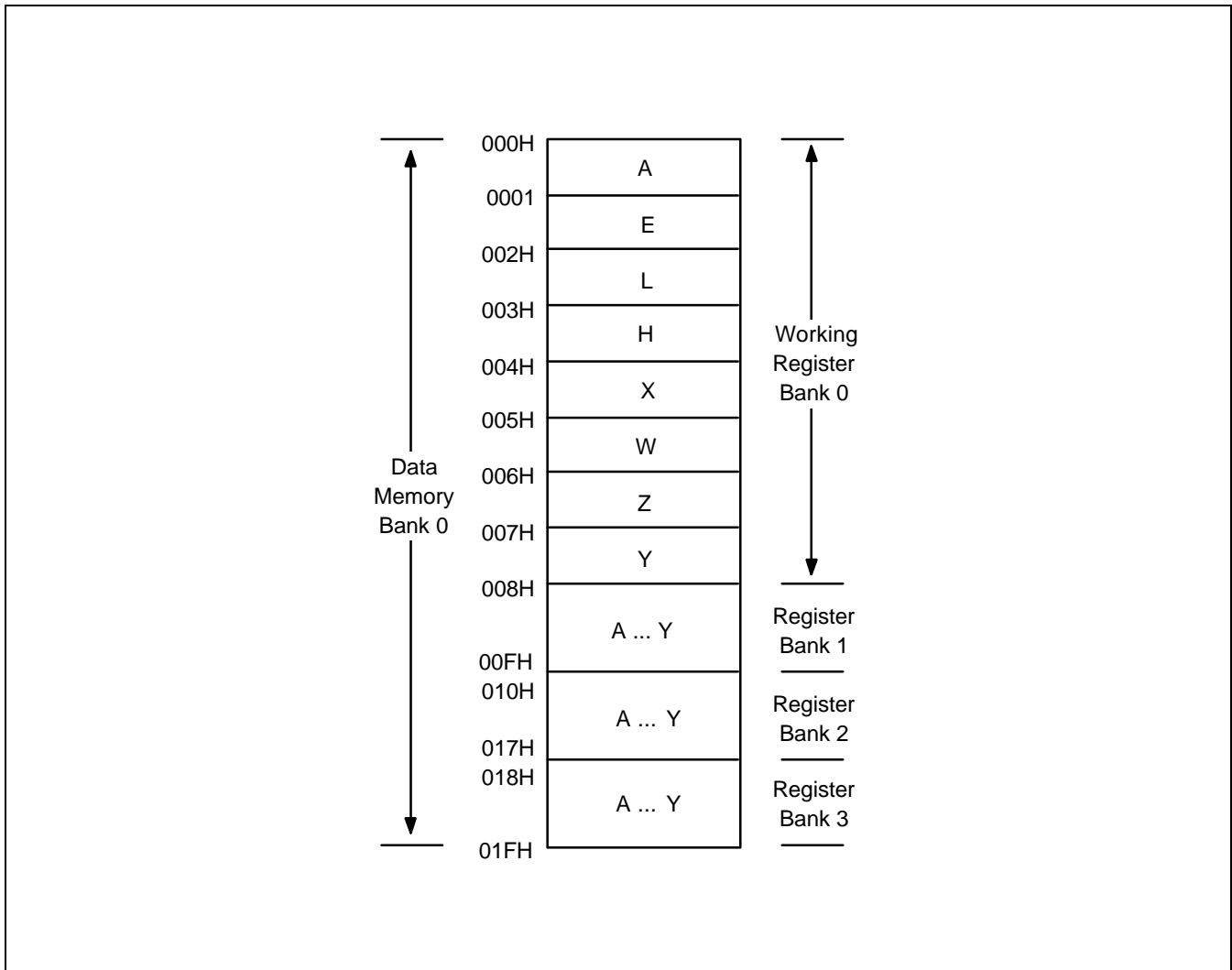


Figure 2-4. Working Register Map

Working Register Banks

For addressing purposes, the working register area is divided into four register banks — bank 0, bank 1, bank 2, and bank 3. Any one of these banks can be selected as the working register bank by the register bank selection instruction (SRBn) and by setting the status of the register bank enable flag (ERB).

Generally, working register bank 0 is used for the main program, and banks 1, 2, and 3 for interrupt service routines. Following this convention helps to prevent possible data corruption during program execution due to contention in register bank addressing.

Table 2-3. Working Register Organization and Addressing

ERB Setting	SRB Settings				Selected Register Bank
	3	2	1	0	
0	0	0	x	x	Always set to bank 0
1	0	0	0	0	Bank 0
			0	1	Bank 1
			1	0	Bank 2
			1	1	Bank 3

NOTE: 'x' means "don't care".

Paired Working Registers

Each of the register banks is subdivided into eight 4-bit registers. These registers are named Y, Z, W, X, H, L, E and A. You can manipulate them individually using 4-bit instructions, or as register pairs for 8-bit data manipulation.

The names of the 8-bit register pairs in each register bank are EA, HL, WX, YZ and WL. Registers A, L, X and Z always become the lower nibble when registers are addressed as 8-bit pairs. This makes a total of eight 4-bit registers or four 8-bit double registers in each of the four working register banks.

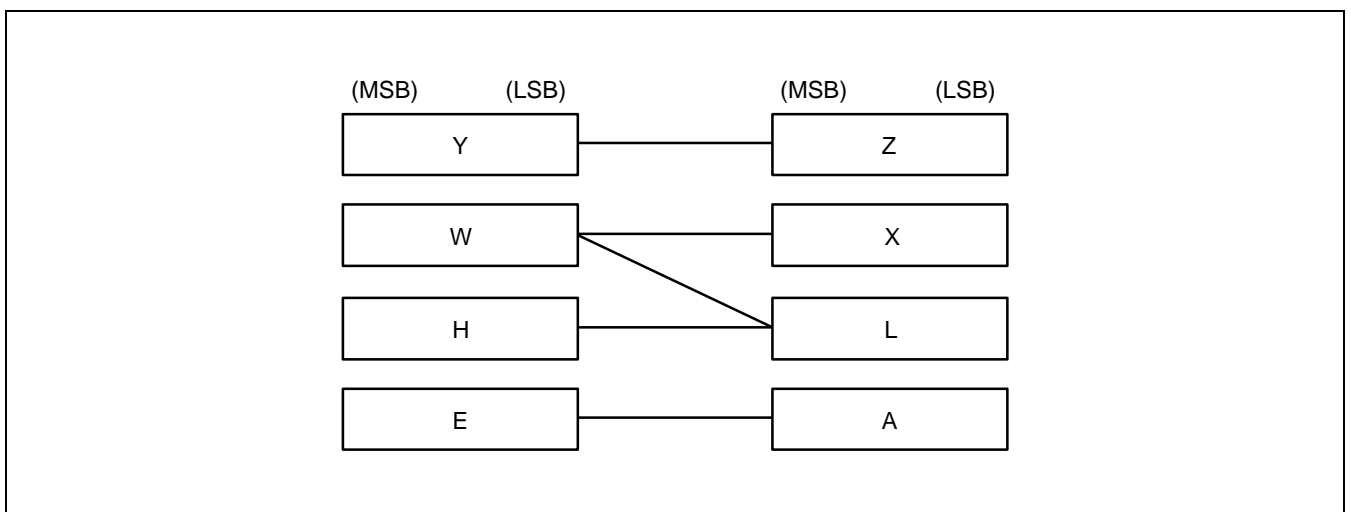


Figure 2-5. Register Pair Configuration

Special-Purpose Working Registers

You use register A as a 4-bit accumulator and double register EA as an 8-bit accumulator. You can use the carry flag as a 1-bit accumulator.

8-bit double registers WX, WL and HL are used as data pointers for indirect addressing. When the HL register serves as a data pointer, the instructions LDI, LDD, XCHI, and XCHD can make very efficient use of working registers as program loop counters by letting you transfer a value and increment or decrement the L register value using a single instruction.

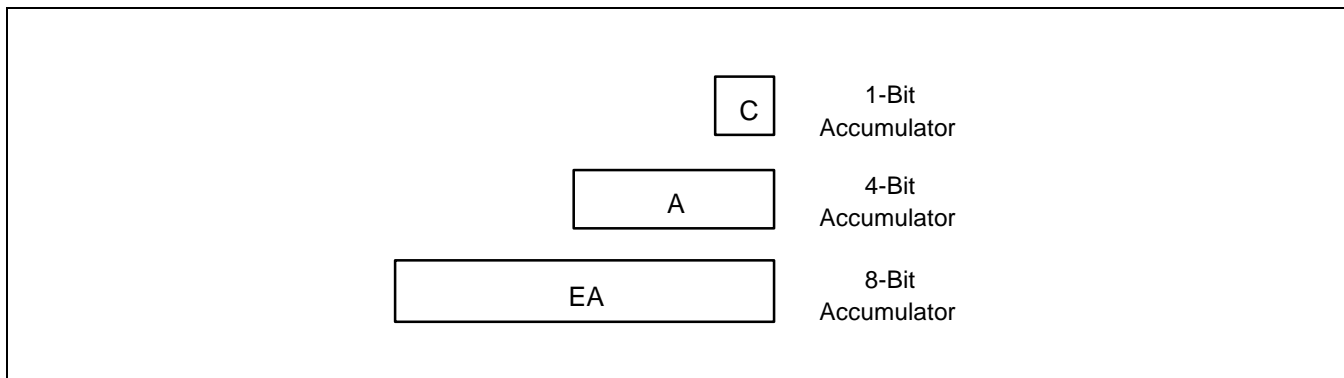


Figure 2-6. 1-Bit, 4-Bit, and 8-Bit Accumulator

Recommendation for Multiple Interrupt Processing

If more than four interrupts are being processed at one time, you can avoid possible loss of working register data by using the PUSH RR instruction to save register contents to the stack before the service routines are executed in the same register bank. When the routines have executed successfully, you can restore the register contents from the stack to working memory using the POP instruction.

PROGRAMMING TIP — Selecting Your Working Register Area

The following examples show the correct programming method for selecting working register area:

1. When ERB = "0":

```

VENT2      1,0,INT0          ; EMB ← 1, ERB ← 0, Jump to INT0 address
;
INT0       PUSH      SB      ; PUSH current SMB, SRB
          SRB        2      ; Non-essential instruction, since ERB = "0"
          PUSH      HL      ; PUSH HL register to stack
          PUSH      WX      ; PUSH WX register to stack
          PUSH      YZ      ; PUSH YZ register to stack
          PUSH      EA      ; PUSH EA register to stack
          SMB        0
          LD        EA,#00H
          LD        80H,EA
          LD        HL,#40H
          INCS      HL
          LD        WX,EA
          LD        YZ,EA
          POP       EA      ; POP EA register from stack
          POP       YZ      ; POP YZ register from stack
          POP       WX      ; POP WX register from stack
          POP       HL      ; POP HL register from stack
          POP       SB      ; POP current SMB, SRB
          IRET
;

```

The POP instructions execute alternately with the PUSH instructions. If an SMB instruction is used in an interrupt service routine, a PUSH and POP SB instruction must be used to store and restore the current SMB and SRB values, as shown in Example 2 below.

2. When ERB = "1":

```

VENT2      1,1,INT0          ; EMB ← 1, ERB ← 1, Jump to INT0 address
;
INT0       PUSH      SB      ; Store current SMB, SRB
          SRB        2      ; Select register bank 2
          SMB        0
          LD        EA,#00H
          LD        80H,EA
          LD        HL,#40H
          INCS      HL
          LD        WX,EA
          LD        YZ,EA
          POP       SB      ; Restore SMB, SRB
          IRET
;

```

STACK OPERATIONS

STACK POINTER (SP)

The stack pointer (SP) is an 8-bit register that stores the address used to access the stack, an area of data memory set aside for temporary storage of data and addresses. The SP is mapped to the RAM addresses F80H-F81H, and can be read or written by 8-bit control instructions. When addressing the SP, bit 0 must always remain cleared to logic zero.

F80H	SP3	SP2	SP1	"0"
F81H	SP7	SP6	SP5	SP4

There are two basic stack operations: writing to the top of the stack (push), and reading from the top of the stack (pop). A push decrements the SP and a pop increments it so that the SP always points to the top address of the last data to be written to the stack.

The program counter contents and program status word are stored in the stack area prior to the execution of a CALL or PUSH instruction, or during interrupt service routines. Stack operation is in a LIFO (Last-In First-Out) type. The stack area is located in general-purpose data memory bank 0.

During an interrupt or subroutine, the PC value and the PSW are saved to the stack area. When the routine has completed, the stack pointer is referenced to restore the PC and PSW, and the next instruction is executed.

The SP can address stack registers in bank 0 (addresses 000H-0FFH) regardless of the current value of the enable memory bank (EMB) flag and the select memory bank (SMB) flag.

Since the reset value of the stack pointer is not defined in firmware, we recommend that you initialize the stack pointer by program code to location 00H. This sets the first register of the stack area to 0FFH.

NOTE

A subroutine call occupies six nibbles in the stack; an interrupt requires six. When subroutine nesting or interrupt routines are used continuously, the stack area should be set in accordance with the maximum number of subroutine levels. To do this, estimate the number of nibbles that will be used for the subroutines or interrupts and set the stack area correspondingly.

Although you may use general-purpose register areas for stack operations, be careful to avoid data loss caused by simultaneous use of the same register(s).

PROGRAMMING TIP — Initializing the Stack Pointer

To initialize the stack pointer (SP):

1. When EMB = "1":

```
SMB      15      ; Select memory bank 15
LD       EA,#00H ; Bit 0 of accumulator A is always cleared to "0"
LD       SP,EA   ; Stack area initial address (0FFH) ← (SP) - 1
```

2. When EMB = "0":

```
LD       EA,#00H
LD       SP,EA   ; Memory addressing area (00H-7FH, F80H-FFFH)
```

PUSH OPERATIONS

Three kinds of push operations reference the stack pointer (SP) to write data from the source register to the stack: PUSH instructions, CALL instructions, and interrupts. In each case, the SP is *decremented* by a number determined by the type of push operation, pointing to the next stack location available.

PUSH Instructions

A PUSH instruction references the SP to write two 4-bit data nibbles from the PC to the stack. Two 4-bit stack addresses are referenced by the stack pointer: one for the upper register value and the other for the lower register. After the PUSH has executed, the SP is decremented *by two*, pointing to the next stack location available.

CALL Instructions

When a subroutine call is issued, the CALL instruction references the SP to write the PC's contents to four 4-bit stack locations. Current values for the enable memory bank (EMB) flag and the enable register bank (ERB) flag are also pushed to the stack. After the CALL has executed, the SP is decremented *by six*, pointing to the next stack location available. Since six 4-bit stack locations are used per CALL, you may nest subroutine calls up to the number of levels permitted in the stack.

Interrupt Routines

An interrupt routine references the SP to push the contents of the PC, as well as current values for the program status word (PSW) to the stack. Six 4-bit stack locations are used to store this data. After the interrupt has executed, the SP is decremented *by six*, pointing to the next stack location available. During an interrupt sequence, subroutines may be nested up to the number of levels which are permitted in the stack area.

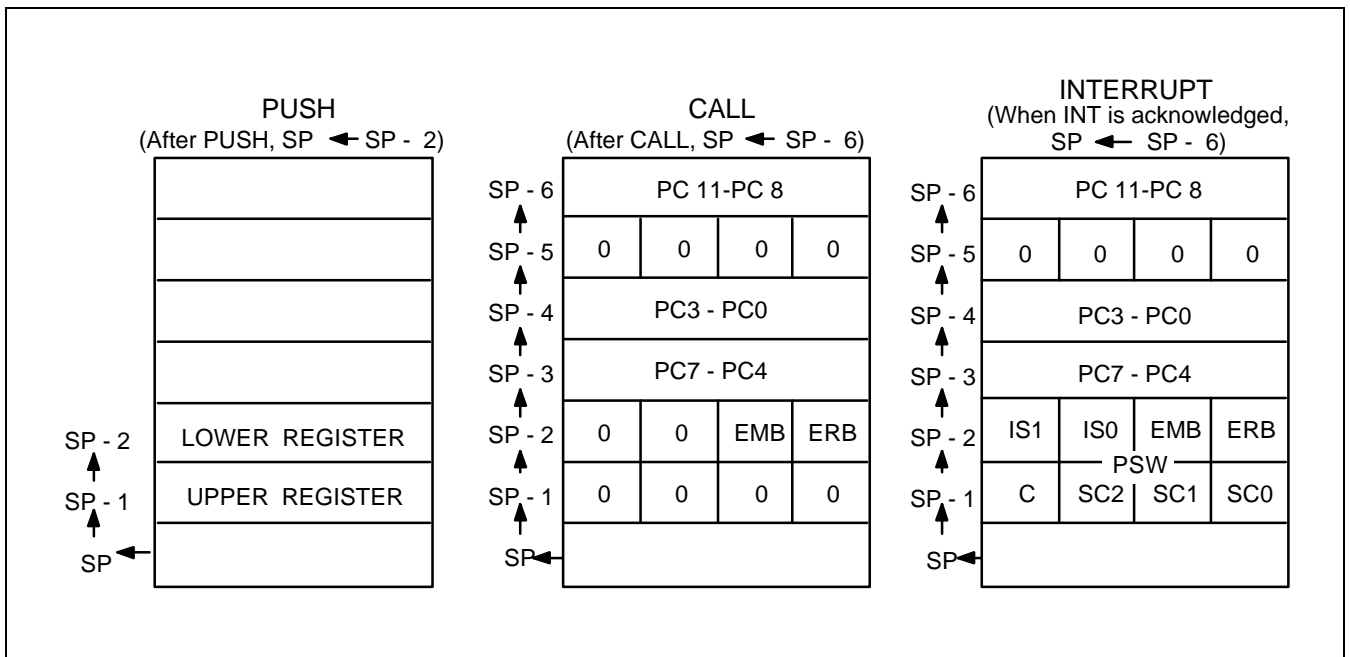


Figure 2-7. Push-Type Stack Operations

POP OPERATIONS

For each push operation there is a corresponding pop operation to write data from the stack back to the source register or registers: for the PUSH instruction it is the POP instruction; for CALL, the instruction RET or SRET; for interrupts, the instruction IRET. When a pop operation occurs, the SP is *incremented* by a number determined by the type of operation, pointing to the next stack location free.

POP Instructions

A POP instruction references the SP to write data stored in two 4-bit stack locations back to the register pairs and SB register. The value for the lower 4-bit register is popped first, followed by the value for the upper 4-bit register. After the POP has executed, the SP is incremented *by two*, pointing to the next stack location free.

RET and SRET Instructions

The end of a subroutine call is signaled by the return instruction, RET or SRET. The RET or SRET uses the SP to reference the four 4-bit stack locations used for the CALL and to write this data back to the PC, the EMB, and ERB. After the RET or SRET has executed, the SP is incremented *by six*, and pointing to the next stack location free.

IRET Instructions

The end of an interrupt sequence is signaled by the instruction IRET. IRET references the SP to locate the six 4-bit stack addresses used for the interrupt and to write this data back to the PC and the PSW. After the IRET has executed, the SP is incremented *by six*, pointing to the next stack location free.

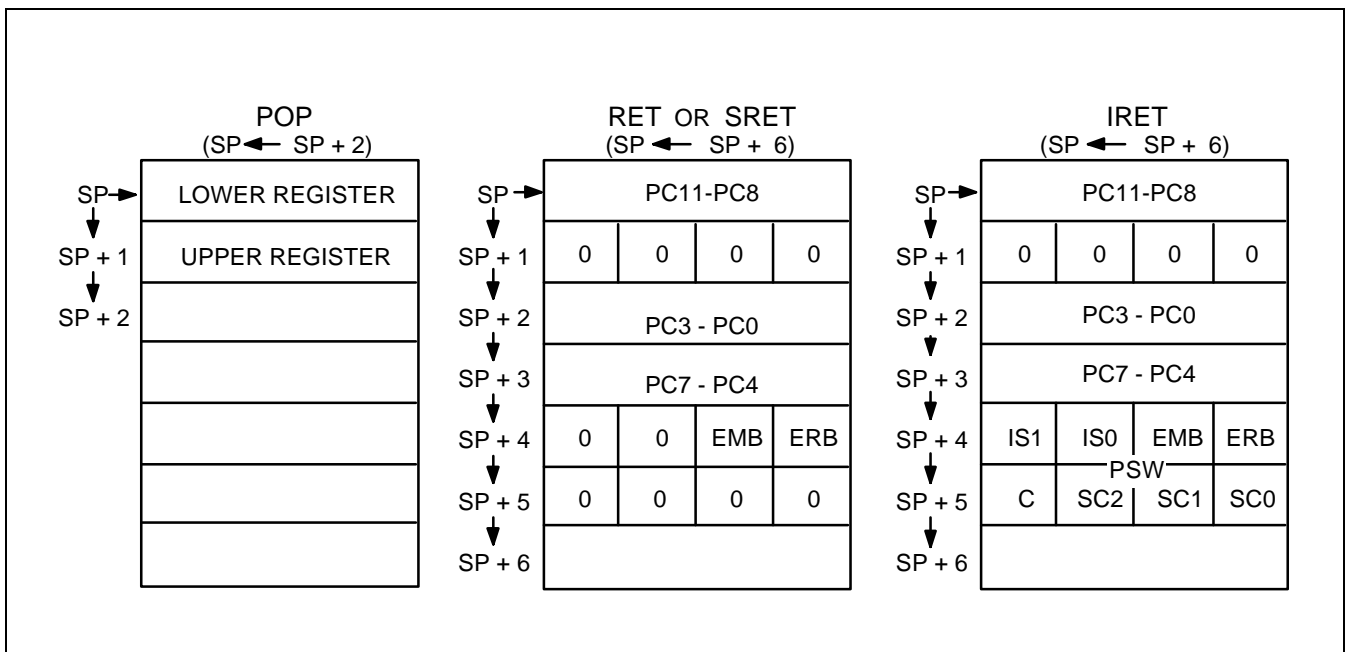


Figure 2-8. Pop-Type Stack Operations

BIT SEQUENTIAL CARRIER (BSC)

The bit sequential carrier (BSC) is a 16-bit register that is mapped to the RAM addresses FC0H–FC3H. You can manipulate the BSC register using 1-, 4-, and 8-bit RAM control instructions. A RESET clears all BSC bit values to logic zero.

Using the BSC, you can specify addresses and bit locations sequentially using 1-bit indirect addressing (memb.@L). Bit addressing is independent of the current EMB value. In this way, programs can process 16-bit data by moving the bit location sequentially and then incrementing or decrementing the value of the L register.

BSC data can also be manipulated using direct addressing. For 8-bit manipulations, specify the 4-bit register names BSC0 and BSC2 and manipulate the upper and lower 8 bits separately.

If the value of the L register is 0H at BSC0.@L, the address and bit location assignment is FC0H.0. If the L register content is FH at BSC0.@L, the address and bit location assignment is FC3H.3.

Table 2-4. BSC Register Organization

Name	Address	Bit 3	Bit 2	Bit 1	Bit 0
BSC0	FC0H	BSC0.3	BSC0.2	BSC0.1	BSC0.0
BSC1	FC1H	BSC1.3	BSC1.2	BSC1.1	BSC1.0
BSC2	FC2H	BSC2.3	BSC2.2	BSC2.1	BSC2.0
BSC3	FC3H	BSC3.3	BSC3.2	BSC3.1	BSC3.0

PROGRAMMING TIP — Using the BSC Register to Output 16-Bit Data

To use the bit sequential carrier (BSC) register to output 16-bit data (5937H) to the P3.0 pin:

```

BITS      EMB
SMB       15
LD        EA,#37H      ;
LD        BSC0,EA     ; BSC0 ← A, BSC1 ← E
LD        EA,#59H      ;
LD        BSC2,EA     ; BSC2 ← A, BSC3 ← E
SMB       0
LD        L,#0H       ;
AGN       LDB C,BSC0.@L ;
LDB       P3.0,C      ; P3.0 ← C
INCS     L
JR        AGN
RET

```

PROGRAM COUNTER (PC)

A 12-bit program counter (PC) stores addresses for instruction fetches during program execution. Whenever a reset operation or an interrupt occurs, the bits PC11 through PC0 are set to the vector address. The bits bit PC12–PC13 are reserved to support future expansion of the device's ROM size.

Usually, the PC is incremented by the number of bytes of the instruction being fetched. One exception is the 1-byte REF instruction which is used to reference instructions stored in the ROM.

PROGRAM STATUS WORD (PSW)

The program status word (PSW) is an 8-bit word, mapped to the RAM locations FB0H–FB1H, which defines system status and program execution status and permits an interrupted process to resume operation after an interrupt request has been serviced. PSW values are mapped as follows:

FB0H	IS1	IS0	EMB	ERB
FB1H	C	SC2	SC1	SC0

The PSW can be manipulated by 1-bit or 4-bit read/write and by 8-bit read instructions, depending on the specific bit or bits being addressed. The PSW can be addressed during program execution regardless of the current value of the enable memory bank (EMB) flag.

Part or all of the PSW is saved to the stack prior to the execution of a subroutine call or hardware interrupt. After the interrupt has been processed, the PSW values are popped from the stack back to the PSW address.

When a RESET is generated, the EMB and ERB values are set according to the RESET vector address, and the carry flag is left undefined (or the current value is retained). PSW bits IS0, IS1, SC0, SC1, and SC2 are all cleared to logic zero.

Table 2-5. Program Status Word Bit Descriptions

PSW Bit Identifier	Description	Bit Addressing	Read/Write
IS1, IS0	Interrupt status flags	1, 4	R/W
EMB	Enable memory bank flag	1	R/W
ERB	Enable register bank flag	1	R/W
C	Carry flag	1	R/W
SC2, SC1, SC0	Program skip flags	8	R

INTERRUPT STATUS FLAGS (IS0, IS1)

PSW bits IS0 and IS1 contain the current interrupt execution status values. They are mapped to the RAM bit locations FB0H.2 and FB0H.3, respectively. You can manipulate IS0 and IS1 flags directly using 1-bit RAM control instructions

By manipulating interrupt status flags in conjunction with the interrupt priority register (IPR), you can process multiple interrupts by anticipating the next interrupt in an execution sequence. The interrupt priority control circuit determines the IS0 and IS1 settings in order to control multiple interrupt processing. When both interrupt status flags are set to "0", all interrupts are allowed. The priority with which interrupts are processed is then determined by the IPR.

When an interrupt occurs, IS0 and IS1 are pushed to the stack as part of the PSW and are automatically incremented to the next higher priority level. Then, when the interrupt service routine ends with an IRET instruction, IS0 and IS1 values are restored to the PSW. Table 2-6 shows the effects of IS0 and IS1 flag settings.

Table 2-6. Interrupt Status Flag Bit Settings

IS1 Value	IS0 Value	Status of Currently Executing Process	Effect of IS0 and IS1 Settings on Interrupt Request Control
0	0	0	All interrupt requests are serviced
0	1	1	Only high-priority interrupt(s) as determined in the interrupt priority register (IPR) is serviced
1	0	2	No more interrupt requests are serviced
1	1	–	Not applicable; these bit settings are undefined

Since interrupt status flags can be addressed by write instructions, programs can exert direct control over interrupt processing status. Before interrupt status flags can be addressed, however, you must first execute a DI instruction to inhibit additional interrupt routines. When the bit manipulation has been completed, execute an EI instruction to re-enable interrupt processing.

PROGRAMMING TIP — Setting ISx Flags for Interrupt Processing

The following instruction sequence shows how to use the IS0 and IS1 flags to control interrupt processing:

```
INTB      DI                ; Disable interrupt
          BITR      IS1     ; IS1 ← 0
          BITS      IS0     ; Allow interrupts according to IPR priority level
          EI                ; Enable interrupt
```

EMB FLAG (EMB)

The enable memory bank flag EMB is mapped to the registers FB0H–FB1H in bank 15 of the RAM. The EMB flag occupies the bit location 1 in the register FB0H.

The EMB flag is used to allocate specific address locations in the RAM by modifying the upper 4 bits of 12-bit data memory addresses. In this way, it controls the addressing mode for data memory banks 0, bank 1 or 15.

When the EMB flag is "0", the data memory address space is restricted to bank 15 and the addresses 000H–07FH of memory bank 0, regardless of the SMB register contents. When the EMB flag is set to "1", you can access general-purpose areas of bank 0, bank 1, and bank 15 by using the appropriate SMB value.

 PROGRAMMING TIP — Using the EMB Flag to Select Memory Banks

EMB flag settings for memory bank selection:

1. When EMB = "0":

SMB	0	; Non-essential instruction, since EMB = "0"
LD	90H,A	; (F90H) ← A, bank 15 is selected
LD	34H,A	; (034H) ← A, bank 0 is selected
SMB	15	; Non-essential instruction, since EMB = "0"
LD	20H,A	; (020H) ← A, bank 0 is selected
LD	90H,A	; (F90H) ← A, bank 15 is selected

;

2. When EMB = "1":

SMB	0	; Select memory bank 0
LD	90H,A	; (090H) ← A, bank 0 is selected
LD	34H,A	; (034H) ← A, bank 0 is selected
SMB	15	; Select memory bank 15
LD	20H,A	; Program error, but assembler does not detect it
LD	90H,A	; (F90H) ← A, bank 15 is selected

;

ERB FLAG (ERB)

The 1-bit register bank enable flag (ERB) determines the range of addressable working register area. When the ERB flag is "1", you should select the working register area from the register banks 0 to 3 according to the register bank selection register (SRB). When the ERB flag is "0", you should select register bank 0 as the working register area, regardless of the current value of the register bank selection register (SRB).

When an internal RESET is generated, bit 6 of the program memory address 0000H is written to the ERB flag. This automatically initializes the flag. When a vectored interrupt is generated, bit 6 of the respective vector address table in the program memory is written to the ERB flag, setting the correct flag status before the interrupt service routine is executed.

During the interrupt routine, the ERB value is automatically pushed to the stack area along with the other PSW bits. Afterwards, it is popped back to the FB0H.0 bit location. The initial ERB flag settings for each vectored interrupt are defined using VENTn instructions.

PROGRAMMING TIP — Using the ERB Flag to Select Register Banks

ERB flag settings for register bank selection:

1. When ERB = "0":

SRB	1	; Register bank 0 is selected (since ERB = "0", the
		; SRB is configured to bank 0)
LD	EA,#34H	; Bank 0 EA ← #34H
LD	HL,EA	; Bank 0 HL ← EA
SRB	2	; Register bank 0 is selected
LD	YZ,EA	; Bank 0 YZ ← EA
SRB	3	; Register bank 0 is selected
LD	WX,EA	; Bank 0 WX ← EA
		;

2. When ERB = "1":

SRB	1	; Register bank 1 is selected
LD	EA,#34H	; Bank 1 EA ← #34H
LD	HL,EA	; Bank 1 HL ← Bank 1 EA
SRB	2	; Register bank 2 is selected
LD	YZ,EA	; Bank 2 YZ ← BANK 2 EA
SRB	3	; Register bank 3 is selected
LD	WX,EA	; Bank 3 WX ← Bank 3 EA
		;

SKIP CONDITION FLAGS (SC2, SC1, SC0)

The skip condition flags SC2, SC1, and SC0 indicate the current program skip conditions and they are set and reset automatically during program execution. These flags are mapped to the RAM bit locations FB1H.0, FB1H.1, and FB1H.2 of the PSW.

Skip condition flags can only be addressed by 8-bit read instructions. Direct manipulation of the SC2, SC1, and SC0 bits is not allowed.

CARRY FLAG (C)

The carry flag is mapped to the bit location FB1H.3 in the PSW. It is used to save the result of an overflow or borrow when executing arithmetic instructions involving a carry (ADC, SBC). The carry flag can also be used as a 1-bit accumulator performing Boolean operations involving bit-addressed data memory.

If an overflow or borrow condition occurs when executing arithmetic instructions with carry (ADC, SBC), the carry flag is set to "1". Otherwise, its value is "0". When a RESET occurs, the current value of the carry flag is retained during power-down mode, but when normal operating mode resumes, its value is undefined.

The carry flag can be directly manipulated by predefined set of 1-bit read/write instructions, independent of other bits in the PSW. Only the ADC and SBC instructions, and the instructions listed in Table 2-7 affect the carry flag.

Table 2-7. Valid Carry Flag Manipulation Instructions

Operation Type	Instructions	Carry Flag Manipulation
Direct manipulation	SCF	Set carry flag to "1"
	RCF	Clear carry flag to "0" (reset carry flag)
	CCF	Invert carry flag value (complement carry flag)
	BTST C	Test carry and skip if C = "1"
Bit transfer	LDB (operand) ⁽¹⁾ ,C	Load carry flag value to the specified bit
	LDB C,(operand) ⁽¹⁾	Load contents of the specified bit to carry flag
Data transfer	RRC A	Rotate right with carry flag
Boolean manipulation	BAND C,(operand) ⁽¹⁾	AND the specified bit with contents of carry flag and save the result to the carry flag
	BOR C,(operand) ⁽¹⁾	OR the specified bit with contents of carry flag and save the result to the carry flag
	BXOR C,(operand) ⁽¹⁾	XOR the specified bit with contents of carry flag and save the result to the carry flag
Interrupt routine	INTn ⁽²⁾	Save carry flag to stack with other PSW bits
Return from interrupt	IRET	Restore carry flag from stack with other PSW bits

NOTES:

1. The operand has three bit addressing formats: mema.a, memb.@L, and @H + DA.b.
2. INTn refers to the specific interrupt being executed, it is not an instruction.

PROGRAMMING TIP — Using the Carry Flag as a 1-Bit Accumulator

1. Set the carry flag to logic one:

```

SCF                ; C ← 1
LD      EA,#0C3H   ; EA ← #0C3H
LD      HL,#0AAH   ; HL ← #0AAH
ADC     EA,HL      ; EA ← #0C3H + #0AAH + #1H, C ← 1

```

2. Logical-AND bit 3 of address 3FH with P3.3 and output the result to P5.0:

```

LD      H,#3H      ; Set the upper four bits of the address to the H register value
LDB    C,@H+0FH.3 ; C ← bit 3 of 3FH
BAND   C,P3.3      ; C ← C AND P3.3
LDB    P5.0,C      ; Output the result from carry flag to P5.0

```

3

ADDRESSING MODES

OVERVIEW

The enable memory bank flag, EMB, controls the two addressing modes for data memory. When you enable the EMB flag, you can address the entire RAM area. When you clear the EMB flag to logic zero, the addressable RAM is restricted to specific areas.

The EMB flag works in connection with the select memory bank instruction, SMB n. Recall that the SMB n instruction is used to select RAM bank 0, bank 1 or 15. The SMB setting is always contained in the upper four bits of a 12-bit RAM address. For this reason, both addressing modes (EMB = "0" and EMB = "1") apply specifically to the memory bank indicated by the SMB instruction, and any restrictions to the addressable area within bank 0, 1 or 15. Direct and indirect 1-, 4-, and 8-bit addressing methods can be used.

In addition, there are several RAM locations that can always be addressed using specific addressing methods, regardless of the current EMB flag setting.

Here are a few things to remember about addressing data memory areas:

- When you address peripheral hardware locations in bank 15, you can use the mnemonic for the memory-mapped hardware component as the operand in place of the actual address location.
- Always use an even-numbered RAM address as the operand in 8-bit direct and indirect addressing.
- With direct addressing, use the RAM address as the instruction operand; with indirect addressing, the instruction specifies a register which contains the operand's address.

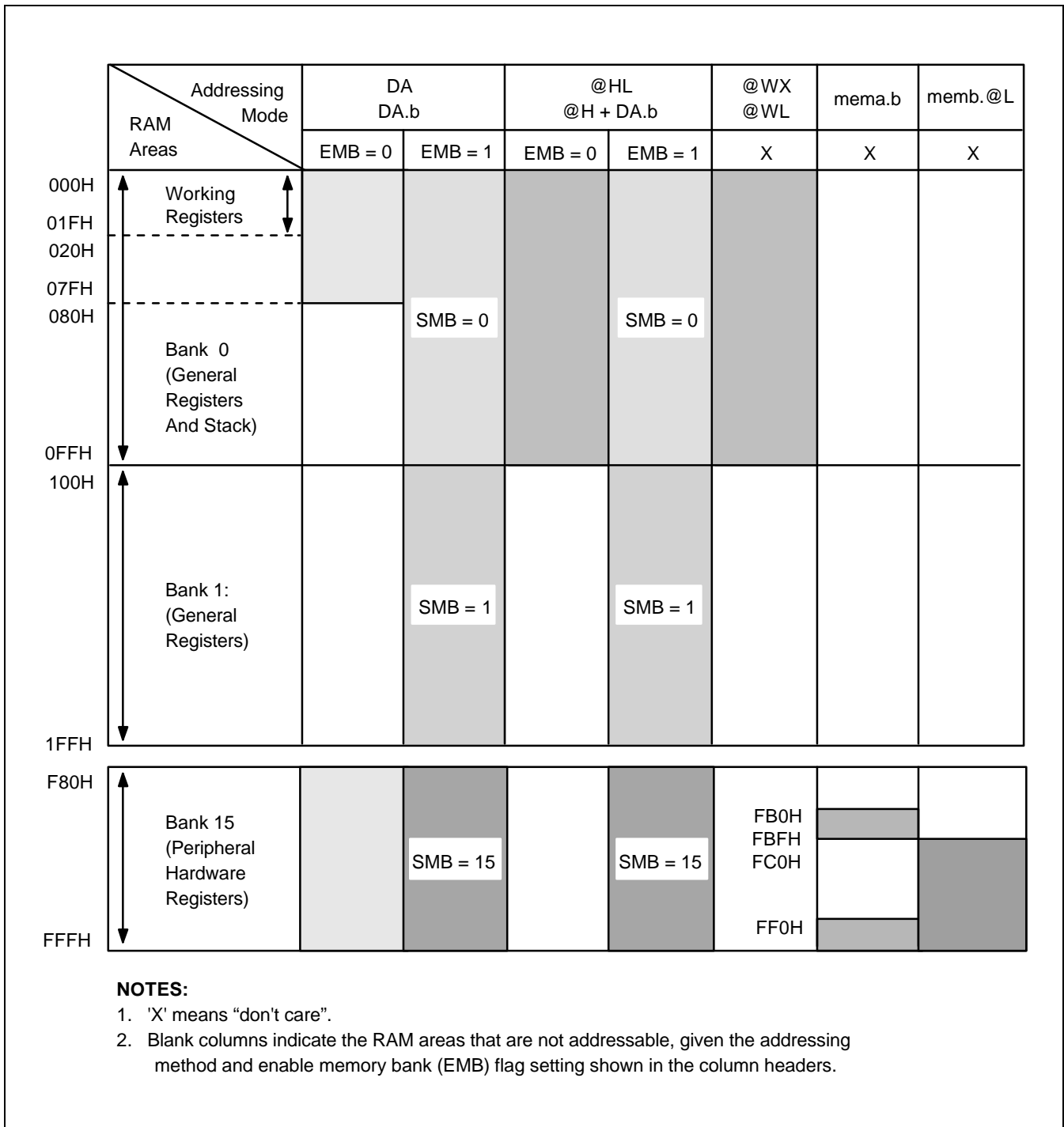


Figure 3-1. RAM Address Structure

EMB AND ERB INITIALIZATION VALUES

The EMB and ERB flag bits are set automatically by the values of the RESET vector address and the interrupt vector address.

When a RESET is generated internally, bit 7 of the program memory address 0000H is written to the EMB flag, initializing it automatically. When a vectored interrupt is generated, bit 7 of the respective vector address table is written to the EMB. This automatically sets the EMB flag status for the interrupt service routine. When the interrupt is serviced, the EMB value is automatically saved to the stack and then restored when the interrupt routine is completed.

At the beginning of a program, the initial EMB flag value for each vectored interrupt must be set by using VENT instruction. The EMB can be set or reset by bit manipulation instructions (BITS, BITR) regardless of the current SMB setting.

PROGRAMMING TIP — Initializing the EMB and ERB Flags

The following assembly instructions show how to initialize the EMB and ERB flag settings:

```

    ORG      0000H           ; ROM address assignment
    VENT0    1,0,RESET      ; EMB ← 1, ERB ← 0, branch RESET
    VENT1    0,1,INTB      ; EMB ← 0, ERB ← 1, branch INTB
    VENT2    0,1,INT0      ; EMB ← 0, ERB ← 1, branch INT0
    VENT3    0,1,INT1      ; EMB ← 0, ERB ← 1, branch INT1
    NOP
    NOP
    VENT5    0,1,INTT0     ; EMB ← 0, ERB ← 1, branch INTT0
    .
    .
    .
RESET      BITR      EMB

```


ENABLE MEMORY BANK SETTINGS**EMB = "1"**

When you set the enable memory bank flag, EMB, to logic one, you can address the data memory bank specified by the select memory bank (SMB) value (0,1 or 15) using 1-, 4-, or 8-bit instructions. You can use both direct and indirect addressing modes. The addressable RAM areas when the EMB flag is set to logic one are as follows:

If SMB = 0, 000H–0FFH

If SMB = 1 100H–1FFH

If SMB = 15, F80H–FFFH

EMB = "0"

When the enable memory bank flag, EMB, is set to logic zero, the addressable area is defined independently of the SMB value, and restricted to specific locations depending on whether direct or indirect address mode is used.

If EMB = "0", the addressable area is restricted to the locations 000H–07FH in bank 0 and to the locations F80H–FFFH in bank 15 in direct addressing. In indirect addressing, only the locations 000H–0FFH in bank 0 are addressable, regardless of the SMB value.

To address the peripheral hardware register (bank 15) using indirect addressing, the EMB flag must first be set to "1" and the SMB value to "15". When a RESET occurs, the EMB flag is set to the value contained in bit 7 of the ROM address 0000H.

EMB-Independent Addressing

You can address several areas of the data memory at any time, regardless of the status of the EMB flag. These exceptions are described in Table 3-1.

Table 3-1. RAM Addressing Not Affected by the EMB Value

Address	Addressing Method	Affected Hardware	Program Examples
000H–0FFH	4-bit indirect addressing using WX and WL register pairs; 8-bit indirect addressing using SP	Not applicable	LD A,@WX PUSH POP
FB0H–FBFH FF0H–FFFH	1-bit direct addressing	PSW, IEx, IRQx, I/O	BITS EMB BITR IE1
FC0H–FFFH	1-bit indirect addressing using the L register	BSC, I/O	BTST FC3H.@L BAND C,P3.@L

SELECT BANK REGISTER (SB)

The select bank register (SB) is used to assign the memory bank and register bank. The 8-bit SB register consists of the 4-bit select register bank register (SRB) and the 4-bit select memory bank register (SMB), as shown in Figure 3-2.

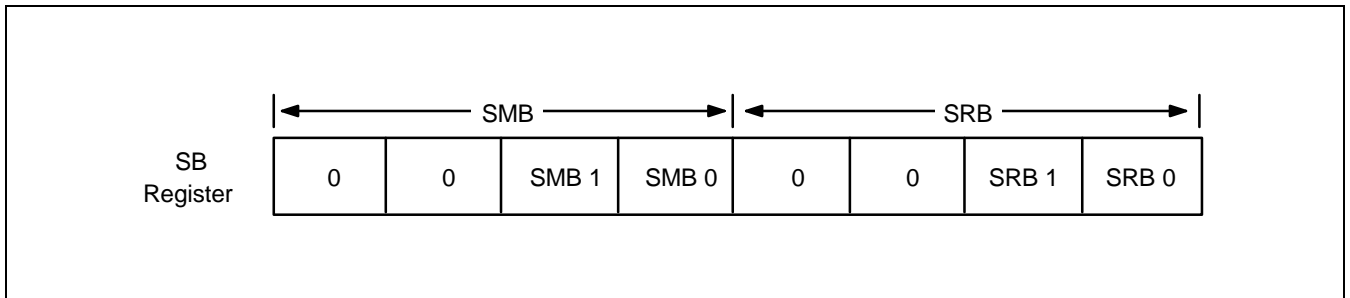


Figure 3-2. 4-Bit SMB and SRB Values in the SB Register

During interrupts and subroutine calls, the contents of the SB register can be saved to the stack in 8-bit units by the PUSH SB instruction. You can later restore the value to the SB using the POP SB instruction.

Select Register Bank (SRB) Instruction

The select register bank (SRB) value specifies which register bank is to be used as a working register bank. The SRB value is set by the 'SRB n' instruction, where $n = 0, 1, 2, 3$. One of the four register banks is selected by the combination of ERB flag status and the SRB value that you set using the 'SRB n' instruction. The current SRB value is retained until another register is requested by program software.

PUSH SB and POP SB instructions are used to save and restore the contents of SRB during interrupts and subroutine calls. A RESET clears the 4-bit SRB value to logic zero.

Select Memory Bank (SMB) Instruction

To select one of the two data memory banks available, you must execute an SMB n instruction specifying the number of the memory bank you want (0, 1 or 15). For example, the instruction 'SMB 1' selects bank 1 and 'SMB 15' selects bank 15. You must also remember to enable the memory bank you select by setting the enable memory bank flag (EMB) appropriately.

The upper four bits of the 12-bit data memory address are stored in the SMB register. If the SMB value is not specified by software (or if a RESET does not occur) the current value is retained. A RESET clears the 4-bit SMB value to logic zero.

PUSH SB and POP SB instructions save and restore the contents of the SMB register to and from the stack area during interrupts and subroutine calls.

DIRECT AND INDIRECT ADDRESSING

You can directly address 1-bit, 4-bit, and 8-bit data stored in data memory locations using a specific register or bit address as the instruction operand.

In indirect addressing, the instruction specifies a specific register pair containing the address of the operand. The S3C7 instruction set supports 1-bit, 4-bit, and 8-bit indirect addressing. In 8-bit indirect addressing, an even-numbered RAM address must always be used as the instruction operand, and the address register can be HL, WX, or WL of the selected register bank.

1-BIT ADDRESSING

Table 3-2. 1-Bit Direct and Indirect RAM Addressing

Instruction Notation	Addressing Mode Description	EMB Flag Setting	Addressable Area	Memory Bank	Hardware I/O Mapping
DA.b	Direct: bit is indicated by the RAM address (DA), memory bank selection, and specified bit number (b).	0	000H–07FH	Bank 0	–
			F80H–FFFH	Bank15	All 1-bit addressable peripherals (SMB = 15)
		1	000H–FFFH	SMB = 0, 1, 15	
mema.b	Direct: bit is indicated by addressable area (mema) and bit number (b).	x	FB0H–FBFH FF0H–FFFH	Bank 15	IS0, IS1, EMB, ERB, IEx, IRQx, Pn.m
memb.@L	Indirect: the lower two bits of register L are indicated by the upper 10 bits of RAM area (memb) and the upper two bits of register L.	x	FC0H–FFFH	Bank 15	BSCn.x Pn.m
@H + DA.b	Indirect: bit is indicated by the lower four bits of the address (DA), memory bank selection, and the H register identifier.	0	000H–0FFH	Bank 0	–
		1	000H–FFFH	SMB = 0, 1, 15	All 1-bit addressable peripherals (SMB = 15)

NOTE: 'x' means "don't care".

PROGRAMMING TIP — 1-Bit Addressing Modes

1-Bit Direct Addressing

1. If EMB = "0":

```
AFLAG    EQU    34H.3
BFLAG    EQU    85H.3
CFLAG    EQU    0BAH.0
          SMB    0                ; Non-essential instruction, since EMB = "0"
          BITS   AFLAG           ; 34H.3 ← 1
          BITS   BFLAG           ; F85H.3 (BMOD.3) ← 1
          BTST   CFLAG           ; If FBAH.0 (IRQW) = 1, skip
          BITS   BFLAG           ; Else if FBAH.0 (IRQW) = 0, F85H.3 (BMOD.3) ← 1
          BITS   P3.0            ; FF3H.0 (P3.0) ← 1
```

;

2. If EMB = "1":

```
AFLAG    EQU    34H.3
BFLAG    EQU    85H.3
CFLAG    EQU    0BAH.0
          SMB    0                ; Select memory bank 0
          BITS   AFLAG           ; 34H.3 ← 1
          BITS   BFLAG           ; 85H.3 ← 1
          BTST   CFLAG           ; If 0BAH.0 = 1, skip
          BITS   BFLAG           ; Else if 0BAH.0 = 0, 085H.3 ← 1
          BITS   P3.0            ; FF3H.0 (P3.0) ← 1
```

;

 PROGRAMMING TIP — 1-Bit Addressing Modes (Continued)

1-Bit Indirect Addressing

1. If EMB = "0":

```

AFLAG    EQU    34H.3
BFLAG    EQU    85H.3
CFLAG    EQU    0BAH.0
          SMB    0           ; Non-essential instruction, since EMB = "0"
          LD     H,#0BH     ; H ← #0BH
          BTSTZ @H+CFLAG   ; If 0BAH.0 = 1, 0BAH.0 ← 0 and skip
          BITS   CFLAG     ; Else if 0BAH.0 = 0, 0BAH.0 (IRQW) ← 1
;

```

2. If EMB = "1":

```

AFLAG    EQU    34H.3
BFLAG    EQU    85H.3
CFLAG    EQU    0BAH.0
          SMB    0           ; Select memory bank 0
          LD     H,#0BH     ; H ← #0BH
          BTSTZ @H+CFLAG   ; If 0BAH.0 = 1, 0BAH.0 ← 0 and skip
          BITS   CFLAG     ; Else if 0BAH.0 = 0, 0BAH.0 ← 1
;


```

4-BIT ADDRESSING

Table 3-3. 4-Bit Direct and Indirect RAM Addressing

Instruction Notation	Addressing Mode Description	EMB Flag Setting	Addressable Area	Memory Bank	Hardware I/O Mapping
DA	Direct: 4-bit address indicated by the RAM address (DA) and the memory bank selection	0	000H–07FH	Bank 0	–
			F80H–FFFH	Bank15	All 4-bit addressable peripherals
		1	000H–FFFH	SMB = 0, 1, 15	(SMB = 15)
@HL	Direct: 4-bit address indicated by the memory bank selection and register HL	0	000H–0FFH	Bank 0	–
		1	000H–FFFH	SMB = 0, 1, 15	All 4-bit addressable peripherals (SMB = 15)
@WX	Indirect: 4-bit address indicated by register WX	x	000H–0FFH	Bank 0	–
@WL	Indirect: 4-bit address indicated by register WL	x	000H–0FFH	Bank 0	–

NOTE: 'x' means "don't care".

 **PROGRAMMING TIP — 4-Bit Addressing Modes**
4-Bit Direct Addressing

1. If EMB = "0":

```

ADATA    EQU    46H
BDATA    EQU    8EH
          SMB    15          ; Non-essential instruction, since EMB = "0"
          LD     A,P3        ; A ← (P3)
          SMB    0          ; Non-essential instruction, since EMB = "0"
          LD     ADATA,A     ; (046H) ← A
          LD     BDATA,A     ; (F8EH) ← A
;

```

2. If EMB = "1":

```

ADATA    EQU    46H
BDATA    EQU    8EH
          SMB    15          ; Select memory bank 15
          LD     A,P3        ; A ← (P3)
          SMB    0          ; Select memory bank 0
          LD     ADATA,A     ; (046H) ← A
          LD     BDATA,A     ; (08EH) ← A
;

```

 **PROGRAMMING TIP — 4-Bit Addressing Modes (Continued)**
4-Bit Indirect Addressing

1. If EMB = "0", compare bank 0, locations 040H–046H with 060H–066H:

```

ADATA    EQU    46H
BDATA    EQU    66H
          SMB    15                ; Non-essential instruction, since EMB = "0"
          LD     HL,#BDATA
          LD     WX,#ADATA
COMP     LD     A,@WL                ; A ← bank 0 (040H–046H)
          CPSE  A,@HL                ; If bank 0 (060H–066H) = A, skip
          SRET
          DECS  L
          JR     COMP
          RET
;

```

2. If EMB = "1", exchange bank 0, 040H–046H with 060H–066H:

```


ADATA    EQU    46H
BDATA    EQU    66H
          SMB    0                ; Select memory bank 0
          LD     HL,#BDATA
          LD     WX,#ADATA
TRANS   LD     A,@WL                ; A ← bank 0 (040H–046H)
          XCHD  A,@HL                ; Bank 0 (060H–066H) ← A
          JR     TRANS
;

```

8-BIT ADDRESSING

Table 3-4. 8-Bit Direct and Indirect RAM Addressing

Instruction Notation	Addressing Mode Description	EMB Flag Setting	Addressable Area	Memory Bank	Hardware I/O Mapping
DA	Direct: 8-bit address indicated by the RAM address (<i>DA = even number</i>) and the memory bank selection	0	000H–07FH	Bank 0	–
			F80H–FFFH	Bank15	All 8-bit addressable peripherals (SMB = 15)
		1	000H–FFFH	SMB = 0, 1, 15	
@HL	Indirect: the 8-bit address indicated by the memory bank selection and register HL; (the 4-bit L register value must be an even number)	0	000H–0FFH	Bank 0	–
		1	000H–FFFH	SMB = 0, 1, 15	All 8-bit addressable peripherals (SMB = 15)

 PROGRAMMING TIP — 8-Bit Addressing Modes

8-Bit Direct Addressing:

1. If EMB = "0":

```

ADATA    EQU    46H
BDATA    EQU    8EH
          SMB    15                ; Non-essential instruction, since EMB = "0"
          LD     EA,P4             ; E ← (P5), A ← (P4)
          LD     ADATA,EA         ; (046H) ← A, (047H) ← E
          LD     BDATA,EA         ; (F8EH) ← A, (F8FH) ← E
;

```

2. If EMB = "1":

```

ADATA    EQU    46H
BDATA    EQU    8EH
          SMB    15                ; Select memory bank 15
          LD     EA,P4             ; E ← (P5), A ← (P4)
          SMB    0                 ; Select memory bank 0
          LD     ADATA,EA         ; (046H) ← A, (047H) ← E
          LD     BDATA,EA         ; (08EH) ← A, (08FH) ← E

```


 **PROGRAMMING TIP — 8-Bit Addressing Modes (Continued)****8-Bit Indirect Addressing**

1. If EMB = "0":

```
ADATA    EQU    8EH
          LD     HL,#ADATA
          LD     EA,@HL           ; A ← (08EH), E ← (08FH)
;
```

2. If EMB = "1":

```
ADATA    EQU    46H
          SMB    0
          LD     HL,#ADATA
          LD     EA,@HL           ; A ← (046H), E ← (047H)
;
```

4

MEMORY MAP

OVERVIEW

To support program control of peripheral hardware, I/O addresses for peripherals are memory-mapped to bank 15 of the RAM. Memory mapping allows you to use a mnemonic as the operand of an instruction in place of a specific memory location.

Access to bank 15 is controlled by the select memory bank (SMB) instruction and by the enable memory bank flag (EMB) setting. If the EMB flag is "0", bank 15 can be addressed using direct addressing, regardless of the current SMB value. You can use 1-bit direct and indirect addressing, however, for specific locations in bank 15, regardless of the current EMB value.

I/O MAP FOR HARDWARE REGISTERS

Table 4-1 contains detailed information about I/O mapping for peripheral hardware in bank 15 (register locations F80H–FFFH). Use the I/O map as a quick-reference source when writing application programs. The I/O map gives you the following information:

- Register address
- Register name (mnemonic for program addressing)
- Bit values (both addressable and non-manipulable)
- Read-only, write-only, or read and write addressability
- 1-, 4-, or 8-bit data manipulation characteristics

Table 4-1. I/O Map for Memory Bank 15

Memory Bank 15							Addressing Mode		
Address	Register	Bit 3	Bit 2	Bit 1	Bit 0	R/W	1-Bit	4-Bit	8-Bit
F80H	SP	.3	.2	.1	"0"	R/W	No	No	Yes
F81H		.7	.6	.5	.4				
Locations F82H–F84H are reserved.									
F85H	BMOD	.3	.2	.1	.0	W	.3	Yes	No
F86H	BCNT					R	No	No	Yes
F87H									
F88H	BUZMOD	.3	"0"	.1	.0	W	No	Yes	No
Location F89H is reserved.									
F90H	TMOD0	.3	.2	"0"	"0"	W	.3	No	Yes
F91H		"0"	.6	.5	.4				
F92H		"0"	TOE0	"0"	"0"	R/W	Yes	Yes	No
Location F93H is reserved.									
F94H	TCNT0					R	No	No	Yes
F95H									
F96H	TREF0					W	No	No	Yes
F97H									
F98H	WDMOD	.3	.2	.1	.0	W	No	No	Yes
F99H		.7	.6	.5	.4				
F9AH	WDFLAG	WDTCF	"0"	"0"	"0"	W	Yes	No	No
Locations F9BH–FAFH are reserved.									
FB0H	PSW	IS1	IS0	EMB	ERB	R/W	Yes	Yes	Yes
FB1H		C (note)	SC2	SC1	SC0	R	No	No	
FB2H	IPR	IME	.2	.1	.0	W	IME	Yes	No
FB3H	PCON	.3	.2	.1	.0	W	No	Yes	No
FB4H	IMOD0	"0"	"0"	.1	.0	W	No	Yes	No
FB5H	IMOD1	"0"	"0"	"0"	.0	W	No	Yes	No
FB6H	IMODK	"0"	"0"	.1	.0	W	No	Yes	No
Location FB7H is reserved.									
FB8H		"0"	"0"	IEB	IRQB	R/W	Yes	Yes	No
Locations FB9H–FBBH are reserved.									
FBCH		"0"	"0"	IET0	IRQT0	R/W	Yes	Yes	No
Location FBDH is reserved.									

Table 4-1. I/O Map for Memory Bank 15 (Continued)

Memory Bank 15							Addressing Mode		
Address	Register	Bit 3	Bit 2	Bit 1	Bit 0	R/W	1-Bit	4-Bit	8-Bit
FBEH		IE1	IRQ1	IE0	IRQ0	R/W	Yes	Yes	No
FBFH		"0"	"0"	IEK	IRQK				
FC0H	BSC0					R/W	Yes	Yes	Yes
FC1H	BSC1								
FC2H	BSC2								
FC3H	BSC3								
Locations FC4H–FCFH are reserved.									
FD0H	CLMOD	.3	"0"	.1	.0	W	No	Yes	No
Locations FD1H–FD4H are reserved.									
FD5H	DAMOD	"0"	"0"	"0"	.0	W	Yes	Yes	No
FD6H	DADATA	.3	.2	.1	.0	W	No	No	Yes
FD7H		.7	.6	.5	.4				
Locations FD8H–FD9H are reserved.									
FDAH	PNE	PNE4.3	PNE4.2	PNE4.1	PNE4.0	W	No	No	Yes
FDBH		PNE5.3	PNE5.2	PNE5.1	PNE5.0				
FDCH	PUMOD	"0"	PUR2	PUR1	PUR0	W	No	No	Yes
FDDH		"0"	"0"	PUR5	PUR4				
Locations FDEH–FE7H are reserved.									
FE8H	PMG1	PM0.3	PM0.2	PM0.1	PM0.0	W	No	No	Yes
FE9H		PM1.3	PM1.2	PM1.1	PM1.0				
FEAH	PMG2	"0"	"0"	"0"	PM2.0	W	No	No	Yes
FEBH		"0"	"0"	"0"	"0"				
FECH	PMG3	PM4.3	PM4.2	PM4.1	PM4.0	W	No	No	Yes
FEDH		PM5.3	PM5.2	PM5.1	PM5.0				
Locations FEEH–FEFH are reserved.									
FF0H	Port 0	.3	.2	.1	.0	R/W	Yes	Yes	No
FF1H	Port 1	.3	.2	.1	.0	R/W			
FF2H	Port 2	–	–	–	.0	R/W			
Location FF3H is reserved.									
FF4H	Port 4	.3	.2	.1	.0	R/W	Yes	Yes	Yes
FF5H	Port 5	.3/.7	.2/.6	.1/.5	.0/.4	R/W			
Locations FF6H–FFFH are reserved.									

NOTE: The carry flag can be read or written by specific bit manipulation instructions only.

REGISTER DESCRIPTIONS

In this section, register descriptions are presented in a consistent format to familiarize you with the memory-mapped I/O locations in bank 15 of the RAM. Figure 4-1 describes the features of the register description format. Register descriptions are arranged in alphabetical order.

Counter registers, buffer registers, and reference registers, as well as the stack pointer and port I/O latches, are not included in the description.

This section can be used as a quick-reference source when writing application programs.

More detailed information about each of these registers is included in Part II of this manual, "Hardware Descriptions," in the context of the corresponding peripheral hardware module descriptions.

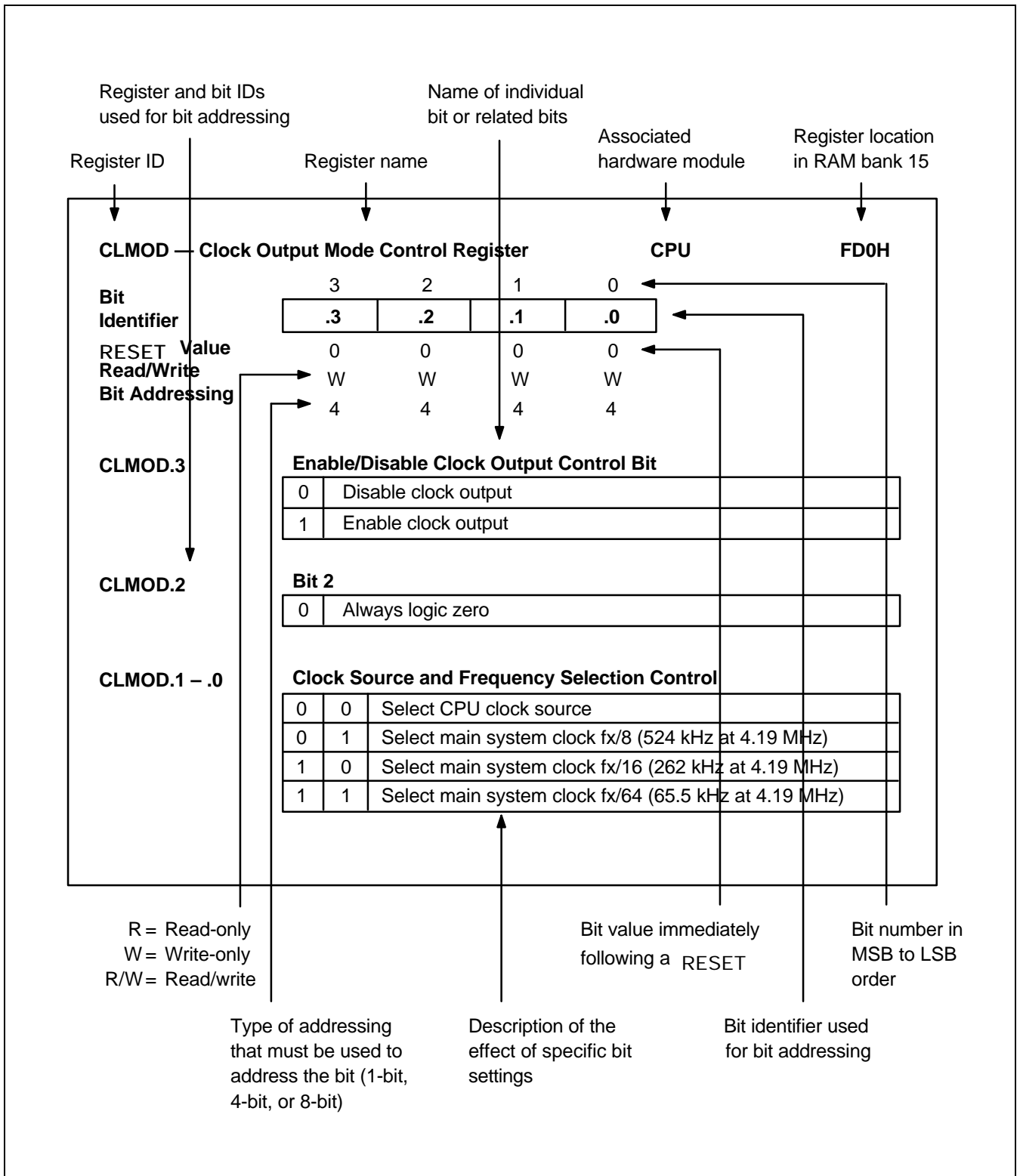


Figure 4-1. Register Description Format

BMOD — Basic Timer Mode Register

BT

F85H

Bit	3	2	1	0
Identifier	.3	.2	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	1/4	4	4	4

BMOD.3**Basic Timer Restart Bit**

1	Restart basic timer, then clear IRQB flag, BCNT and BMOD.3 to logic zero
---	--

BMOD.2 – .0**Input Clock Frequency and Signal Stabilization Interval Control Bits**

0	0	0	Input clock frequency: Signal stabilization interval:	$f_x/2^{12}$ (1.02 kHz) $2^{20}/f_x$ (250 ms)
0	1	1	Input clock frequency: Signal stabilization interval:	$f_x/2^9$ (8.18 kHz) $2^{17}/f_x$ (31.3 ms)
1	0	1	Input clock frequency: Signal stabilization interval:	$f_x/2^7$ (32.7 kHz) $2^{15}/f_x$ (7.82 ms)
1	1	1	Input clock frequency: Signal stabilization interval:	$f_x/2^5$ (131 kHz) $2^{13}/f_x$ (1.95 ms)

NOTES:

- Signal stabilization interval is the time required to stabilize clock signal oscillation after stop mode is terminated by an interrupt. The stabilization interval can also be interpreted as "Interrupt Interval Time".
- When a RESET occurs, the oscillation stabilization time is 31.3 ms ($2^{17}/f_x$) at 4.19 MHz.
- 'fx' is the system clock rate given a clock frequency of 4.19 MHz.

BUZMOD — Buzzer Mode Register

F88H

Bit	3	2	1	0
Identifier	.3	"0"	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

BUZMOD.3**Buzzer Mode Register Bit**

0	Disable buzzer (BUZ) signal output
1	Enable buzzer (BUZ) signal output

BUZMOD.2**Bit 2**

0	Always logic zero
---	-------------------

BUZMOD.1 – .0**Clock Source and Frequency Selection Control Bits**

0	0	0.5 kHz buzzer (BUZ) signal output
0	1	1 kHz buzzer (BUZ) signal output
1	0	2 kHz buzzer (BUZ) signal output
1	1	4 kHz buzzer (BUZ) signal output

NOTE: System clock frequency (fx) is assumed to be 4.19 MHz.

CLMOD — Clock Output Mode Register

CPU

FD0H

Bit	3	2	1	0
Identifier	.3	"0"	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

CLMOD.3 **Enable/Disable Clock Output Control Bit**

0	Disable clock output
1	Enable clock output

CLMOD.2 **Bit 2**

0	Always logic zero
---	-------------------

CLMOD.1 – .0 **Clock Source and Frequency Selection Control Bits**

0	0	Select CPU clock source $fx/4$, $fx/8$, or $fx/64$ (1.05 MHz, 524 kHz, or 65.6 kHz)
0	1	Select system clock $fx/8$ (524 kHz)
1	0	Select system clock $fx/16$ (262 kHz)
1	1	Select system clock $fx/64$ (65.5 kHz)

NOTE: 'fx' is the system clock, given a clock frequency of 4.19 MHz.

DADATA — D/A Converter Data Register

FD7H, FD6H

Bit	7	6	5	4	3	2	1	0
Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8

DADATA.7 – .0

DATATA.7	DATATA.6	DATATA.5	DATATA.4	DATATA.3	DATATA.2	DATATA.1	DATATA.0	V_{DAO}
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	$V_{DD}/2^1$
0	1	0	0	0	0	0	0	$V_{DD}/2^2$
0	0	1	0	0	0	0	0	$V_{DD}/2^3$
0	0	0	1	0	0	0	0	$V_{DD}/2^4$
0	0	0	0	1	0	0	0	$V_{DD}/2^5$
0	0	0	0	0	1	0	0	$V_{DD}/2^6$
0	0	0	0	0	0	1	0	$V_{DD}/2^7$
0	0	0	0	0	0	0	1	$V_{DD}/2^8$

NOTE: These are the values determined by setting just one-bit of DADATA.0–DADATA.7. Other value of DAO can be obtained with superimposition.

$$V_{DAO} = V_{DD} \times \frac{n}{256} \quad (n = 0-255, \text{ DADATA value})$$

DAMOD — D/A Converter Mode Register

FD5H

Bit	3	2	1	0
Identifier	"0"	"0"	"0"	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	1/4	1/4	1/4	1/4

DAMOD.3 – .1**Bit 3–1**

0	Always logic zero
---	-------------------

DAMOD.0**Digital-to-Analog Converter Enable Bit**

0	Disable digital-to-analog converter
1	Enable digital-to-analog converter

IE0, 1, IRQ0, 1 — INT0, 1 Interrupt Enable/Request Flags CPU FBEH

Bit	3	2	1	0
Identifier	IE1	IRQ1	IE0	IRQ0
RESET Value	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W
Bit Addressing	1/4	1/4	1/4	1/4

IE1 INT1 Interrupt Enable Flag

0	Disable interrupt request at the INT1 pin
1	Enable interrupt request at the INT1 pin

IRQ1 INT1 Interrupt Request Flag

–	Generate INT1 interrupt (This bit is set and cleared by hardware when rising or falling edge is detected at INT1 pin.)
---	--

IE0 INT0 Interrupt Enable Flag

0	Disable interrupt request at the INT0 pin
1	Enable interrupt request at the INT0 pin

IRQ0 INT0 Interrupt Request Flag

–	Generate INT0 interrupt (This bit is set and cleared automatically by hardware when rising or falling edge is detected at INT0 pin.)
---	--

IEB, IRQB — INTB Interrupt Enable/Request Flags

CPU

FB8H

Bit	3	2	1	0
Identifier	"0"	"0"	IEB	IRQB
RESET Value	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W
Bit Addressing	1/4	1/4	1/4	1/4

.3 – .2

Bits 3–2

0	Always logic zero
---	-------------------

IEB

INTB Interrupt Enable Flag

0	Disable INTB interrupt request
1	Enable INTB interrupt request

IRQB

INTB Interrupt Request Flag

–	Generate INTB interrupt (bit is set and cleared automatically by hardware when reference interval signal received from basic timer.)
---	--

IEK, IRQK — Key Interrupt Enable/Request Register

CPU

FBFH

Bit	3	2	1	0
Identifier	"0"	"0"	IEK	IRQK
RESET Value	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W
Bit Addressing	1/4	1/4	1/4	1/4

.3 – .2

Bits 3–2

0	Always logic zero
---	-------------------

IEK

Key Interrupt Request Enable Flag

0	Disable INTK interrupt request at the KS0–KS1 pins
1	Enable INTK interrupt request at the KS0–KS1 pins

IRQK

Key Interrupt Request Flag

–	Generate INTK interrupt. (This bit is set when falling edge is detected at any one of the KS0–KS1 pins. INTK is a quasi-interrupt and IRQK must be cleared by software.)
---	--

IETO, IRQT0 — INTT0 Interrupt Enable/Request Flags

CPU

FBCH

Bit	3	2	1	0
Identifier	"0"	"0"	IETO	IRQT0
RESET Value	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W
Bit Addressing	1/4	1/4	1/4	1/4

.3 – .2

Bits 3–2

0	Always logic zero
---	-------------------

IETO

INTT0 Interrupt Enable Flag

0	Disable INTT0 interrupt request
1	Enable INTT0 interrupt request

IRQT0

INTT0 Interrupt Request Flag

–	Generate INTT0 interrupt (bit is set and cleared automatically by hardware when contents of TCNT0 and TREF0 registers match.)
---	---

IMOD0 — External Interrupt 0 (INT0) Mode Register

CPU

FB4H

Bit	3	2	1	0
Identifier	"0"	"0"	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

IMOD0.3 – .2**Bit 3–2**

0	Always logic zero
---	-------------------

IMOD0.1 – .0**External Interrupt Mode Control Bits**

0	0	Interrupt request are triggered by a rising signal edge
0	1	Interrupt request are triggered by a falling signal edge
1	0	Interrupt request are triggered by both rising and falling signal edges
1	1	Interrupt request flag (IRQ0) cannot be set to logic one

IMOD1 — External Interrupt 1 (INT1) Mode Register

CPU

FB5H

Bit	3	2	1	0
Identifier	"0"	"0"	"0"	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

IMOD1.3 – .1**Bits 3–1**

0	Always logic zero
---	-------------------

IMOD1.0**External Interrupt 1 Edge Detection Control Bit**

0	Rising edge detection
1	Falling edge detection

IMODK — Key Interrupt Mode Register

CPU

FB6H

Bit	3	2	1	0
Identifier	"0"	"0"	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

IMODK.3 – .2**Bits 3–2**

0	Always logic zero
---	-------------------

IMODK.1 – .0**Key Interrupt Edge Detection Selection Bit**

0	0	Disable key interrupt
0	1	Select falling edge at KS0
1	0	Select falling edge at KS1
1	1	Select falling edge at KS0–KS1

NOTE: If one of KS0 and KS1 is in Low input (or Low output) state, the key interrupt cannot be occurred.
Refer to the paged 7-11.

IPR — Interrupt Priority Register

CPU

FB2H

Bit	3	2	1	0
Identifier	IME	.2	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	1/4	4	4	4

IME	Interrupt Master Enable Bit		
	0	Inhibit all interrupt processing	
1	Enable processing for all interrupt service requests		

IPR.2 – .0	Interrupt Priority Assignment Bits			
	0	0	0	Process all interrupt requests at low priority
	0	0	1	Process INTB interrupts only
	0	1	0	Process INT0 interrupts only
	0	1	1	Process INT1 interrupts only
1	0	1	Process INTT0 interrupts only	

PCON — Clock Power Control Register

CPU

FB3H

Bit	3	2	1	0
Identifier	.3	.2	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

PCON.3 – .2**CPU Operating Mode Control Bits**

0	0	Enable normal CPU operating mode
0	1	Initiate idle power-down mode
1	0	Initiate stop power-down mode

PCON.1 – .0**CPU Clock Frequency Selection Bits**

0	0	Select fx/64
1	0	Select fx/8
1	1	Select fx/4

NOTE: fx = system clock

PMG1 — Port I/O Mode Flags (Group 1: Ports 0, 1)

I/O FE9H, FE8H

Bit	7	6	5	4	3	2	1	0
Identifier	PM1.3	PM1.2	PM1.1	PM1.0	PM0.3	PM0.2	PM0.1	PM0.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8

PM1.3 P1.3 I/O Mode Selection Flag

0	Set P1.3 to input mode
1	Set P1.3 to output mode

PM1.2 P1.2 I/O Mode Selection Flag

0	Set P1.2 to input mode
1	Set P1.2 to output mode

PM1.1 P1.1 I/O Mode Selection Flag

0	Set P1.1 to input mode
1	Set P1.1 to output mode

PM1.0 P1.0 I/O Mode Selection Flag

0	Set P1.0 to input mode
1	Set P1.0 to output mode

PM0.3 P0.3 I/O Mode Selection Flag

0	Set P0.3 to input mode
1	Set P0.3 to output mode

PM0.2 P0.2 I/O Mode Selection Flag

0	Set P0.2 to input mode
1	Set P0.2 to output mode

PM0.1 P0.1 I/O Mode Selection Flag

0	Set P0.1 to input mode
1	Set P0.1 to output mode

PM0.0 P0.0 I/O Mode Selection Flag

0	Set P0.0 to input mode
1	Set P0.0 to output mode

PMG2 — Port I/O Mode Flags (Group 2: Port 2)

I/O FEBH, FEAH

Bit	7	6	5	4	3	2	1	0
Identifier	"0"	"0"	"0"	"0"	"0"	"0"	"0"	PM2.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8

.7 – .1

Bits 7–1

0	Always logic zero
---	-------------------

PM2.0

P2.0 I/O Mode Selection Flag

0	Set P2.0 to input mode
1	Set P2.0 to output mode

PMG3 — Port I/O Mode Flags (Group 3: Port 4, 5)**I/O FEDH, FECH**

Bit	7	6	5	4	3	2	1	0
Identifier	PM5.3	PM5.2	PM5.1	PM5.0	PM4.3	PM4.2	PM4.1	PM4.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8

PM5.3 P5.3 I/O Mode Selection Flag

0	Set P5.3 to input mode
1	Set P5.3 to output mode

PM5.2 P5.2 I/O Mode Selection Flag

0	Set P5.2 to input mode
1	Set P5.2 to output mode

PM5.1 P5.1 I/O Mode Selection Flag

0	Set P5.1 to input mode
1	Set P5.1 to output mode

PM5.0 P5.0 I/O Mode Selection Flag

0	Set P5.0 to input mode
1	Set P5.0 to output mode

PM4.3 P4.3 I/O Mode Selection Flag

0	Set P4.3 to input mode
1	Set P4.3 to output mode

PM4.2 P4.2 I/O Mode Selection Flag

0	Set P4.2 to input mode
1	Set P4.2 to output mode

PM4.1 P4.1 I/O Mode Selection Flag

0	Set P4.1 to input mode
1	Set P4.1 to output mode

PM4.0 P4.0 I/O Mode Selection Flag

0	Set P4.0 to input mode
1	Set P4.0 to output mode

PNE — N-channel Open-drain Enable Register

I/O

FDAH

Bit	7	6	5	4	3	2	1	0
Identifier	PNE5.3	PNE5.2	PNE5.1	PNE5.0	PNE4.3	PNE4.2	PNE4.1	PNE4.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8

PNE5.3 P5.3 N-channel Open-drain Enable Bit

0	Set P5.3 Open-drain Disabled
1	Set P5.3 Open-drain Enabled

PNE5.2 P5.2 N-channel Open-drain Enable Bit

0	Set P5.2 Open-drain Disabled
1	Set P5.2 Open-drain Enabled

PNE5.1 P5.1 N-channel Open-drain Enable Bit

0	Set P5.1 Open-drain Disabled
1	Set P5.1 Open-drain Enabled

PNE5.0 P5.0 N-channel Open-drain Enable Bit

0	Set P5.0 Open-drain Disabled
1	Set P5.0 Open-drain Enabled

PNE4.3 P4.3 N-channel Open-drain Enable Bit

0	Set P4.3 Open-drain Disabled
1	Set P4.3 Open-drain Enabled

PNE4.2 P4.2 N-channel Open-drain Enable Bit

0	Set P4.2 Open-drain Disabled
1	Set P4.2 Open-drain Enabled

PNE4.1 P4.1 N-channel Open-drain Enable Bit

0	Set P4.1 Open-drain Disabled
1	Set P4.1 Open-drain Enabled

PNE4.0 P4.0 N-channel Open-drain Enable Bit

0	Set P4.0 Open-drain Disabled
1	Set P4.0 Open-drain Enabled

PSW — Program Status Word

CPU

FB1H, FB0H

Bit	7	6	5	4	3	2	1	0
Identifier	C	SC2	SC1	SC0	IS1	IS0	EMB	ERB
RESET Value	(1)	0	0	0	0	0	0	0
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W
Bit Addressing	(2)	8	8	8	1/4	1/4	1	1

C Carry Flag

0	No overflow or borrow condition exists
1	An overflow or borrow condition does exist

SC2 – SC0 Skip Condition Flags

0	No skip condition exists; no direct manipulation of these bits is allowed
1	A skip condition exists; no direct manipulation of these bits is allowed

IS1, IS0 Interrupt Status Flags

0	0	Service all interrupt requests
0	1	Service only the high-priority interrupt(s) as determined in the interrupt priority register (IPR)
1	0	Do not service interrupt requests any more
1	1	Undefined

EMB Enable Data Memory Bank Flag

0	Restrict program access to data memory to bank 15 (F80H–FFFH) and to the locations 000H–07FH in the bank 0 only
1	Enable full access to data memory banks 0, 1, and 15

ERB Enable Register Bank Flag

0	Select register bank 0 as working register area
1	Select register banks 0, 1, 2, or 3 as working register area in accordance with the select register bank (SRB) instruction operand

NOTES:

1. The value of the carry flag is undefined after a RESET occurs during the normal operation. If a RESET occurs during power-down mode (IDLE or STOP), the current value of the carry flag is retained.
2. The carry flag can only be addressed by a specific set of 1-bit manipulation instructions. See Section 2 for detailed information.

PUMOD — Pull-up Register Mode Register

I/O FDDH, FDCH

Bit	7	6	5	4	3	2	1	0
Identifier	"0"	"0"	PUR5	PUR4	"0"	PUR2	PUR1	PUR0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8

.7 – .6

Bits 7–6

0	Always cleared to logic zero
---	------------------------------

PUR5

Connect/Disconnect Port 5 Pull-up Resistor Control Bit

0	Disconnect port 5 pull-up resistor
1	Connect port 5 pull-up resistor

PUR4

Connect/Disconnect Port 4 Pull-up Resistor Control Bit

0	Disconnect port 4 pull-up resistor
1	Connect port 4 pull-up resistor

.3

Bit 3

0	Always cleared to logic zero
---	------------------------------

PUR2

Connect/Disconnect Port 2 Pull-up Resistor Control Bit

0	Disconnect port 2 pull-up resistor
1	Connect port 2 pull-up resistor

PUR1

Connect/Disconnect Port 1 Pull-up Resistor Control Bit

0	Disconnect port 1 pull-up resistor
1	Connect port 1 pull-up resistor

PUR0

Connect/Disconnect Port 0 Pull-up Resistor Control Bit

0	Disconnect port 0 pull-up resistor
1	Connect port 0 pull-up resistor

TMOD0 — Timer/Counter 0 Mode Register

T/C0 F91H, F90H

Bit	3	2	1	0	3	2	1	0
Identifier	"0"	.6	.5	.4	.3	.2	"0"	"0"
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	1/8	8	8	8

.7

Bit 7

0	Always logic zero
---	-------------------

.6 – .4

Timer/Counter 0 Input Clock Selection Bits

0	0	0	External clock input at TCLK pin on rising edge
0	0	1	External clock input at TCLK pin on falling edge
1	0	0	Internal system clock (fx) of 4.19 MHz/2 ¹⁰ (4.09 kHz)
1	0	1	Selected clock: fx/2 ⁶ (65.5 kHz)
1	1	0	Selected clock: fx/2 ⁴ (262 kHz)
1	1	1	Selected clock: fx (4.19 MHz)

.3

Clear Counter and Resume Counting Control Bit

1	Clear TCNT0, IRQT0, and TOL0 and resume counting immediately. (This bit is cleared automatically when counting starts.)
---	---

.2

Enable/Disable Timer/Counter 0 Bit

0	Disable timer/counter 0; retain TCNT0 contents
1	Enable timer/counter 0

.1 – .0

Bits 1–0

0	Always logic zero
---	-------------------

NOTE: System clock frequency (fx) is assumed to be 4.19 MHz.

TOE0 — Timer Output Enable Flag

T/C

F92H

Bit	3	2	1	0
Identifier	"0"	TOE0	"0"	"0"
RESET Value	0	0	0	0
Read/Write	R/W	R/W	R/W	W
Bit Addressing	1/4	1/4	1/4	1/4

.3

Bit 3

0	Always logic zero
---	-------------------

TOE0

Timer/Counter 0 Output Enable Flag

0	Disable timer/counter 0 output to the TCLO0 pin
1	Enable timer/counter 0 output to the TCLO0 pin

.1 – .0

Bits 1–0

0	Always logic zero
---	-------------------

WDMOD — Watchdog Timer Mode Register**F99H, F98H**

Bit	7	6	5	4	3	2	1	0
Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	1	0	1	0	0	1	0	1
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8

.7 - .0**Watchdog Timer Enable/Disable Control**

5AH	Disable watch-dog timer function
Any other value	Enable watch-dog timer function

WDFLAG — Watchdog Timer Flag**F9AH**

Bit	3	2	1	0
Identifier	WDTCF	"0"	"0"	"0"
RESET Value	0	0	0	0
Read/Write	W	–	–	–
Bit Addressing	1	–	–	–

.3**Watch-dog timer's counter clear bit**

1	Clear and restart the watch-dog timer's counter
---	---

NOTE: Instructions that clear the watch-dog timer ("BITS WDTCF") should be executed at proper points in a program within a given period. If the instructions are not executed within a given period and the watch-dog timer overflows, a RESET signal is generated and the system is restarted in a reset status.

6 OSCILLATOR CIRCUITS

OVERVIEW

The S3C7544 has a system clock circuit. The CPU and peripheral hardware operate on the system clock frequency supplied through these on-chip circuits. Specifically, a clock is required by the following peripheral modules:

- Basic timer
- Timer/counter 0
- Clock output circuit
- Buzzer

The system clock frequency can be divided by 4, 8, or 64. By manipulating PCON bits 1 and 0, you can select one of the following frequencies as the CPU clock.

$$\frac{f_x}{4}, \frac{f_x}{8}, \frac{f_x}{64}$$

When the PCON register is cleared to zero after a RESET, the normal CPU operating mode is enabled and, a system clock of $f_x/64$ is selected.

Bits 3 and 2 of the PCON register can be manipulated by a STOP or IDLE instruction to engage stop or idle power-down mode.

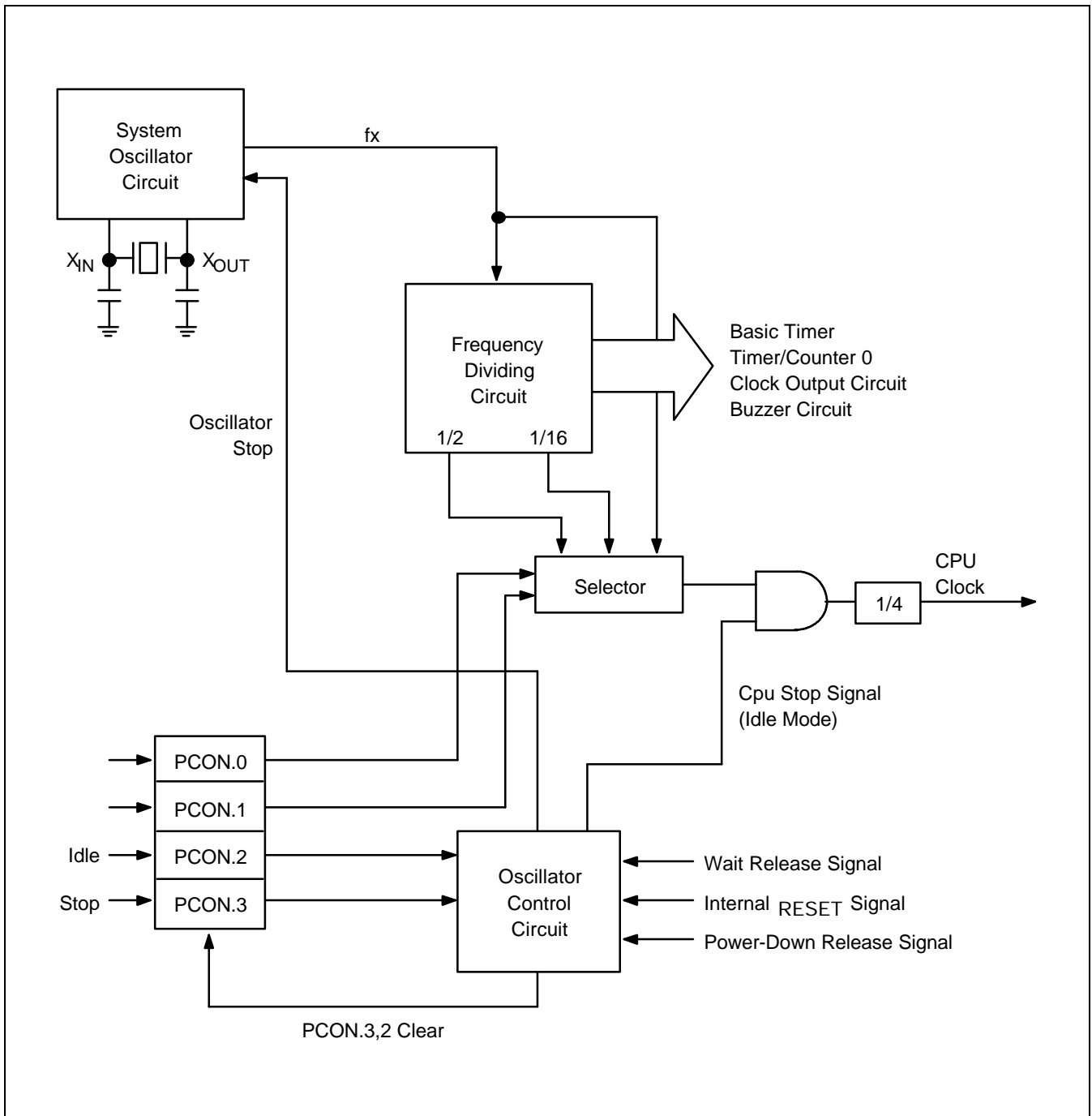


Figure 6-1. Clock Circuit Diagram

SYSTEM OSCILLATOR CIRCUITS

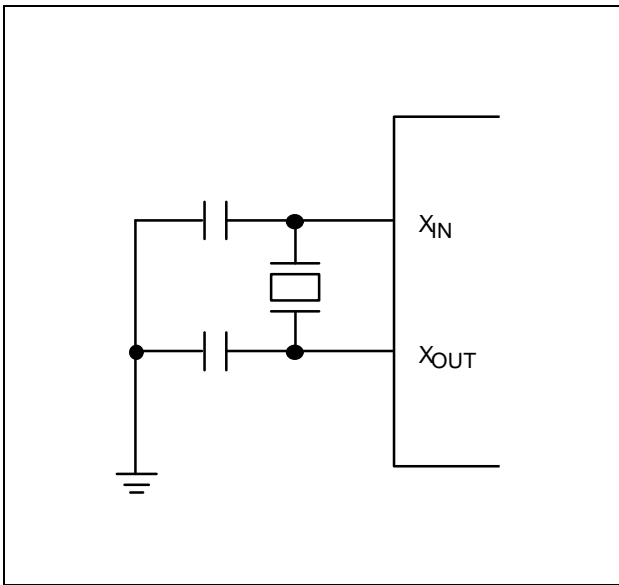


Figure 6-2. Crystal/Ceramic Oscillator

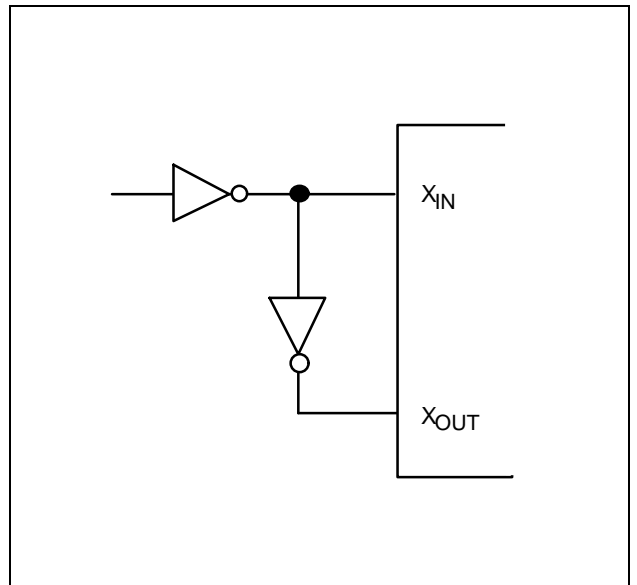
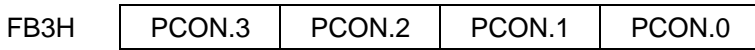


Figure 6-3. External Clock

POWER CONTROL REGISTER (PCON)

The power control register, PCON, is a 4-bit register that is used to select the CPU clock frequency and control CPU operating and power-down modes. PCON is mapped to the RAM address FB3H and can be addressed directly by 4-bit write instructions or by the instructions IDLE and STOP.




PCON bits 3 and 2 are controlled by the STOP and IDLE instructions to engage the idle and stop power-down modes. Idle and stop modes can be initiated by these instruction regardless of the current value of the enable memory bank flag (EMB). PCON bits 1 and 0 are used to select a specific system clock frequency.

A RESET sets PCON register values to logic zero. PCON.1 and PCON.0 divide the frequency (fx) by 4, 8, and 64. PCON.3 and PCON.2 enable normal CPU operating mode.

Table 6-1. Power Control Register (PCON) Organization

PCON Bit Settings		Resulting CPU Operating Mode
PCON.3	PCON.2	
0	0	Normal CPU operating mode
0	1	Idle power-down mode
1	0	Stop power-down mode

PCON Bit Settings		Resulting CPU Clock Frequency
PCON.1	PCON.0	
0	0	fx/64
1	0	fx/8
1	1	fx/4

 **PROGRAMMING TIP — Setting the CPU Clock**

To set the CPU clock to 1.05 MHz at 4.19 MHz:

```

BITS      EMB
SMB      15
LD        A,#3H
LD        PCON,A
    
```

INSTRUCTION CYCLE TIMES

The unit of time that equals one machine cycle varies depending on how the oscillator clock signal is divided (by 4, 8, or 64). Table 6-2 shows corresponding cycle times in microseconds.

Table 6-2. Instruction Cycle Times for CPU Clock Rates

Selected CPU Clock	Resulting Frequency	Oscillation Source	Cycle Time (μ s)
fx/64	65.5 kHz	fx = 4.19 MHz	15.3
fx/8	524.0 kHz		1.91
fx/4	1.05 MHz		0.95

CLOCK OUTPUT MODE REGISTER (CLMOD)

The clock output mode register, CLMOD, is a 4-bit register that is used to enable or disable clock output to the CLO pin and select the CPU clock source and frequency. CLMOD is mapped to the RAM address FD0H and is addressable by 4-bit write instructions only.

FD0H	CLMOD.3	"0"	CLMOD.1	CLMOD.0
------	---------	-----	---------	---------

A RESET clears CLMOD to logic zero, which automatically selects the CPU clock as the clock source (without initiating clock oscillation), and disables clock output.

CLMOD.3 is the enable/disable clock output control bit. CLMOD.1 and CLMOD.0 are used to select one of four possible clock sources and frequencies: normal CPU clock, fx/8, fx/16, or fx/64.

Table 6-3. Clock Output Mode Register (CLMOD) Organization

CLMOD Bit Settings		Resulting Clock Output	
CLMOD.1	CLMOD.0	Clock Source	Frequency
0	0	CPU clock (fx/4, fx/8, fx/64)	1.05 MHz, 524 kHz, 65.5 kHz
0	1	fx/8	524 kHz
1	0	fx/16	262 kHz
1	1	fx/64	65.5 kHz

CLMOD.3	Result of CLMOD.3 Setting
0	Clock output is disabled
1	Clock output is enabled

NOTE: Frequencies assume that fx = 4.19 MHz.

CLOCK OUTPUT CIRCUIT

The clock output circuit, used to output clock pulses to the CLO pin, has the following components:

- 4-bit clock output mode register (CLMOD)
- Clock selector
- Output latch
- Port mode flag
- CLO output pin (P2.2)

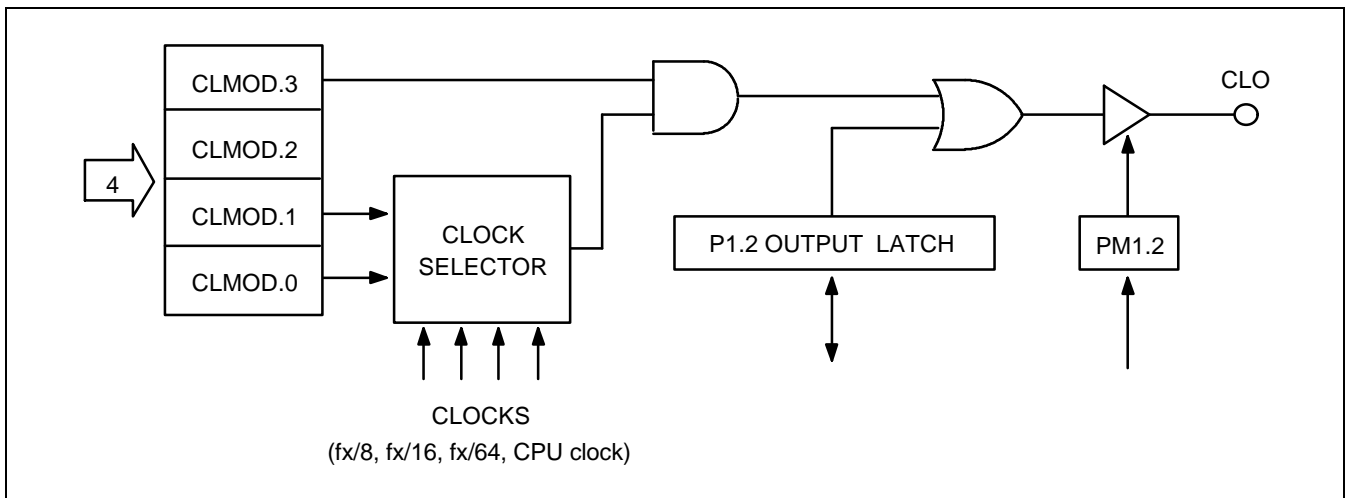


Figure 6-4. CLO Output Pin Circuit Diagram

CLOCK OUTPUT PROCEDURE

To output clock pulses to the CLO pin, follow this general procedure:

1. Disable clock output by clearing CLMOD.3 to logic zero.
2. Set the clock output frequency (CLMOD.1, CLMOD.0).
3. Load a "0" to the output latch of the CLO pin (P1.2).
4. Set the P1.2 mode flag (PM1.2) to output mode.
5. Enable clock output by setting CLMOD.3 to logic one.

PROGRAMMING TIP — CPU Clock Output to the CLO Pin

To output the CPU clock to the CLO pin:

```

BITS      EMB          ; OR BITR EMB
SMB      15
LD       EA,#40H
LD       PMG1,EA      ; P1.2 ← Output mode
BITR    P1.2         ; Clear P1.2 output latch
LD       A,#8H
LD       CLMOD,A
    
```

7 INTERRUPTS

OVERVIEW

The S3C7544 microcontrollers process three types of interrupts:

- Internal interrupts generated by on-chip processes
- External interrupts generated by external peripheral devices
- Quasi-interrupts used for edge detection and clock sources

Table 7-1. Interrupts and Corresponding I/O Pin(s)

Interrupt Type	Interrupt Name	I/O Port Pin(s)
External Interrupts	INT0, INT1	P0.0, P0.1
Internal Interrupts	INTB, INTT0	Not applicable
Quasi-interrupts	INTK	P0.2, P0.3 (KS0, KS1)

The interrupt control circuit has four functional components:

- Interrupt enable flags (IEx)
- Interrupt request flags (IRQx)
- Interrupt priority registers (IME and IPR)
- Power-down release signal circuit

Vectored Interrupts

Interrupt requests may be processed as vectored interrupts in hardware, or they can be generated by program software. A vectored interrupt is generated when the following flags and register settings, corresponding to the specific interrupt, are enabled (set to logic one):

- Interrupt enable flag (IEx)
- Interrupt master enable flag (IME)
- Interrupt request flag (IRQx)
- Interrupt status flags (IS0, IS1)
- Interrupt priority register (IPR)

If all conditions are satisfied, the start address of the interrupt is loaded into the program counter and the program starts executing the service routine from this address.

Vectored Interrupts (Continued)

The EMB and ERB flags for the RAM memory and register banks are stored in the vector address area of the ROM during interrupt service routines. The flags are stored at the beginning of the program with the VENT instruction. Enable flag values are saved during the main routine, as well as the service routines. Any change you make to enable flag values during a service routine is not stored in the vector address.

When an interrupt occurs, the enable flag values before the interrupt is initiated are saved along with the program status word (PSW), and the enable flag values for the interrupt is fetched from the respective vector address.

Then, if required, you can modify the enable flags during the interrupt service routine. When the interrupt service routine returns to the main routine by the IRET instruction, however, the original values saved in the stack are restored and the main program continues program execution with these values.

Software-Generated Interrupts

To generate an interrupt request from software, the program manipulates the IRQx flag appropriately. When the interrupt request value in the IRQx flag is set, it is retained until all other conditions for the interrupt have been met, and the service routine can be initiated.

Multiple Interrupts

By manipulating the two interrupt status flags (IS0 and IS1), you can control service routine initialization and thereby process multiple interrupts simultaneously.

Power-Down Mode Release

An interrupt can be used to release power-down mode (stop or idle). Interrupts for power-down mode release are initiated by setting the corresponding interrupt enable flag. Even if the IME flag is cleared to zero, power-down mode will be released by an interrupt request signal when the interrupt enable flag is set. In such cases, the interrupt routine will not be executed since IME = "0".

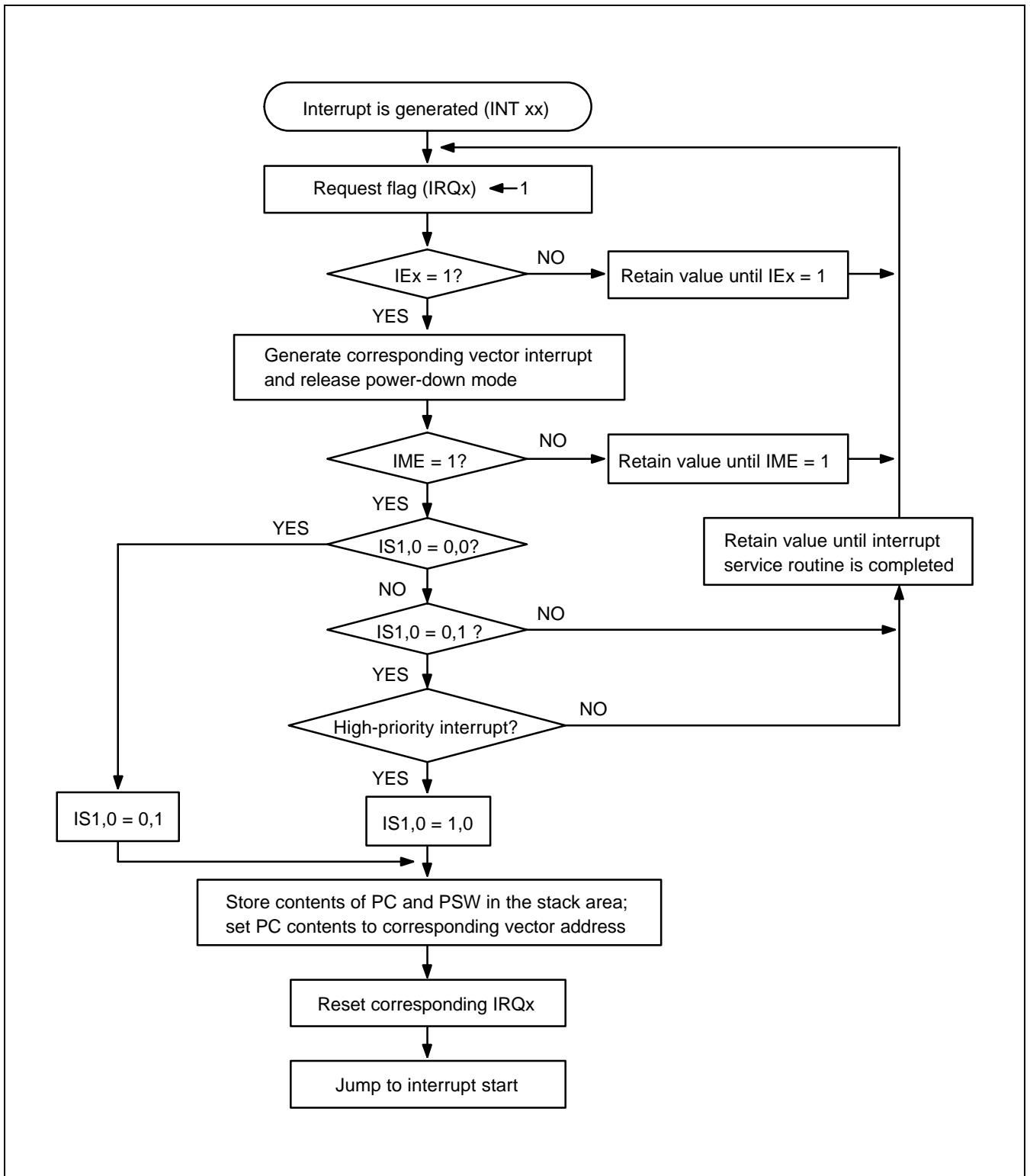


Figure 7-1. Interrupt Execution Flowchart

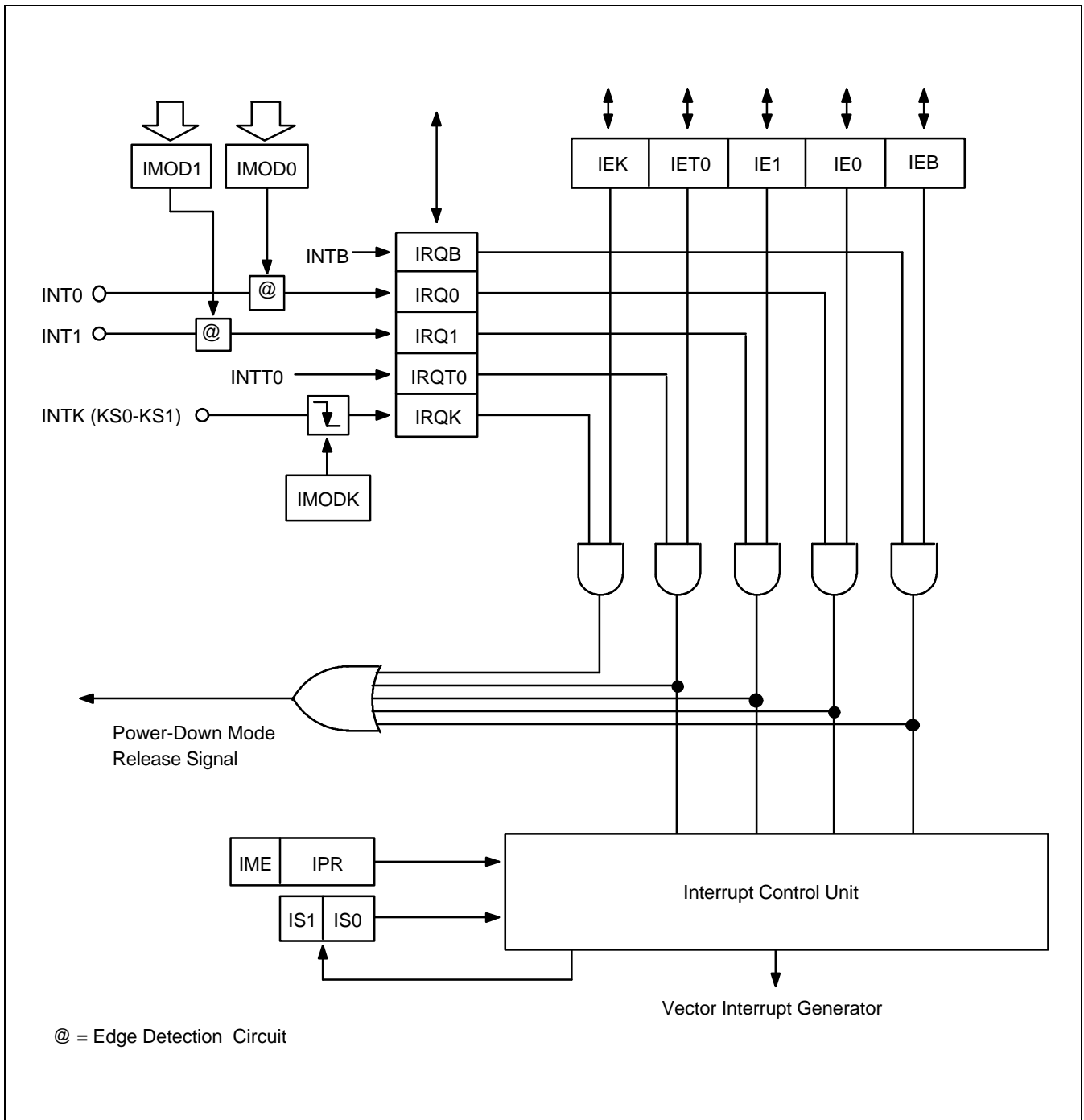


Figure 7-2. Interrupt Control Circuit Diagram

MULTIPLE INTERRUPTS

The interrupt controller can service multiple interrupts in two ways: as two-level interrupts, where either all interrupt requests or only those of highest priority are serviced, or as multi-level interrupts, when the interrupt service routine for a lower-priority request is accepted during the execution of a higher priority routine.

Two-Level Interrupt Handling

Two-level interrupt handling is the standard method for processing multiple interrupts. When the IS1 and IS0 bits of the PSW (FB0H.3 and FB0H.2, respectively) are both logic zero, program execution mode is normal and all interrupt requests are serviced. See Figure 7-3.

Whenever an interrupt request is accepted, IS1 and IS0 are incremented by one ("0" → "1" or "1" → "0"), and the values are stored in the stack along with other PSW bits. After the interrupt routine is serviced, the modified IS1 and IS0 values are automatically restored from the stack by an IRET instruction.

IS0 and IS1 can be manipulated directly by 1-bit write instructions, regardless of the current value of the enable memory bank flag (EMB). Before you can modify an interrupt service flag, however, you must first disable interrupt processing with a DI instruction.

When you set IS1 to "0" and IS0 to "1", you should inhibit all interrupt service routines except for the highest priority interrupt currently defined by the interrupt priority register (IPR).

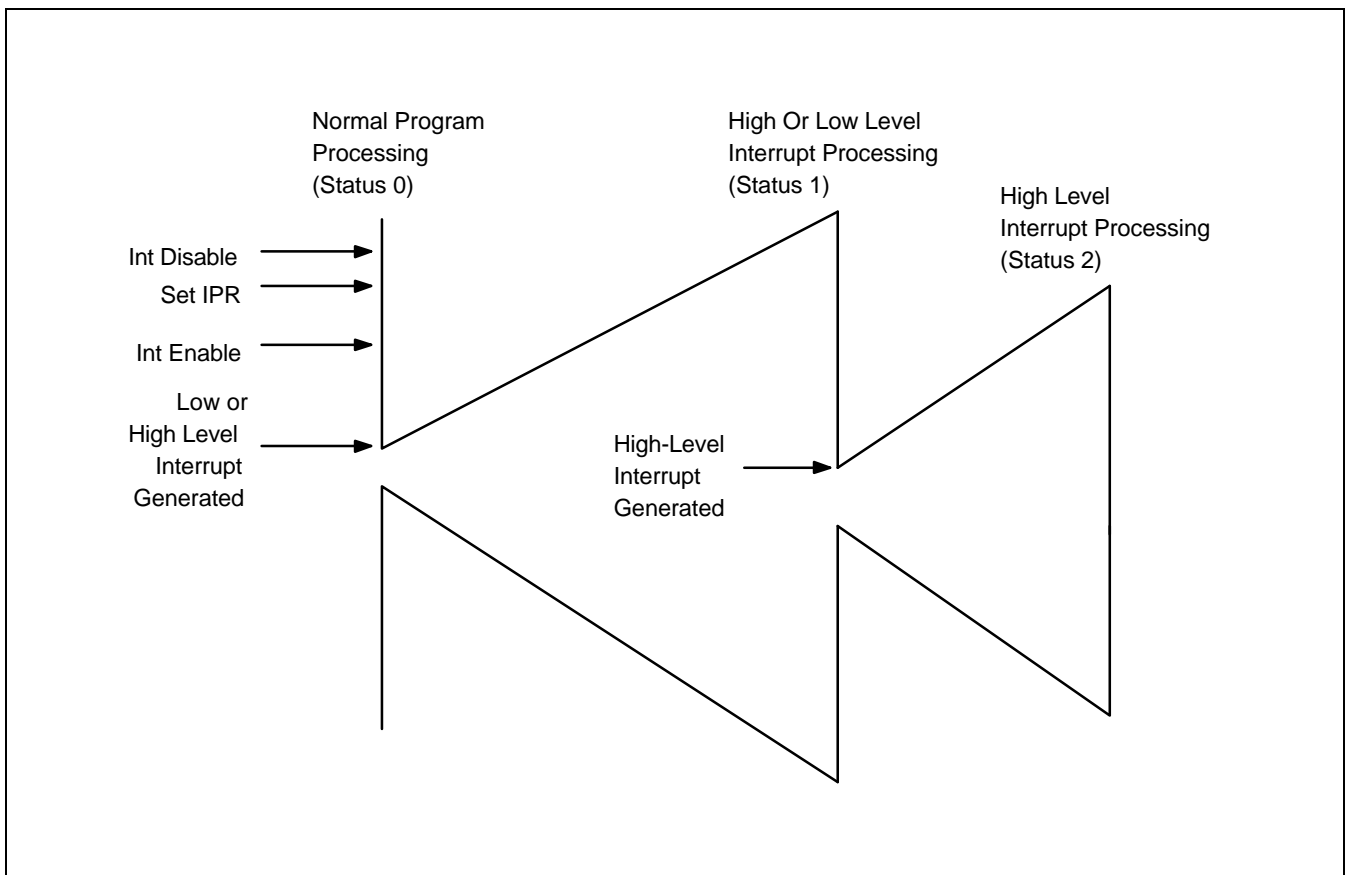


Figure 7-3. Two-Level Interrupt Handling

Multi-Level Interrupt Handling

With multi-level interrupt handling, a lower-priority interrupt request can be executed while a high-priority interrupt is being serviced. This is done by manipulating the interrupt status flags, IS0 and IS1. See Figure 7-4.

When an interrupt is requested during the normal program execution, the interrupt status flags IS0 and IS1 are set to "0" and "1", respectively. This setting allows only highest-priority interrupts to be serviced. When a high-priority request is accepted, both interrupt status flags are then cleared to "0" by software so that a request of any priority level can be serviced. In this way, the high-priority and low-priority requests are serviced in parallel.

Table 7-2. IS1 and IS0 Function

Process Status	Before INT		Effect of ISx Bit Setting	After INT ACK	
	IS1	IS0		IS1	IS0
0	0	0	All interrupt requests are serviced.	0	1
1	0	1	Only high-priority interrupts as determined by the current settings in the IPR register are serviced.	1	0
2	1	0	No additional interrupt requests will be serviced.	-	-
-	1	1	Value undefined	-	-

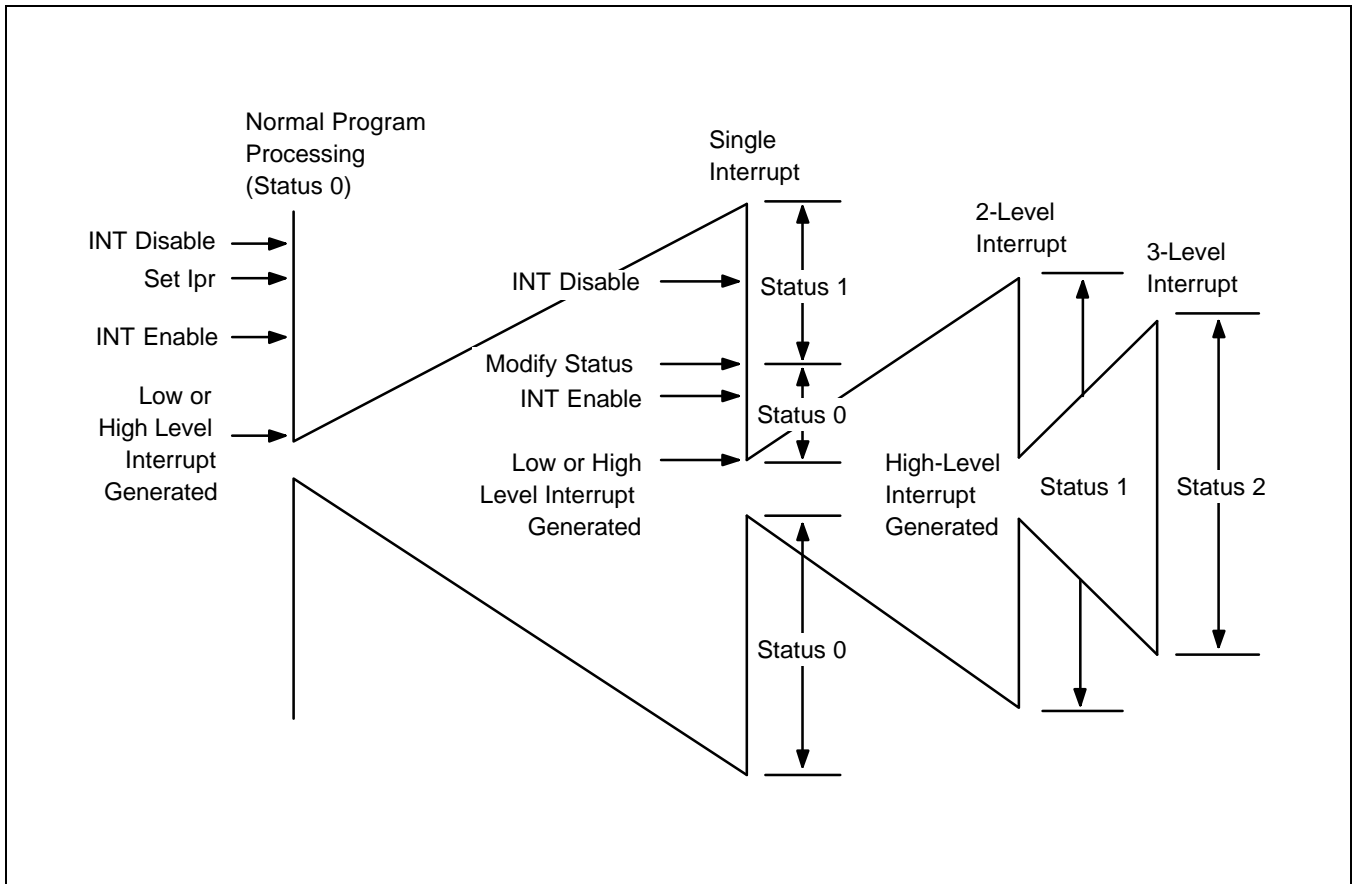


Figure 7-4. Multiple-Level Interrupt Handling

INTERRUPT PRIORITY REGISTER (IPR)

The 4-bit interrupt priority register (IPR) is used to control multi-level interrupt handling. The IPR is mapped to the RAM address FB2H, and its reset value is logic zero. Before the IPR is modified by 4-bit write instructions, all interrupts must first be disabled by a DI instruction.

FB2H	IME	IPR.2	IPR.1	IPR.0
------	-----	-------	-------	-------

By manipulating the IPR settings, you can choose to process all interrupt requests with the same priority level, or you can select one type of interrupt for high-priority processing. A low-priority interrupt can itself be interrupted by a high-priority interrupt, but not by another low-priority interrupt. A high-priority interrupt cannot be interrupted by any other interrupt source.

Table 7-3. Standard Interrupt Priorities

Interrupt	Default Priority
INTB	1
INT0	2
INT1	3
INTT0	4

The MSB of the IPR, the interrupt master enable flag (IME), enables and disables all interrupt processing. Even if an interrupt request flag and its corresponding enable flag are set, a service routine cannot be executed until the IME flag is set to logic one.

The IME flag is mapped to FB2H.3 and can be directly manipulated by EI and DI instructions, regardless of the current enable memory bank (EMB) value.

Table 7-4. Interrupt Priority Register Settings

IPR.2	IPR.1	IPR.0	Result of IPR Bit Setting
0	0	0	Process all interrupt requests at low priority.
0	0	1	Process INTB interrupt at highest priority.
0	1	0	Process INT0 interrupt at highest priority.
0	1	1	Process INT1 interrupt at highest priority.
1	0	1	Process INTT0 interrupt at highest priority.

NOTE: When all interrupts are at low priority (the lower three bits of the IPR register are logic zero), the interrupt generated first has the highest priority. Therefore, the interrupt generated first cannot be superceded by any other interrupt. If two or more interrupt requests are received simultaneously, the priority level is determined according to the standard interrupt priorities in Table 7-3 (e.g., the default priority assigned by hardware when the lower three IPR bits = "0"). In this case, the highest-priority interrupt request is serviced and other interrupts are inhibited. Then, when the high-priority interrupt returns from its service routine by an IRET instruction, the service routine of an interrupt inhibited is started.

 **PROGRAMMING TIP — Setting the INT Interrupt Priority**

Set the INT1 interrupt to high priority:

```

BITS      EMB
SMB       15
DI                ; IPR.3 (IME) ← 0
LD        A,#3H
LD        IPR,A
EI                ; IPR.3 (IME) ← 1
    
```

EXTERNAL INTERRUPT MODE REGISTERS (IMOD0, IMOD1)

The following components are used to process external interrupts at the INT0 and INT1 pin:

- Noise filtering circuit for INT0
- Edge detection circuit
- Two mode registers, IMOD0 and IMOD1

The mode registers are used to control the triggering edge of the input signal. IMOD settings let you choose either the rising or falling edge of the incoming signal at the INT0 and INT1 pins as the interrupt request trigger.

FB4H	"0"	"0"	IMOD0.1	IMOD0.0
FB5H	"0"	"0"	"0"	IMOD1.0

IMOD0 and IMOD1 bits are mapped to the RAM addresses FB4H (IMOD0) and FB5H (IMOD1), and they are addressable by 4-bit write instructions. A RESET clears all IMOD values to logic zero, selecting rising edges as the trigger for incoming interrupt requests.

Table 7-5. IMOD0 and IMOD1 Register Organization

IMOD0	0	0	IMOD0.1	IMOD0.0	Effect of IMOD0 Settings
			0	0	Rising edge detection
			0	1	Falling edge detection
			1	0	Both rising and falling edge detection
			1	1	IRQ0 flag cannot be set to "1"

IMOD1	0	0	0	IMOD1.0	Effect of IMOD1 Settings
				0	Rising edge detection
				1	Falling edge detection

EXTERNAL INTERRUPT 0 and 1 MODE REGISTERS (Continued)

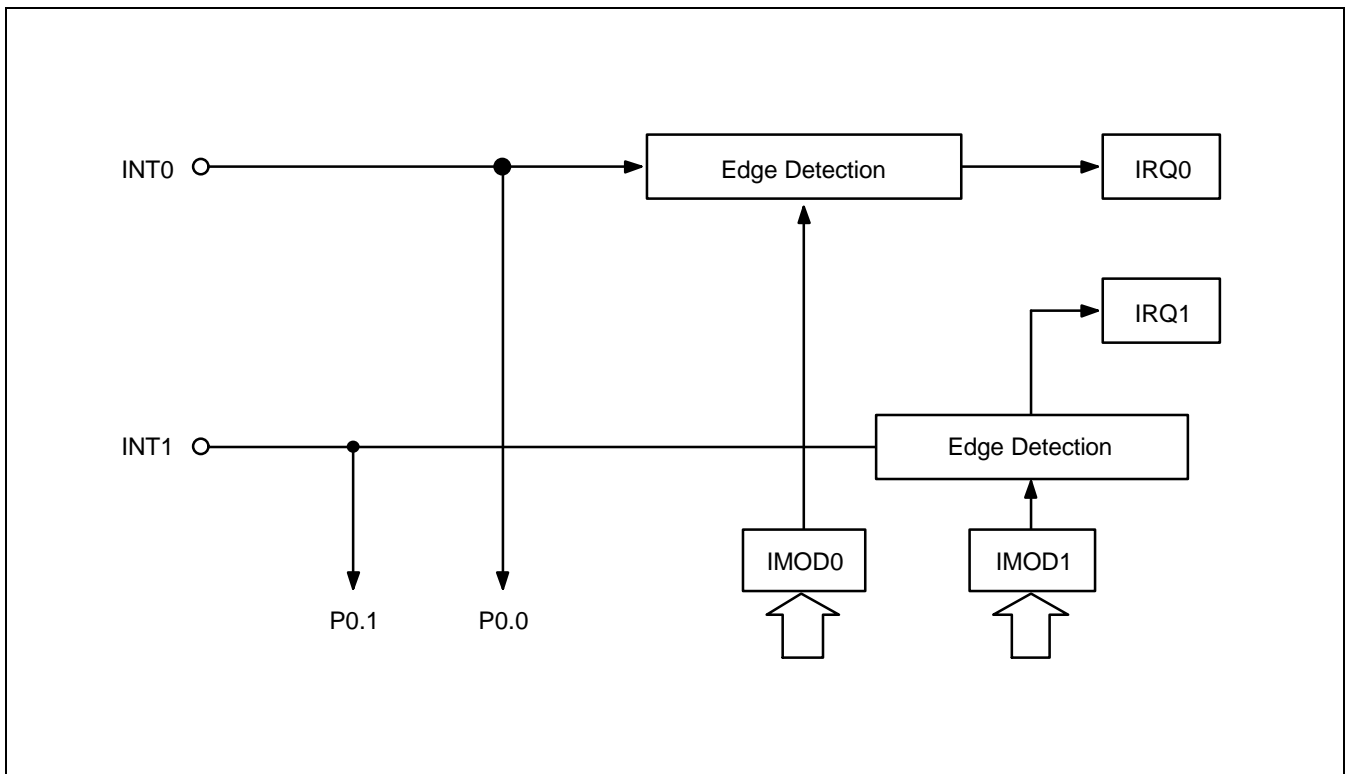


Figure 7-5. Circuit Diagram for INT0 and INT1 Pins

When modifying the IMOD0 and IMOD1 registers, it is possible to accidentally set an interrupt request flag. To avoid unwanted interrupts, take these precautions when writing your programs:

1. Disable all interrupts with a DI instruction.
2. Modify the IMOD0 or IMOD1 register.
3. Clear all relevant interrupt request flags.
4. Enable the interrupt by setting the IEx flag appropriately.
5. Enable all interrupts with an EI instruction.

KEY INTERRUPT MODE REGISTER (IMODK)

The mode register for external interrupts at the KS0–KS1 pins, IMODK, is a 4-bit register at the RAM address FB6H. IMODK is addressable by 4-bit write instructions. A RESET clears all IMODK bits to logic zero.

FB6H	"0"	"0"	IMODK.1	IMODK.0
------	-----	-----	---------	---------

When bits in the IMODK register are set to logic one, INTK uses the falling edge of an incoming signal at corresponding pins as the interrupt request trigger. When a falling edge is detected at any of the pins KS0–KS1, the IRQK flag is set to logic one and a release signal for power-down mode is generated.

If one of KS0 and KS1 is in Low input (or Low output) state, the key interrupt cannot be occurred.

Table 7-6. IMODK Register Bit Settings

IMODK	IMODK.1	IMODK.0	Effect of IMODK Settings
	0	0	Disable key interrupt
	0	1	Select falling edge at KS0
	1	0	Select falling edge at KS1
	1	1	Select falling edge at KS0–KS1

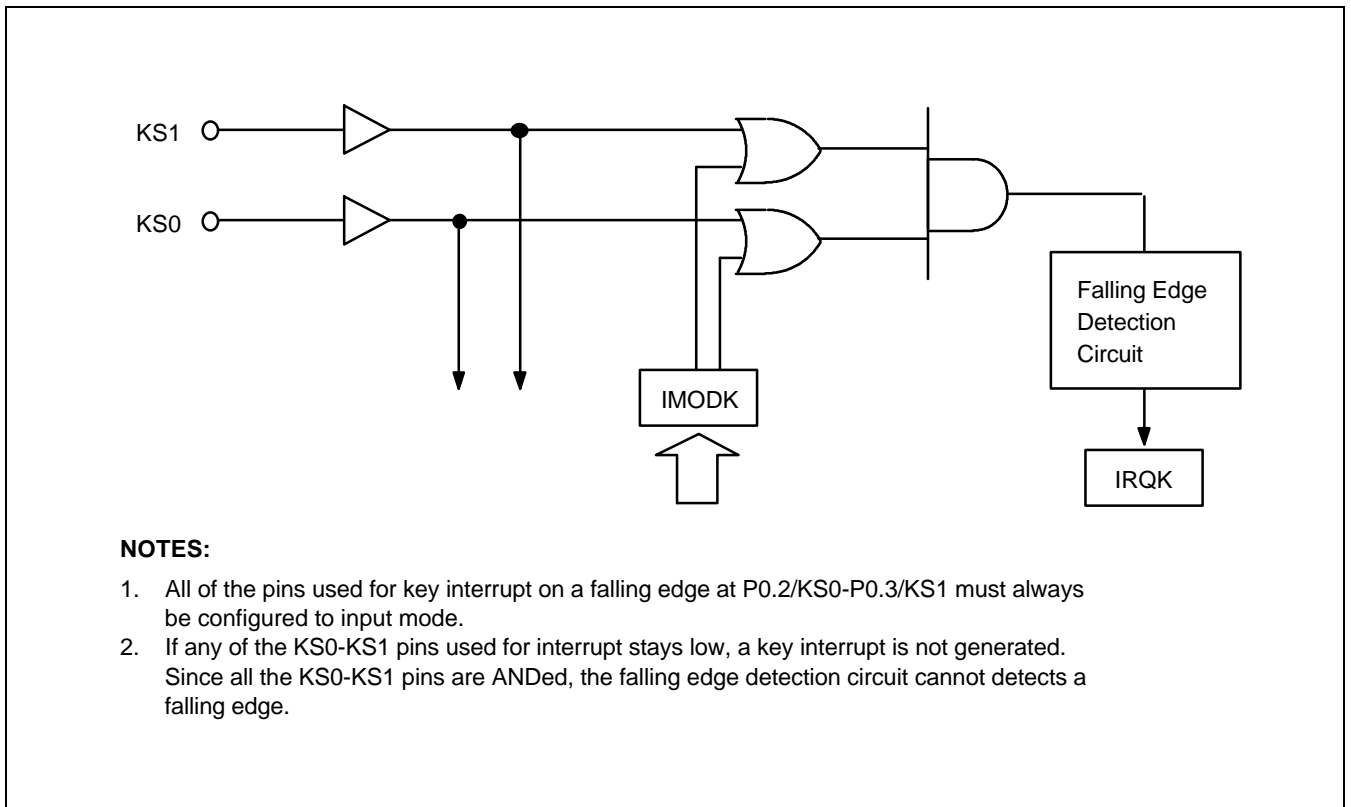


Figure 7-6. Circuit Diagram for KS0–KS1 Pins

PROGRAMMING TIP — Using INTK as a Key Input Interrupt

When the INTK interrupt is used as a key interrupt, the key interrupt pin must be set to input.

1. When KS0–KS1 are selected:

BITS	EMB	
SMB	15	
LD	A,#3H	
LD	IMODK,A	; (IMODK) ← #3H, KS0–KS1 falling edge select
LD	EA,#00H	
LD	PMG1,EA	; P0 ← Input mode
LD	EA,#D1H	
LD	PUMOD,EA	; Enable P0 pull-up resistors

INTERRUPT FLAGS

There are three types of interrupt flags: interrupt request and interrupt enable flags that correspond to each interrupt; the interrupt master enable flag, which enables or disables all interrupt processing.

Interrupt Master Enable Flag (IME)

The interrupt master enable flag, IME, enables or disables all interrupt processing. Therefore, even when an IRQx flag is set and its corresponding IEx flag is enabled, the interrupt service routine is not executed until the IME flag is set to logic one.

The IME flag is located in the IPR register (IPR.3), and mapped to the bit address FB2H.3. It can directly be manipulated by EI and DI instructions, regardless of the current value of the enable memory bank flag (EMB).

Interrupt Enable Flags (IEx)

IEx flags, when set to logic one, enable specific interrupt requests to be serviced. When the interrupt request flag is set to logic one, an interrupt will not be serviced until its corresponding IEx flag is also enabled.

Interrupt enable flags are mapped to the RAM address area FB8H–FBFH, and can be read, written, or tested directly by 1-bit instructions (BITS and BITR). IEx flags can be addressed directly at their specific RAM addresses, regardless of the current value of the enable memory bank (EMB) flag.

Interrupt Request Flags (IRQx)

Interrupt request flags, located in the RAM area FB8H-FBFH, are read/write addressable by 1-bit or 4-bit instructions. IRQx flags can be addressed directly at their specific RAM addresses, regardless of the current value of the enable memory bank (EMB) flag.

When a specific IRQx flag is set to logic one, the corresponding interrupt request is generated. The flag is then automatically cleared to logic zero by hardware when the interrupt is serviced. An exception is the key interrupt request flag IRQK, which must be cleared by software after the interrupt service routine is executed. IRQx flags are also used to execute interrupt requests from software. In summary, follow these guidelines for using IRQx flags:

1. IRQx is set to request an interrupt when an interrupt meets the set condition for interrupt generation.
2. IRQx is set to "1" then cleared by hardware when the interrupt is serviced (except for IRQK).
3. When IRQx is set to "1" by software, an interrupt is generated.

INTERRUPT MASTER ENABLE FLAG (IME)

The interrupt master enable flag, IME, inhibits or enables all interrupt processing. Therefore, even when an IRQx flag and its corresponding IEx flag are enabled, an interrupt request will not be serviced until the IME flag is set to logic one. The IME flag is the most significant bit of the 4-bit IPR register at the RAM location FB2H.

IME	IPR.2	IPR.1	IPR.0	Effect of Bit Settings
0				Inhibit all interrupts
1				Enable all interrupts

You can manipulate the IME flag using EI and DI instructions, regardless of the current value of the enable memory bank (EMB) flag.

INTERRUPT ENABLE FLAGS (IEx)

Interrupt enable flags are used to control the execution of service routines for specific interrupt requests. The enable flag has priority over a request flag — even if the IRQx flag is enabled, the interrupt request will not be serviced until the corresponding IEx flag is set to logic one.

Using 1-bit or 4-bit instructions and direct addressing, you can read, write, or test IEx (and IRQx) flags regardless of the current enable memory bank (EMB) value. The IEx and IRQx flags are mapped to the RAM area FB8H–FBFH.

Table 7-7. Interrupt Enable and Interrupt Request Flag Addresses

Address	Bit 3	Bit 2	Bit 1	Bit 0
FB8H	0	0	IEB	IRQB
FBCH	0	0	IET0	IRQT0
FBEH	IE1	IRQ1	IE0	IRQ0
FBFH	0	0	IEK	IRQK

NOTES:

1. IEx refers generically to all interrupt enable flags.
2. IRQx refers generically to all interrupt request flags.
3. IEx = 0 is interrupt disable mode.
4. IEx = 1 is interrupt enable mode.

INTERRUPT REQUEST FLAGS (IRQx)

When an interrupt request flag (IRQx) is set, a software-generated interrupt is enabled for the corresponding interrupt. IRQx flags can be written by 1- or 4-bit RAM control instructions. IRQx flags are then cleared automatically when the interrupt is serviced. An exception is the key interrupt request flag, IRQK, which must be cleared by software after the interrupt service routine is executed.

Table 7-8. Interrupt Request Flag Conditions and Priorities

Interrupt Source	Internal / External	Pre-condition for IRQx Flag Setting	Interrupt Priority	IRQx Flag Name
INTB	I	Reference time interval signal from basic timer	1	IRQB
INT0	E	Rising or falling edge detected at INT0 pin	2	IRQ0
INT1	E	Rising or falling edge detected at INT1 pin	3	IRQ1
INTT0	I	Signals for TCNT0 and TREF0 registers coincide	5	IRQT0
INTK ^(note)	E	Falling edge is detected at any one of the KS0–KS1 pins	–	IRQK

NOTE: INTK is quasi-interrupt and used only for testing incoming signals.

8

POWER-DOWN

OVERVIEW

The S3C7544 microcontroller has two power-down modes reducing power consumption: idle and stop. Idle mode is initiated by the IDLE instruction and stop mode by the STOP instruction. (Several NOP instructions must always follow an IDLE or STOP instruction in a program.) In idle mode, the CPU clock stops while peripherals and the oscillation source continue to operate normally.

When a RESET occurs during the normal operation or a power-down mode, a reset operation is initiated and the CPU enters idle mode. When the standard oscillation stabilization time interval (31.3 ms at 4.19 MHz) has elapsed, the normal CPU operation resumes.

In stop mode, system clock oscillation is halted (assuming it currently has been operating), and peripheral hardware components are powered-down. The effect of stop mode on specific peripheral hardware components — CPU, basic timer, timer/counters 0, and — and on external interrupt requests, is detailed in Table 8-1.

NOTE

Do not use stop mode if you are using an external clock source because X_{IN} input must be restricted internally to V_{SS} to reduce current leakage.

Idle or stop mode is terminated either by a RESET, or by an interrupt, which are enabled by the corresponding interrupt enable flag, IEx. When power-down mode is terminated by RESET input, a normal reset operation is executed. (Assuming that both the interrupt enable flag and the interrupt request flag are set to "1", power-down mode is released immediately upon entering power-down mode.

When an interrupt is used to release power-down mode, the operation differs depending on the value of the interrupt master enable flag (IME):

- If the IME flag = "0", program execution is started immediately after the instruction which issues the request to enter power-down mode. The interrupt request flag remains set to logic one.
- If the IME flag = "1", two instructions are executed after the power-down mode release. Then, the vectored interrupt is initiated. However, when the release signal is caused by INTK, the operation is identical to the IME = 0 condition. That is, a vector interrupt is not generated.

Table 8-1. Hardware Operation During Power-Down Modes

Operation	Stop Mode (STOP)	Idle Mode (IDLE)
Clock oscillator	System clock oscillation stops	CPU clock oscillation stops (system clock oscillation continues)
Basic timer	Basic timer stops	Basic timer operates (with IRQB set at each reference interval)
Timer/counter 0	Operates only if TCL0 is selected as the counter clock	Timer/counter 0 operates
External interrupts	INT0, INT1 and INTK are acknowledged	INT0, INT1 and INTK are acknowledged
CPU	All CPU operations are disabled	All CPU operations are disabled
Power-down mode release signal	Interrupt request signals which are enabled by an interrupt enable flag or by RESET input	Interrupt request signals which are enabled by an interrupt enable flag or by RESET input
Buzzer	Buzzer operation is stopped	Buzzer operates
D/A Converter	D/A Converter retains the last analog value	D/A Converter retains the last analog value

IDLE MODE TIMING DIAGRAMS

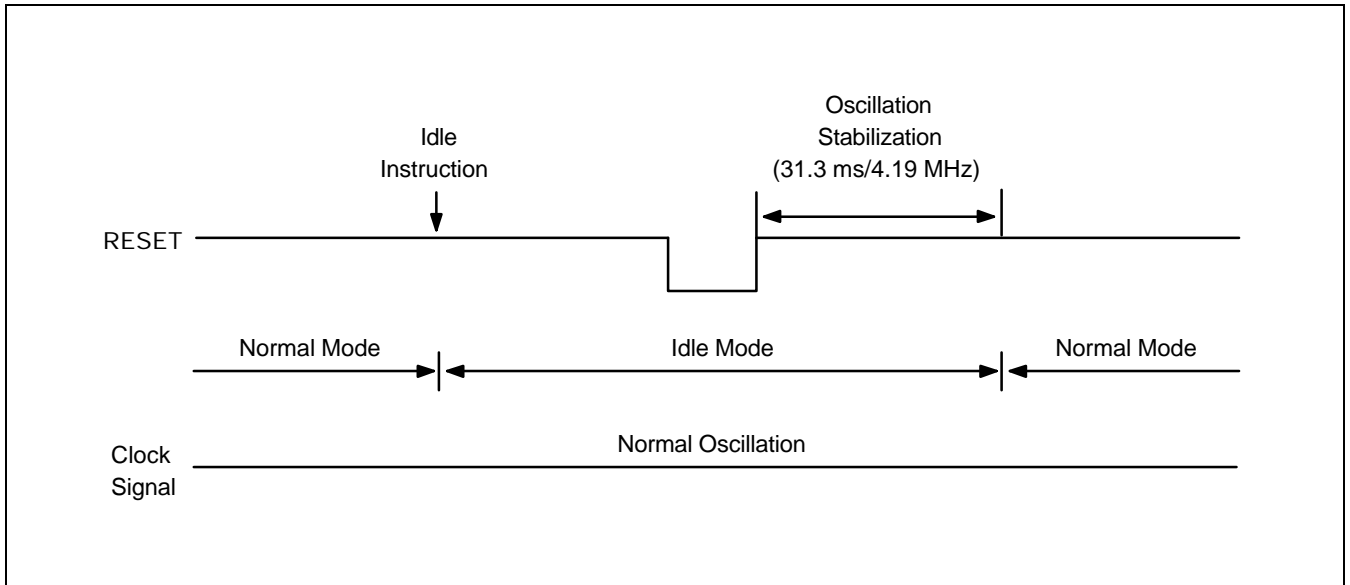


Figure 8-1. Timing When Idle Mode is Released by RESET

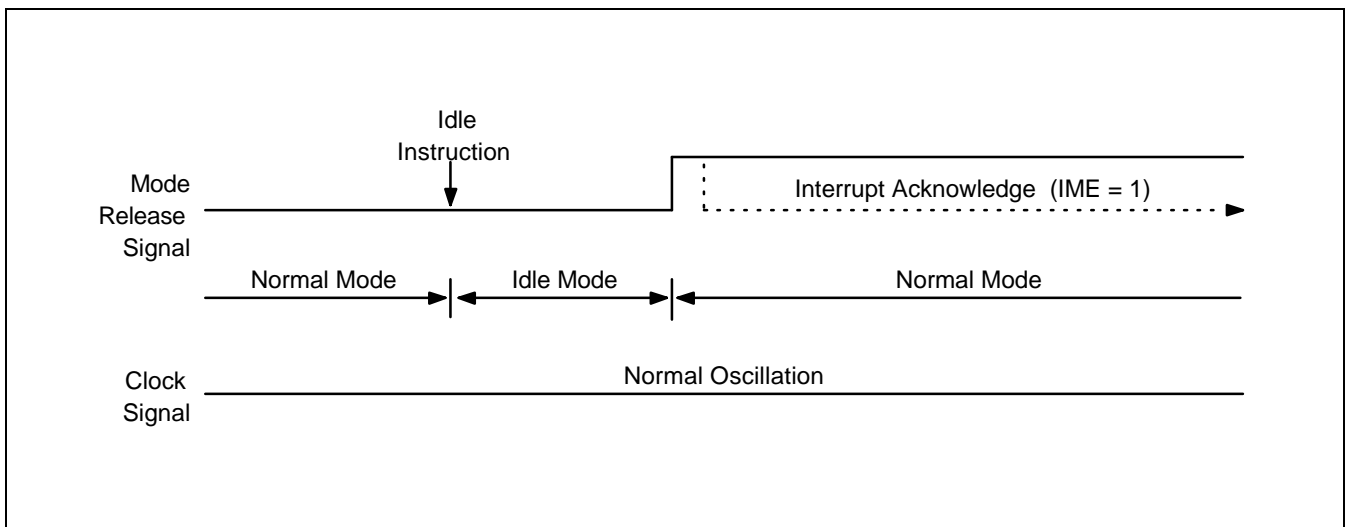


Figure 8-2. Timing When Idle Mode is Released by an Interrupt

STOP MODE TIMING DIAGRAMS

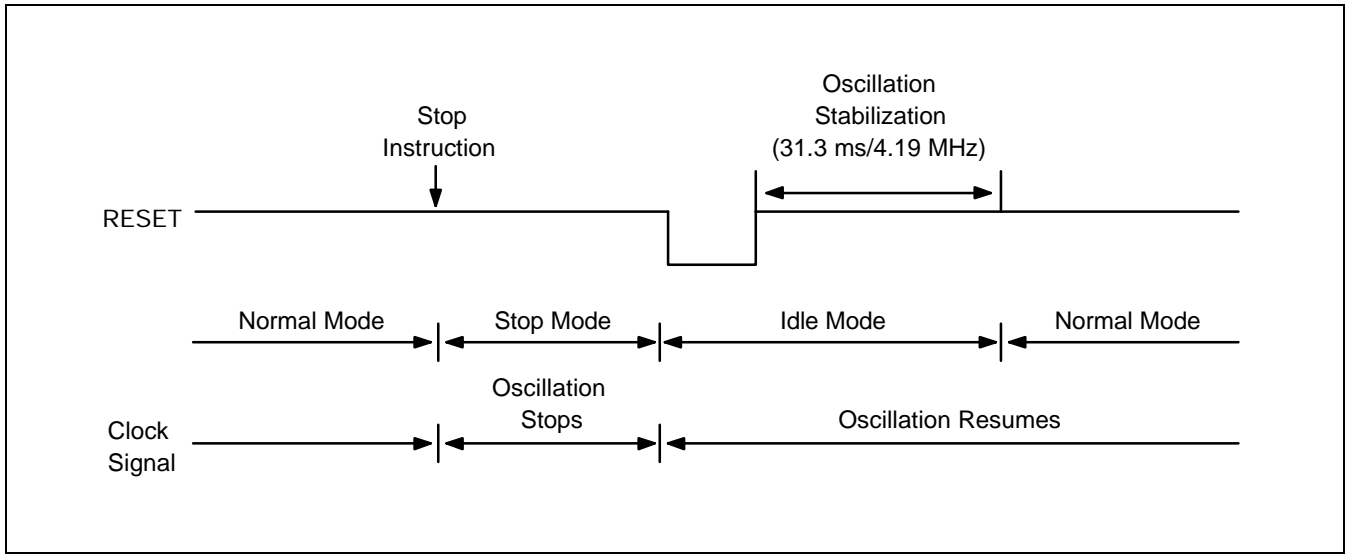


Figure 8-3. Timing When Stop Mode is Released by RESET

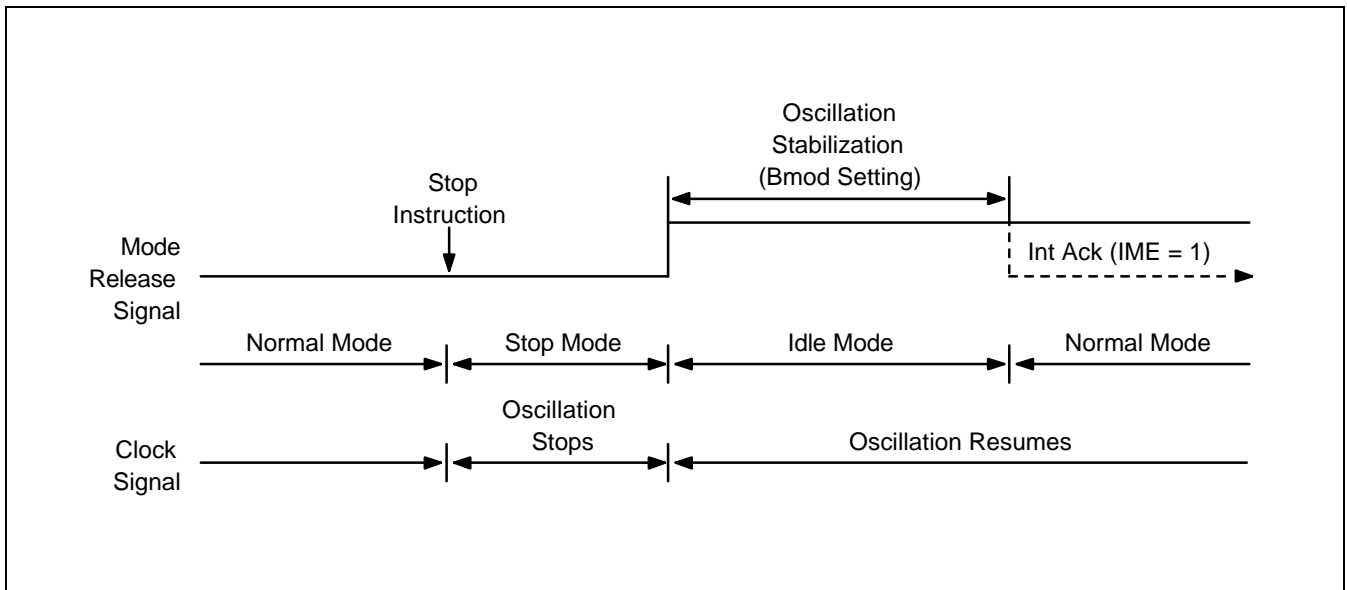


Figure 8-4. Timing When Stop Mode is Release by an Interrupt

I/O PORT PIN CONFIGURATION FOR POWER-DOWN

The following method describes how to configure I/O port pins to reduce power consumption during power-down modes (stop, idle):

Condition 1: If the microcontroller is not configured to an external device:

1. Connect unused port pins according to the information in Table 8-2.
2. Disable all pull-up resistors for output pins by making appropriate modifications to the pull-up resistor mode register, PUMOD. Reason: If output goes low when the pull-up resistor is enabled, there may be unexpected surges of current through the pull-up.
3. Disable pull-up resistors for input pins configured to V_{DD} or V_{SS} levels in order to check the current input option. Reason: If the input level of a port pin is set to V_{SS} when a pull-up resistor is enabled, it will draw an unnecessarily large current.

Condition 2: If the microcontroller is configured to an external device and the external device's V_{DD} source is turned off in power-down mode.

1. Connect unused port pins according to the information in Table 8-2.
2. Disable the pull-up resistors of output pins by making appropriate modifications to the pull-up resistor mode register, PUMOD. Reason: If output goes low when the pull-up resistor is enabled, there may be unexpected surges of current through the pull-up.
3. Disable pull-up resistors for input pins configured to V_{DD} or V_{SS} levels in order to check the current input option. Reason: If the input level of a port pin is set to V_{SS} when a pull-up resistor is enabled, it will draw an unnecessarily large current.
4. Disable the pull-up resistors of input pins connected to the external device by making necessary modifications to the PUMOD register.
5. Configure the output pins that are connected to the external device to low level. Reason: When the external device's V_{DD} source is turned off, and if the microcontroller's output pins are set to high level, $V_{DD}-0.7$ V is supplied to the V_{DD} of the external device through its input pin. This causes the device to operate at the level $V_{DD}-0.7$ V. In this case, total current consumption would not be reduced.
6. Determine the correct output pin state necessary to block current pass in accordance with the external transistors (PNP, NPN).

RECOMMENDED CONNECTIONS FOR UNUSED PINS

To reduce overall power consumption, please configure unused pins according to the guidelines described in Table 8-2.

Table 8-2. Unused Pin Connections for Reduced Power Consumption

Pin/Share Pin Names	Recommended Connection
P0.0/INT0 P0.1/INT1 P0.2/KS0 P0.3/KS1	Input mode: Connect to V_{DD} Output mode: Do not connect
P1.0/TCL0 P1.1/TCLO0 P1.2/CLO P1.3/BUZ	
P2.0	
P4.0–P4.3 P5.0–P5.3	
DAO	

9

RESET

OVERVIEW

When a RESET signal is input during the normal operation or power-down mode, a reset operation is initiated and the CPU enters idle mode. Then, when the standard oscillation stabilization interval of 31.3 ms at 4.19 MHz has elapsed, the normal system operation resumes.

Regardless of when the RESET occurs — during normal operating mode or power-down mode — the effect on most hardware register values is almost identical. The exceptions are as follows:

- Carry flag
- Data memory values
- General-purpose registers E, A, L, H, X, W, Z, and Y

If a RESET occurs during idle or stop mode, the current values in these registers are retained. Otherwise, their values are undefined.

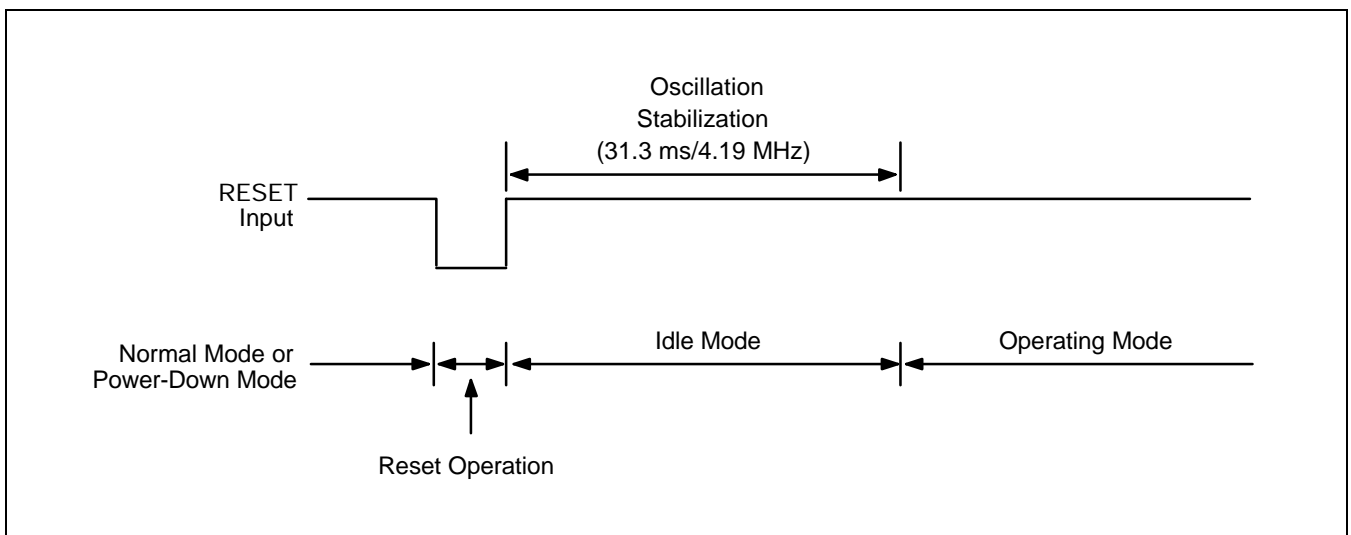


Figure 9-1. Timing for Oscillation Stabilization after RESET

HARDWARE REGISTER VALUES AFTER RESET

Table 9-1 gives you detailed information about hardware register values after a RESET occurs during power-down or normal operation mode.

Table 9-1. Hardware Register Values after RESET

Hardware Component or Subcomponent	If a RESET Occurs During Power-Down Mode	If a RESET Occurs During Normal Operation
Program counter (PC)	Lower three bits of address 0000H are transferred to PC11–8, and the contents of 0001H to PC7–0.	Lower three bits of address 0000H are transferred to PC11–8, and the contents of 0001H to PC7–0.
Program Status Word (PSW):		
Carry flag (C)	Retained	Undefined
Skip flag (SC0–SC2)	0	0
Interrupt status flags (IS0, IS1)	0	0
Bank enable flags (EMB, ERB)	Bit 6 of address 0000H in program memory is transferred to the ERB flag, and bit 7 of the address to the EMB flag.	Bit 6 of address 0000H in program memory is transferred to the ERB flag, and bit 7 of the address to the EMB flag.
Stack pointer (SP)	Undefined	Undefined
Data Memory (RAM):		
General registers E, A, L, H, X, W, Z, Y	Values retained	Undefined
General-purpose registers	Values retained (note)	Undefined
Bank selection registers (SMB, SRB)	0, 0	0, 0
BSC register (BSC0–BSC3)	0	0
Clocks:		
Power control register (PCON)	0	0
Clock output mode register (CLMOD)	0	0
Interrupts:		
Interrupt request flags (IRQx)	0	0
Interrupt enable flags (IEx)	0	0
Interrupt priority flag (IPR)	0	0
Interrupt master enable flag (IME)	0	0
INT0 mode register (IMOD0)	0	0
INT1 mode register (IMOD1)	0	0
INTK mode register (IMODK)	0	0

Table 9-1. Hardware Register Values after RESET (Continued)

Hardware Component or Subcomponent	If a RESET Occurs During Power-Down Mode	If a RESET Occurs During Normal Operation
I/O Ports:		
Output buffers	Off	Off
Output latches	0	0
Port mode flags (PM)	0	0
Pull-up resistor mode reg (PUMOD)	0	0
Port open-drain enable register (PNE)	0	0
Watch-dog Timer:		
WDT mode register (WDMOD)	A5H	A5H
WDT clear flag (WDTCF)	0	0
Basic Timer:		
Count register (BCNT)	Undefined	Undefined
Mode register (BMOD)	0	0
Timer/Counter 0:		
Count registers (TCNT0)	0	0
Reference registers (TREF0)	FFH	FFH
Mode registers (TMOD0)	0	0
Output enable flags (TOE0)	0	0
Buzzer:		
Buzzer mode register (BUZMOD)	0	0
D/A Converter:		
D/A converter mode register (DAMOD)	0	0
D/A converter DATA register (DADATA)	Undefined	Undefined

NOTE: The value of the 0F8H–0FDH are not retained when a RESET signal is input.

10 I/O PORTS

OVERVIEW

The S3C7544 has five I/O ports. Pin addresses for all I/O ports are mapped to the locations FF0H–FF5H in bank 15 of the RAM. The contents of I/O port pin latches can be read, written, or tested at the corresponding address using bit manipulation instructions.

There are a total of 17 configurable I/O pins, including 8 high-current I/O pins for a maximum number of 17 I/O pins.

Port Mode Flags

Port mode flags (PM) are used to configure I/O ports 0 and 1 (port mode group 1), port 2 (port mode group 2), and ports 4 and 5 (port mode group 3) to input or output mode by setting or clearing the corresponding I/O buffer. PM flags are stored in three 8-bit registers in the RAM area FE8H–FEDH, and they are addressable by 8-bit write instructions only.

Pull-up Resistors

Pull-up resistors are assignable to input pins of ports 0, 1, 2, 4, and 5. When a configurable I/O port pin serves as an output pin, its assigned pull-up resistor is automatically disabled, even though the pin's pull-up resistor is enabled by a corresponding bit setting in the pull-up resistor mode register (PUMOD).

PUMOD Control Register

The pull-up mode register (PUMOD) is an 8-bit register used to assign internal pull-up resistors by software to specific I/O ports.

When a configurable I/O port pin is used as an output pin, its assigned pull-up resistor is automatically disabled, even though the pin's pull-up is enabled by a corresponding PUMOD bit setting.

PUMOD is mapped to the RAM address FDCH–FDDH and is addressable by 8-bit write instructions only. A RESET clears PUMOD register values to logic zero, automatically disconnecting all software-assignable port pull-up resistors.

Table 10-1. I/O Port Overview

Port	I/O	Pins	Pin Names	Address	Function Description
0	I/O	4	P0.0–P0.3	FF0H	4-bit I/O port. 1- or 4-bit read/write and test is possible. Pull-up resistors are assignable to input pins by software and are automatically disabled for output pins. Pins are individually configurable as input or output.
1	I/O	4	P1.0–P1.3	FF1H	Same as port 0.
2	I/O	1	P2.0	FF2H	1-bit I/O port. 1- or 4-bit read/write and test is possible. Pull-up resistor is assignable to input pin by software and is automatically disabled for output pin.
4, 5	I/O	8	P4.0–P4.3 P5.0–P5.3	FF4H FF5H	4-bit I/O ports. 1-, 4-, and 8-bit read/write/test is possible. Pins are individually configurable as input or output. Pull-up resistors are assignable to input pins by software and are automatically disabled for output pins. The N-channel open drain or push-pull output can be selected by software (1-bit unit)

Table 10-2. I/O Port Pin Status During Instruction Execution

Instruction Type	Example	Input Mode Status	Output Mode Status
1-bit test 1-bit input 4-bit input 8-bit input	BTST P0.1 LDB C,P1.3 LD A,P5 LD EA,P4	Input or test data at each pin	Input or test data at output latch
1-bit output	BITR P1.0	Output latch contents undefined	Output pin status is modified
4-bit output 8-bit output	LD P2,A LD P4,EA	Transfer accumulator data to the output latch	Transfer accumulator data to the output pin

PORT MODE FLAGS (PM FLAGS)

Port mode flags (PM) are used to configure I/O ports 0–2, 4 and 5 to input or output mode by setting or clearing the corresponding I/O buffer. PM flags are stored in three 8-bit registers in the RAM area FE8H–FEDH, and are addressable by 8-bit write instructions only.

For convenient program reference, PM flags are organized into three groups — PMG1, PMG2, and PMG3, as shown in Table 10-3.

Table 10-3. Port Mode Groups and Corresponding I/O Ports

Port Mode Group ID	Corresponding I/O Ports	Port Mode Group Address
PMG1	Ports 0 and 1	FE8H–FE9H
PMG2	Port 2	FEAH–FEBH
PMG3	Ports 4 and 5	FECH–FEDH

When a PM flag is "0", the port is set to input mode; when it is "1", the port is enabled for output. A RESET clears all port mode flags to logic zero, automatically configuring the corresponding I/O ports to input mode.

Table 10-4. Port Mode Flag Map

PM Group ID	Address	Bit 3	Bit 2	Bit 1	Bit 0
PMG1	FE8H	PM0.3	PM0.2	PM0.1	PM0.0
	FE9H	PM1.3	PM1.2	PM1.1	PM1.0
PMG2	FEAH	"0"	"0"	"0"	PM2.0
	FEBH	"0"	"0"	"0"	"0"
PMG3	FECH	PM4.3	PM4.2	PM4.1	PM4.0
	FEDH	PM5.3	PM5.2	PM5.1	PM5.0

NOTE: If bit = "0", the corresponding I/O pin is set to input mode. If bit = "1", the pin is set to output mode. All flags are cleared to "0" after a RESET.

PROGRAMMING TIP — Configuring I/O Ports as Input or Output

Configure P0.0 and P3.0 as an output port and other ports as input ports:

```

BITS      EMB
SMB      15
LD      EA,#11H
LD      PMG1,EA      ; P0.0 and P1.0 ← Output
LD      EA,#00H
LD      PMG2,EA      ; P2 ← Input
LD      EA,#00H
LD      PMG3,EA      ; P4, P5 ← Input

```

PULL-UP RESISTOR MODE REGISTER (PUMOD)

The pull-up resistor mode register (PUMOD) is an 8-bit register used to assign internal pull-up resistors by software to specific I/O ports. When a configurable I/O port pin is used as an output pin, its assigned pull-up resistor is automatically disabled, even though the pin's pull-up is enabled by a corresponding PUMOD bit setting.

PUMOD is mapped to the RAM address FDCH–FDDH and is addressable by 8-bit write instructions only. A RESET clears PUMOD register values to logic zero, automatically disconnecting all software-assignable port pull-up resistors.

Table 10-5. Pull-up Resistor Mode Register (PUMOD) Organization

Address	Bit 3	Bit 2	Bit 1	Bit 0
FDCH	"0"	PUR2	PUR1	PUR0
FDDH	"0"	"0"	PUR5	PUR4

NOTE: When bit = "1", a pull-up resistor is assigned to the corresponding I/O port: PUMOD.0 for port 0, PUMOD.1 for port 1, and so on.

N-CHANNEL OPEN-DRAIN ENABLE REGISTER (PNE)

The N-channel, open-drain mode register, PNE, is used to configure ports 4 and 5 to n-channel, open-drain mode or as push-pull outputs.

When a bit in the PNE register is set to "1", the corresponding output pin is configured to n-channel open-drain; when set to "0", the output pin is configured to push-pull; PNE4.3 for P4.3, PNE4.2 for P4.2, PNE4.1 for P4.1, PNE4.0 for P4.0, PNE5.3 for P5.3, PNE5.2 for P5.2, PNE5.1 for P5.1 and PNE5.0 for P5.0.

FDAH	PNE4.3	PNE4.2	PNE4.1	PNE4.0	PNE4
FDBH	PNE5.3	PNE5.2	PNE5.1	PNE5.0	PNE5

 PROGRAMMING TIP — Enabling and Disabling I/O Port Pull-up Resistors

P5 pull-up resistor is enabled; P0, P1, and P4 pull-up resistors are disabled as follows:

```

BITS      EMB
SMB       15
LD        EA,#20H
LD        PUMOD,EA      ; P5 enable

```

PORT 0 CIRCUIT DIAGRAM

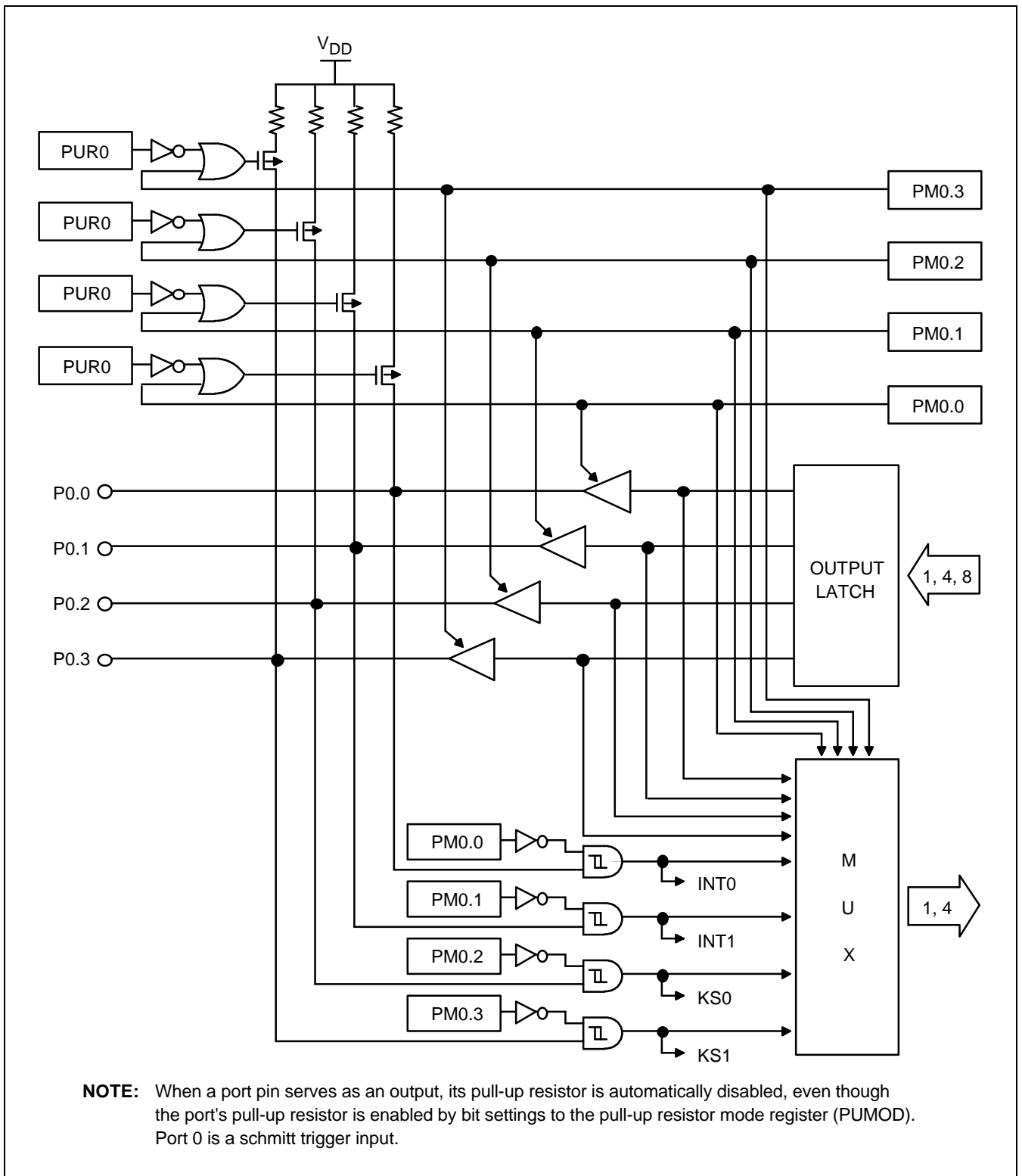


Figure 10-1. I/O Port 0 Circuit Diagram

PORT 1 CIRCUIT DIAGRAM

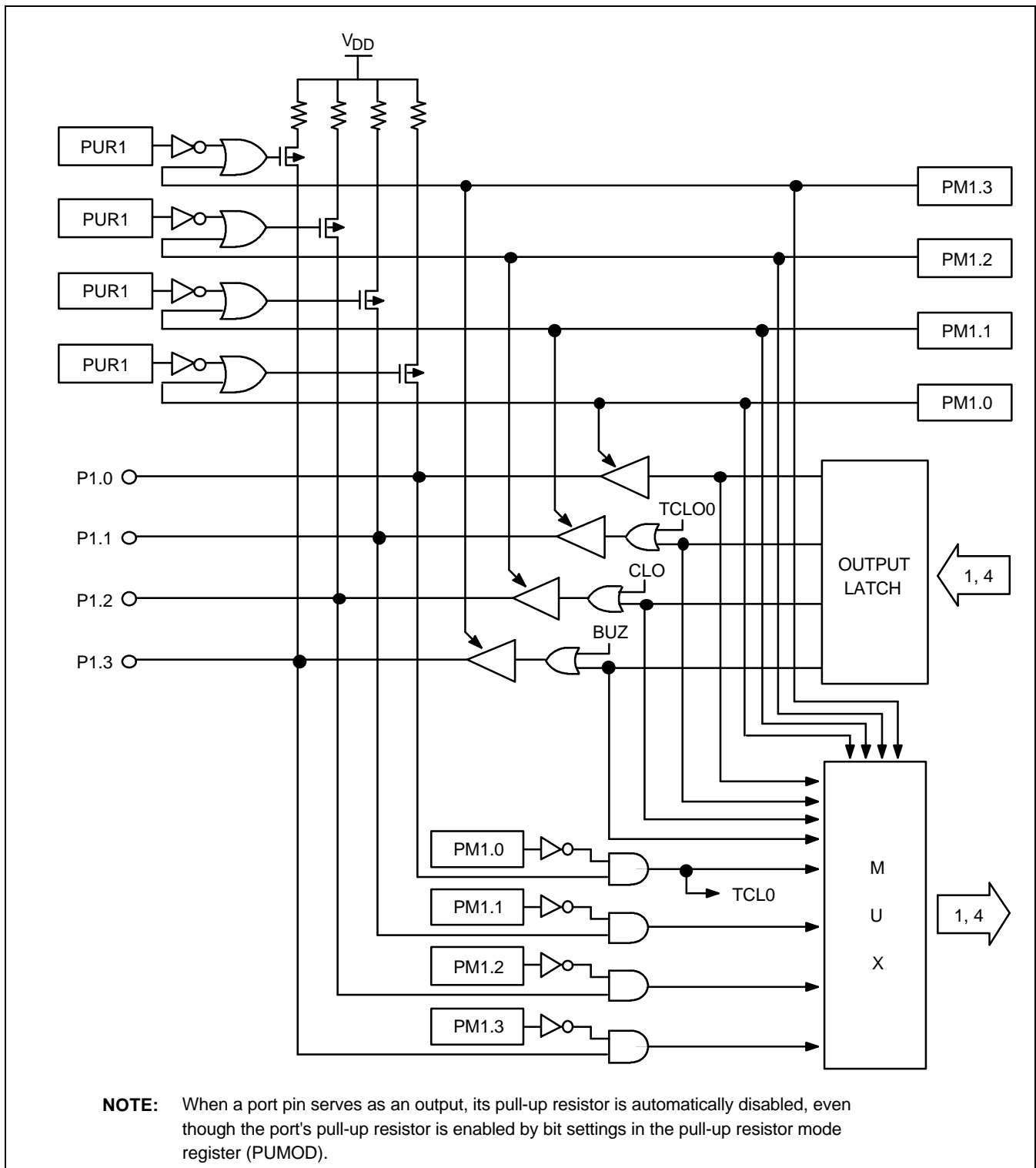


Figure 10-2. Input Port 1 Circuit Diagram

PORT 2 CIRCUIT DIAGRAM

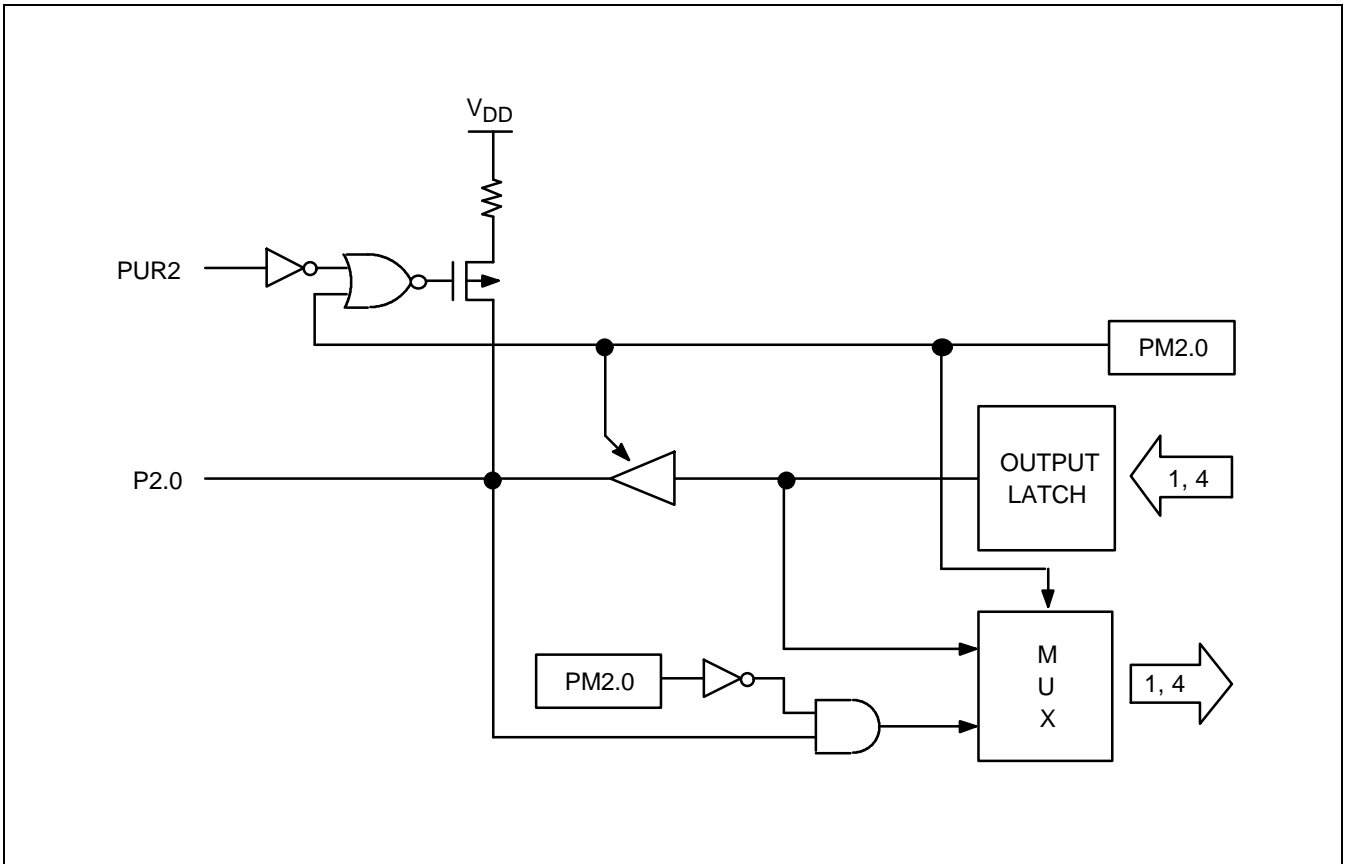


Figure 10-3. Port 2 Circuit Diagram

PORTS 4 AND 5 CIRCUIT DIAGRAM

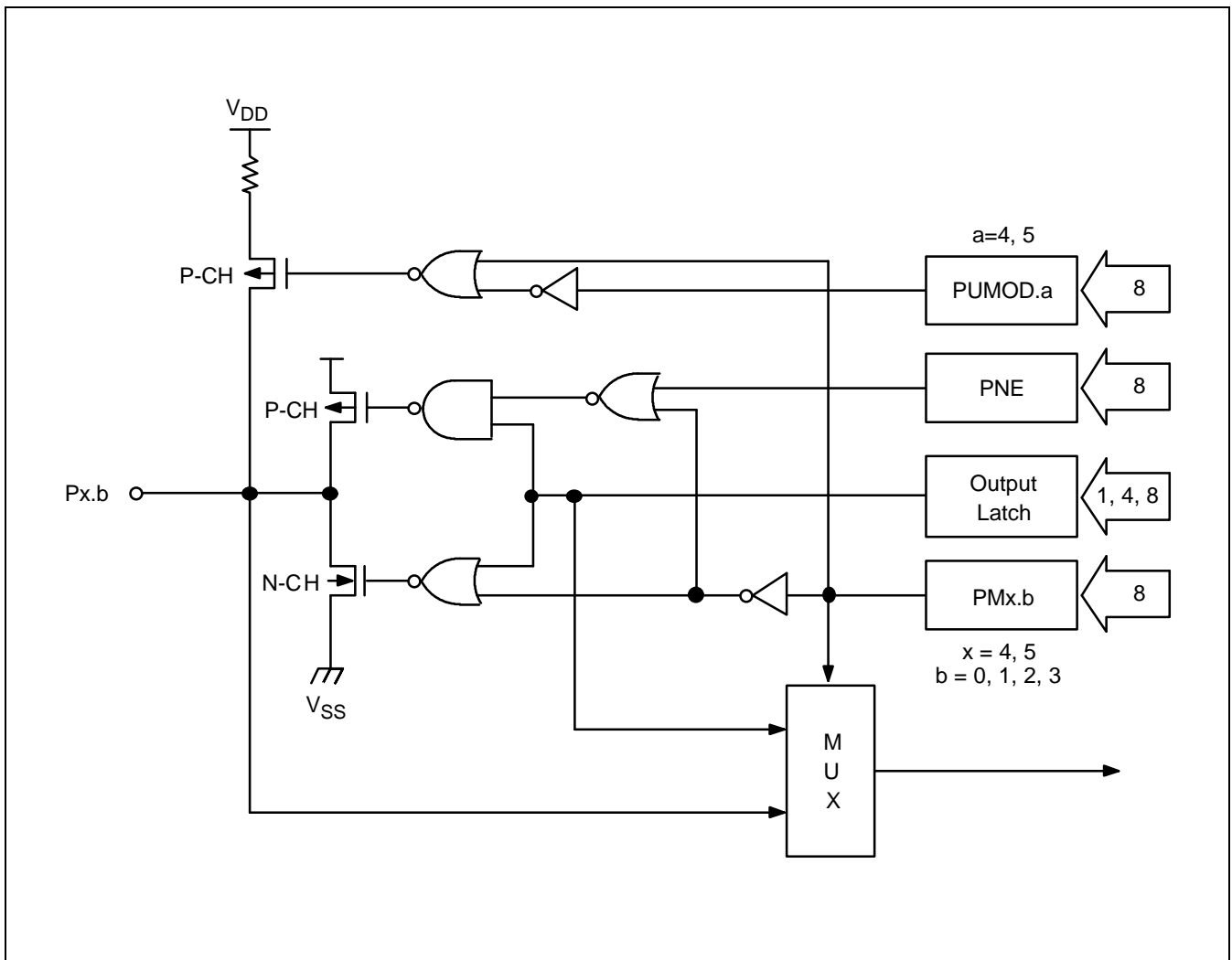


Figure 10-4. Circuit Diagram for Ports 4 and 5

11

TIMERS and TIMER/COUNTER

OVERVIEW

There are two timer and timer/counter function modules:

- 8-bit basic timer (BT)
- 8-bit timer/counter 0 (TC0)

The 8-bit basic timer (BT) is the microcontroller's main interval timer. It generates an interrupt request at a fixed time interval by making appropriate modification to the mode register.

The basic timer also functions as a 'watchdog' timer and is used to determine clock oscillation stabilization time when stop mode is released by an interrupt or a RESET.

The 8-bit timer/counter 0 (TC0) is a programmable timer/counter that is used primarily for event counting and clock frequency modification and output.

BASIC TIMER (BT)

OVERVIEW

The 8-bit basic timer (BT) has six functional components:

- Clock selector logic
- 4-bit basic timer mode register (BMOD)
- 8-bit basic timer counter register (BCNT)
- 8-bit Watchdog timer mode register (WDMOD)
- Watchdog timer counter clear flag (WDTCF)
- 3-bit watchdog timer clear flag (WDCNT)

The basic timer generates interrupt requests at precise intervals, based on the frequency of the system clock. You can use the basic timer as a "watchdog" timer for monitoring system events or use BT output to stabilize clock oscillation when stop mode is released by an interrupt or after a RESET.

Use the basic timer mode register, BMOD, to turn the BT on and off, to select input clock frequency, and to control interrupt or stabilization intervals.

Interval Timer Function

The measurement of elapsed time intervals is the primary function the basic timer's. The standard interval is 256 BT clock pulses.

To restart the basic timer, set bit 3 of the mode register BMOD to logic one. The input clock frequency and the interrupt and stabilization interval are selected by loading appropriate bit values to BMOD.2–BMOD.0.

The 8-bit counter register, BCNT, is incremented each time a clock signal that corresponds to the frequency selected by BMOD is detected. BCNT continues incrementing as it counts BT clocks until an overflow occurs. An overflow causes the BT interrupt request flag (IRQB) to be set to logic one to signal that the designated time interval has elapsed. An interrupt request is then generated, BCNT is cleared to logic zero, and counting continues from 00H.

Watchdog Timer Function

The basic timer can also be used as a "watchdog" timer that detecting any inadvertent program loop, that is, a system or program operation error. For this purpose, instructions that clear the watchdog timer (*BITS WDTCF*) should be executed at proper points in a program within a given period. If such an instruction is not executed within the period and the watchdog timer overflows, a reset signal is generated and the system is restarted with a reset. The operation of the watchdog timer is as follows:

- Write some values (except #5AH) to Watchdog Timer Mode register, WDMOD.
- If WDCNT overflows, a system reset is generated.

Oscillation Stabilization Interval Control

Bits 2–0 of the BMOD register are used to select the input clock frequency for the basic timer. This setting also determines the time interval (also referred to as 'wait time') required to stabilize clock signal oscillation when power-down mode is released by an interrupt. When a RESET signal is generated, the standard stabilization interval for system clock oscillation after a RESET is 31.3 ms at 4.19 MHz.

Table 11-1. Basic Timer Register Overview

Register Name	Type	Description	Size	RAM Address	Addressing Mode	Reset Value
BMOD	Control	Controls the clock frequency (mode) of the basic timer and the oscillation stabilization interval after power-down a mode release or RESET	4-bit	F85H	4-bit write-only; BMOD.3 is also 1-bit writeable	"0"
BCNT	Counter	Counts clock pulses matching the BMOD frequency setting	8-bit	F86H– F87H	8-bit read-only	U (note)
WDMOD	Control	Controls watchdog timer operation.	8-bit	F98H–F99H	8-bit write-only	A5H
WDTCF	Control	Clear the watchdog timer's counter.	1-bit	F9AH.3	1-bit write-only	"0"

NOTE: 'U' means the value is undetermined after a RESET.

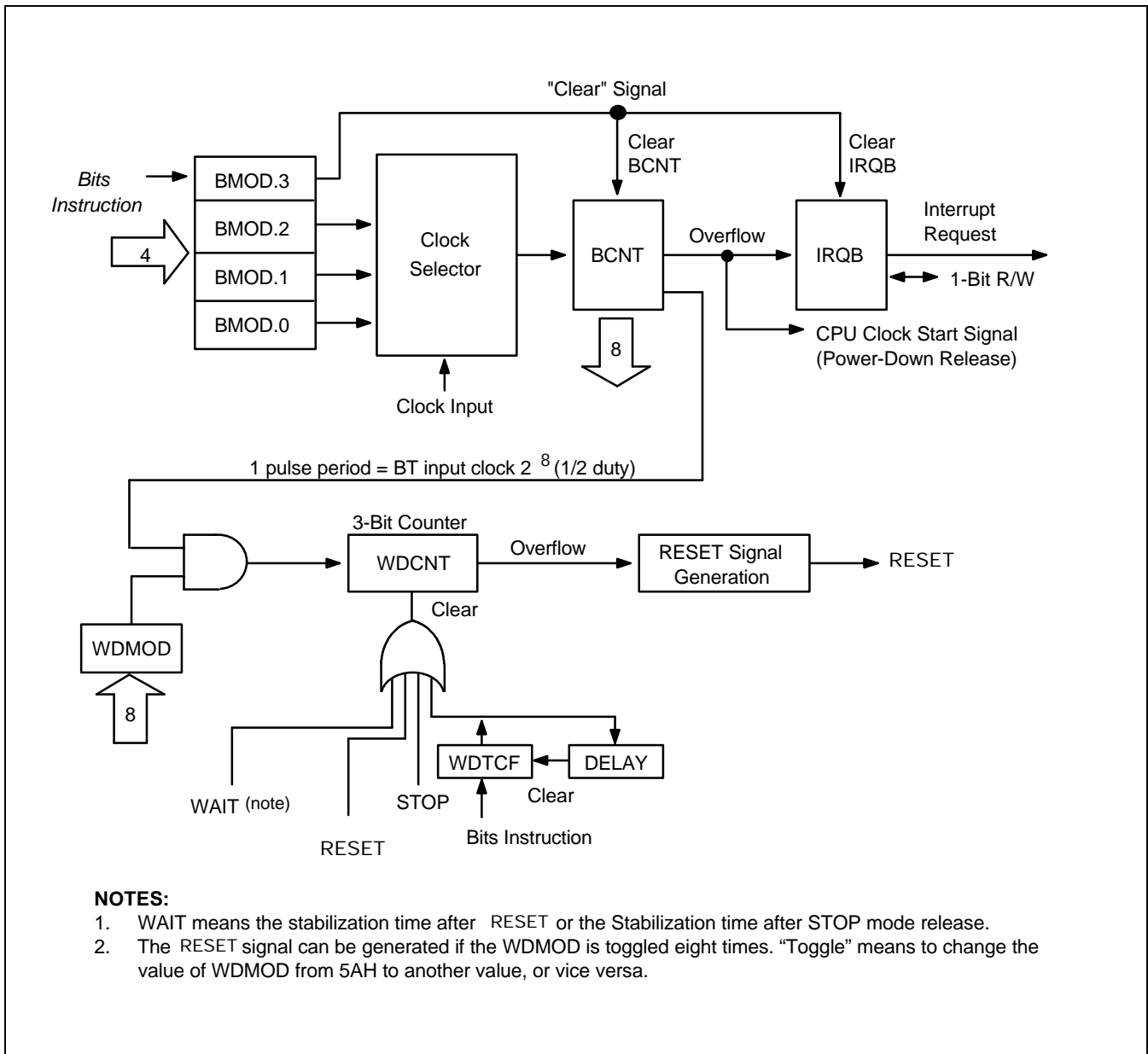


Figure 11-1. Basic Timer Circuit Diagram

BASIC TIMER MODE REGISTER (BMOD)

The basic timer mode register, BMOD, is a 4-bit write-only register located at the RAM address F85H. Bit 3, the basic timer start control bit, is also 1-bit addressable. All BMOD values are set to logic zero after a RESET and interrupt request signal generation is set to the longest interval. (BT counter operation cannot be stopped.) BMOD settings have the following effects:

- Restart the basic timer,
- Control the frequency of clock signal input to the basic timer,
- Determine time interval required for clock oscillation to stabilize after the release of stop mode by an interrupt.

By loading different values into the BMOD register, you can dynamically modify the basic timer clock frequency during program execution. Four BT frequencies, ranging from $fx/2^{12}$ (1.02 kHz) to $fx/2^5$ (131 kHz), are selectable. Since BMOD's reset value is logic zero, the default clock frequency setting is $fx/2^{12}$. (kHz frequencies assume a system clock (fx) frequency of 4.19 MHz.)

The most significant bit of the BMOD register, BMOD.3, is used to start the basic timer again. When BMOD.3 is set to logic one (enabled) by a 1-bit write instruction, the contents of the BT counter register (BCNT) and the BT interrupt request flag (IRQB) are both cleared to logic zero, and timer operation is restarted.

The combination of bit settings in the remaining three registers — BMOD.2, BMOD.1, and BMOD.0 — determines the clock input frequency and oscillation stabilization interval.

Table 11-2. Basic Timer Mode Register (BMOD) Organization

BMOD.3			Basic Timer Enable/Disable Control Bit	
1			Start basic timer; clear IRQB, BCNT, and BMOD.3 to "0"	

BMOD.2	BMOD.1	BMOD.0	Basic Timer Input Clock	Oscillation Stabilization
0	0	0	$fx/2^{12}$ (1.02 kHz)	$2^{20}/fx$ (250 ms)
0	1	1	$fx/2^9$ (8.18 kHz)	$2^{17}/fx$ (31.3 ms)
1	0	1	$fx/2^7$ (32.7 kHz)	$2^{15}/fx$ (7.82 ms)
1	1	1	$fx/2^5$ (131 kHz)	$2^{13}/fx$ (1.95 ms)

NOTES:

1. Clock frequencies and stabilization intervals assume a system oscillator clock frequency (fx) of 4.19 MHz.
2. fx = system clock frequency.
3. Oscillation stabilization time is the time required to stabilize clock signal oscillation after stop mode is released.
4. The standard stabilization time for system clock oscillation after a RESET is 31.3 ms at 4.19 MHz.

BASIC TIMER COUNTER (BCNT)

BCNT is an 8-bit counter register for the basic timer. It is mapped to the RAM addresses F86H–F87H and can be addressed by 8-bit read instructions.

A RESET leaves the BCNT register value undetermined. BCNT is automatically cleared to logic zero whenever the BMOD register control bit (BMOD.3) is set to "1" to restart the basic timer. It is incremented each time a clock pulse of the frequency determined by the current BMOD bit settings is detected.

When BCNT has incremented to hexadecimal 'FFH' (256 clock pulses), it is cleared to '00H' and an overflow is generated. The overflow causes the interrupt request flag, IRQB, to be set to logic one. When the interrupt request is generated, BCNT immediately resumes counting incoming clock signals.

NOTE

Always execute a BCNT read operation twice to eliminate the possibility of reading unstable data while the counter is incrementing. If, after two consecutive reads, the BCNT values match, you can select the latter value as valid data. Until the results of the consecutive reads match, however, the read operation must be repeated until the validation condition is met.

BASIC TIMER OPERATION SEQUENCE

The basic timer's sequence of operations may be summarized as follows:

1. Set bit BMOD.3 to logic one to restart basic timer operation
2. BCNT is incremented by one after each clock pulse corresponding to BMOD selection
3. BCNT overflows if $BCNT \geq 255$ (FFH)
4. When an overflow occurs, the IRQB flag is set to logic one by hardware
5. The interrupt request is generated
6. BCNT is automatically cleared to logic zero (BCNT = 00H)
7. BCNT resumes counting BT clock pulse

PROGRAMMING TIP — Using the Basic Timer

1. To read the basic timer count register (BCNT):

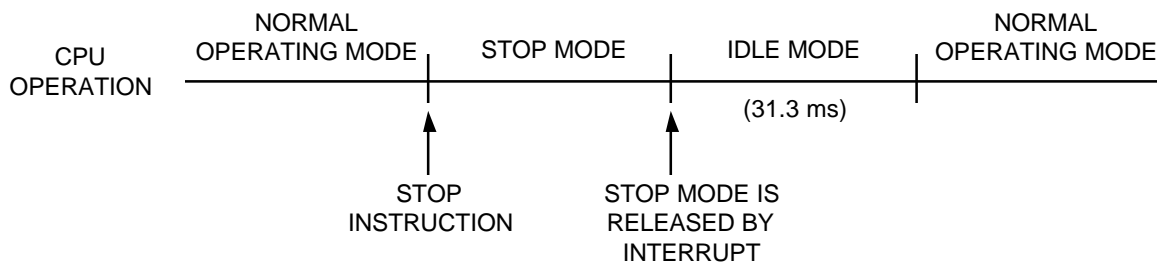
```

          BITS      EMB
          SMB      15
BCNTR    LD      EA,BCNT
          LD      YZ,EA
          LD      EA,BCNT
          CPSE    EA,YZ
          JR      BCNTR
  
```

2. When stop mode is released by an interrupt, set the oscillation stabilization interval to 31.3 ms (at 4.19 MHz):

```

          BITS      EMB
          SMB      15
          LD      A,#0BH
          LD      BMOD,A           ; Wait time is 31.3 ms
          STOP                    ; Set stop power-down mode
          NOP
          NOP
          NOP
  
```



3. To set the basic timer interrupt interval time to 1.95 ms (at 4.19 MHz):

```

          BITS      EMB
          SMB      15
          LD      A,#0FH
          LD      BMOD,A
          EI
          BITS      IEB           ; Basic timer interrupt enable flag is set to "1"
  
```

4. Clear BCNT and the IRQB flag and restart the basic timer:

```

          BITS      EMB
          SMB      15
          BITS      BMOD.3
  
```

WATCHDOG TIMER MODE REGISTER (WDMOD)

The watchdog timer mode register, WDMOD, is an 8-bit write-only register located at the RAM address F98H–F99H. WDMOD register controls enabling or disabling the watchdog timer function. WDMOD values are set to logic "A5H" after a RESET and this value enables the watchdog timer. The watchdog timer's period is set to the longest interval because a BT overflow signal is generated with the longest interval. (BT counter operation cannot be stopped.)

Table 11-3. Watchdog Timer Mode Control Register

WDMOD	Watchdog Timer Enable/Disable Control
5AH	Disable Watchdog timer function value
Any other Value	Enable Watchdog timer function

WATCHDOG TIMER COUNTER (WDCNT)

WDCNT is a 3-bit counter. WDCNT is automatically cleared to logic zero whenever the WDTCF register control bit (WDTCF) is set to "1" to restart WDCNT. Reset, stop, and wait signal also clear WDCNT to logic zero. WDCNT is incremented each time a clock pulse of the overflow frequency determined by the current BMOD bit settings is detected. When WDCNT has incremented to hexadecimal '07H' (8 BT overflow pulses), it is cleared to '00H' and an overflow is generated. The overflow causes a system reset. When an interrupt request is generated, BCNT immediately resumes counting incoming clock signals.

WATCHDOG TIMER'S COUNTER CLEAR FLAG (WDTCF)

WDTCF(F9AH.3) setting clears the WDT's counter to zero and restarts the WDT's counter.

Table 11-4. Watchdog Timer Interval Time

BMOD	BT Input Clock	WDCNT Input Clock	WDT Interval Time (1)
x000b	$f_{xx}/2^{12}$	$f_{xx}/(2^{12} \times 2^8)$	$(7 \text{ or } 8) \times (2^{12} \times 2^8) / f_{xx} = 1.75\text{--}2 \text{ sec}$
x011b	$f_{xx}/2^9$	$f_{xx}/(2^9 \times 2^8)$	$(7 \text{ or } 8) \times (2^9 \times 2^8) / f_{xx} = 218.7\text{--}250 \text{ ms}$
x101b	$f_{xx}/2^7$	$f_{xx}/(2^7 \times 2^8)$	$(7 \text{ or } 8) \times (2^7 \times 2^8) / f_{xx} = 54.6\text{--}62.5 \text{ ms}$
x111b	$f_{xx}/2^5$	$f_{xx}/(2^5 \times 2^8)$	$(7 \text{ or } 8) \times (2^5 \times 2^8) / f_{xx} = 13.6\text{--}15.6 \text{ ms}$

NOTES:

1. Clock frequencies assume a system oscillator clock frequency (f_{xx}) of 4.19MHz .
2. f_{xx} = system clock frequency.

8-BIT TIMER/COUNTER 0 (TC0)

Timer/counter 0 (TC0) is used to count system 'events' by identifying the transition (high-to-low or low-to-high) of incoming square wave signals. To indicate that an event has occurred, or that a specified time interval has elapsed, TC0 generates an interrupt request. Counting signal transitions and comparing the current counter value with the reference register value, TC0 can be used to measure specific time intervals.

TC0 has a reloadable counter that consists of two parts: an 8-bit reference register (TREF0) into which you write the counter reference value, and an 8-bit counter register (TCNT0) whose value is automatically incremented by counter logic.

An 8-bit mode register, TMOD0, is used to activate the timer/counter and select the basic clock frequency to be used for timer/counter operations. You can modify the basic frequency dynamically by loading new values into TMOD0 during program execution.

TC0 FUNCTION SUMMARY

8-bit programmable timer	Generates interrupts at specific time intervals based on the selected clock frequency.
External event counter	Counts various system "events" based on edge detection of external clock signals at the TC0 input pin, TCL0. To start the event counting operation, TMOD0.2 is set to "1" and TMOD0.6 is cleared to "0".
Arbitrary frequency output	Outputs selectable clock frequencies to the TC0 output pin, TCLO0.
External signal divider	Divides the frequency of an incoming external clock signal according to a modifiable reference value (TREF0), and outputs the modified frequency to the TCLO0 pin.

TC0 COMPONENT SUMMARY

Mode register (TMOD0)	Activates the timer/counter and selects the internal clock frequency or the external clock source at the TCLK pin.
Reference register (TREF0)	Stores the reference value for the desired number of clock pulses between interrupt requests.
Counter register (TCNT0)	Counts internal or external clock pulses based on the bit settings in TMOD0 and TREF0.
Clock selector circuit	Together with the mode register (TMOD0), lets you select one of four internal clock frequencies, or external clock frequency.
8-bit comparator	Determines when to generate an interrupt by comparing the current value of the counter register (TCNT0) with the reference value previously programmed into the reference register (TREF0).
Output latch (TOL0)	A TC0 interrupt request or clock pulse is stored output latch, which is connected to the TC0 output pin, TCLK0. When the contents of the TCNT0 and TREF0 registers coincide, the timer/counter interrupt request flag (IRQT0) is set to "1", the status of TOL0 is inverted, and an interrupt is generated.
Output enable flag (TOE0)	You must set this flag to logic one before the contents of the TOL0 latch can be output to TCLK0.
Interrupt request flag (IRQT0)	This flag is cleared when TC0 operation starts and the TC0 interrupt service routine is executed and is enabled whenever the counter value and reference value coincide.
Interrupt enable flag (IET0)	Must be set to logic one before the interrupt requests generated by timer/counter can be processed.

Table 11-5. TC0 Register Overview

Register Name	Type	Description	Size	RAM Address	Addressing Mode	Reset Value
TMOD0	Control	Controls TC0 restart (bit 2); clears and resumes counting operation (bit 3); sets input clock and clock frequency (bits 6–4)	8-bit	F90H–F91H	8-bit write-only; (TMOD0.3 is also 1-bit write-only)	"0"
TCNT0	Counter	Counts clock pulses matching the TMOD0 frequency setting	8-bit	F94H–F95H	8-bit read-only	"0"
TREF0	Reference	Stores reference value for the timer/counter 0 interval setting	8-bit	F96H–F97H	8-bit write-only	FFH
TOE0	Flag	Controls timer/counter 0 output to the TCLK0 pin	1-bit	F92H.2	1/4-bit write-only	"0"

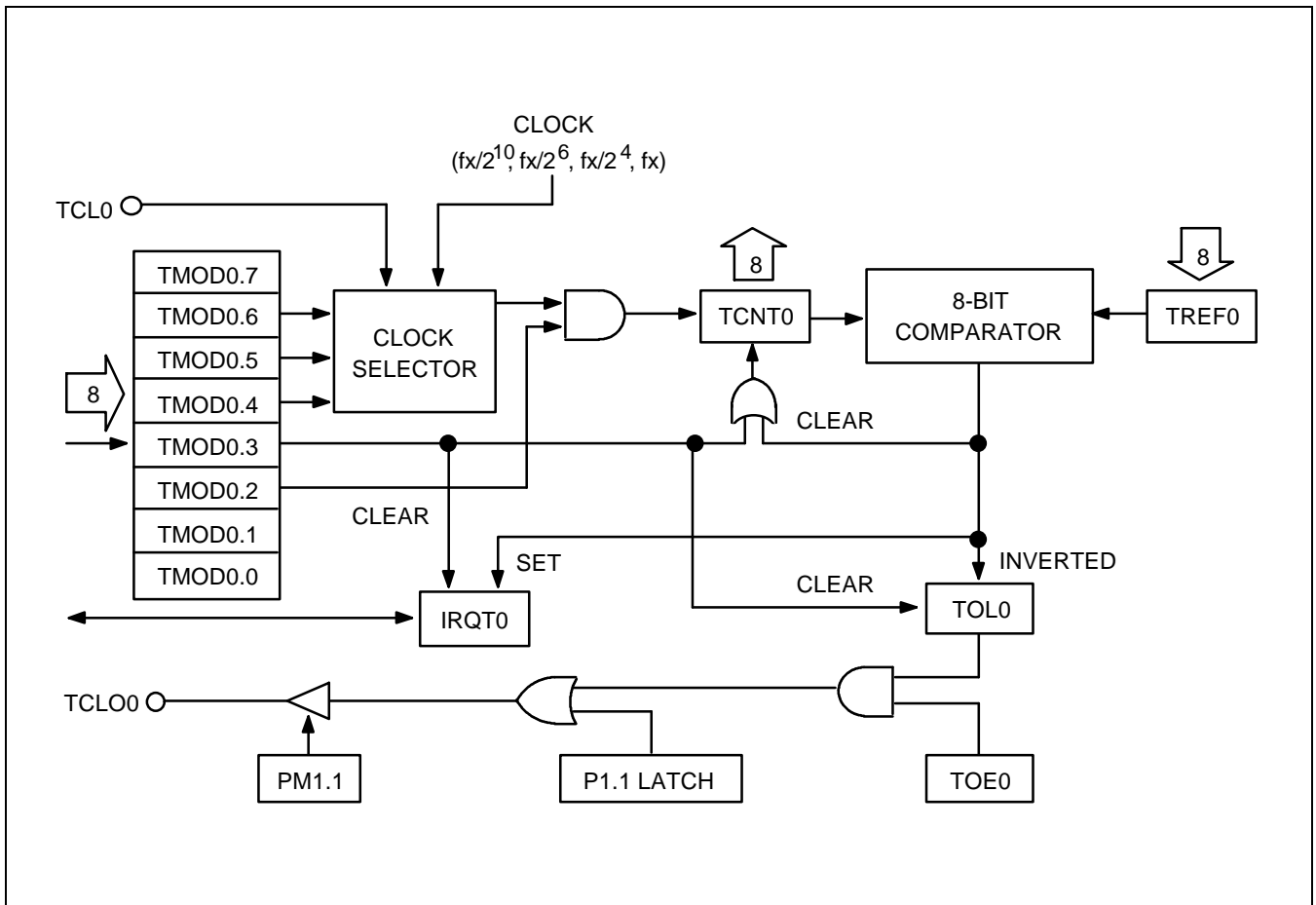


Figure 11-2. TC0 Circuit Diagram

TC0 ENABLE/DISABLE PROCEDURE

Enable Timer/Counter 0

- Set TMOD.2 to logic one (RAM address F90H.2)
- Set the TC0 interrupt enable flag IET0 to logic one (RAM address FBCH.1)
- Set TMOD0.3 to logic one (RAM address F90H.3)

TCNT0, IRQT0, and TOL0 are cleared to logic zero, and timer/counter operation starts.

Disable Timer/Counter

- Set TMOD0.2 to logic zero (RAM address F90H.2)

Clock signal input to the counter register TCNT0 is halted. The current TCNT0 value is retained and can be read if necessary.

TC0 PROGRAMMABLE TIMER/COUNTER FUNCTION

Timer/counter 0 can be programmed to generate interrupt requests at various intervals, based on the system clock frequency you select.

The 8-bit TC0 mode register, TMOD0, is used to activate the timer/counter and to select the clock frequency. The reference register, TREF0, stores your value for the number of clock pulses to be generated between interrupt requests. The counter register, TCNT0, counts the incoming clock pulses, which are compared to the TREF0 value as TCNT0 is incremented. When there is a match ($TREF0 = TCNT0$), an interrupt request is generated.

To program the timer/counter to generate interrupt requests at specific intervals, you should choose one of four internal clock frequencies (divisions of the system clock, f_x) and load your own counter reference value into the TREF0 register.

TCNT0 is incremented each time an internal counter pulse is detected with the reference clock frequency specified by TMOD0.4–TMOD0.6 settings. When the TC0 interrupt request flag (IRQT0) is set to logic one and the status of TOL0 is inverted, the interrupt is generated. The content of TCNT0 is then cleared to 00H, and TC0 continues counting.

The interrupt request mechanism for the programmable timer/counter consists of the TC0 interrupt enable flag IET0 and the TC0 interrupt request flag IRQT0.

TC0 OPERATION SEQUENCE

The general sequence of operations when using TC0 as a programmable timer/counter can be summarized as follows:

1. Set TMOD0.2 to "1" to enable TC0
2. Set TMOD0.6 to "1" to enable the system clock (f_x) input
3. Set TMOD0.5 and TMOD0.4 bits to desired internal frequency ($f_x/2^n$)
4. Load a value to TREF0 to specify the interval between interrupt requests
5. Set the TC0 interrupt enable flag (IET0) to "1"
6. Set TMOD0.3 bit to "1" to clear TCNT0, IRQT0, and TOL0, and start counting
7. TCNT0 increments with each internal clock pulse
8. When the comparator shows $TCNT0 = TREF0$, the IRQT0 flag is set to "1"
9. Output latch (TOL0) logic toggles high or low
10. Interrupt request is generated
11. TCNT0 is cleared to 00H and counting resumes
12. Programmable timer/counter operation continues until TMOD0.2 is cleared to "0".

TC0 EVENT COUNTER FUNCTION

Timer/counter 0 can be used to monitor or detect system 'events' by using the external clock input at the TCL0 pin (I/O port 1.0) as the counter source. The TC0 mode register is used to specify rising or falling edge detection for incoming clock signals. The counter register TCNT0 is incremented each time the selected state transition of the external clock signal occurs. To activate the TC0 event counter function,

- Set TMOD0.2 to "1" to enable TC0
- Clear TMOD0.6 to "0" to select the external clock source at the TCL0 pin
- Select TCL0 edge detection for rising or falling signal edges by loading appropriate values to TMOD0.5 and TMOD0.4.
- P1.0 must be set to input mode.

Table 11-6. TMOD0 Settings for TCL0 Edge Detection

TMOD0.5	TMOD0.4	TCL0 Edge Detection
0	0	Rising edges
0	1	Falling edges

With the exception of the different TMOD0.4–TMOD0.6 settings, the operation sequence for TC's event counter function is identical to its programmable counter/timer function.

TC0 CLOCK FREQUENCY OUTPUT

Using the timer/counter, you can output a modifiable clock frequency to the TC0 clock output pin, TCLO0. To select the clock frequency, you should load appropriate values to the TC0 mode register, TMOD0. The clock interval is determined by loading the desired reference value into the reference register TREF0. Then, to enable the output to the TCLO0 pin at I/O port 1.1, the following conditions must be met:

- TC0 output enable flag TOE0 must be set to "1"
- I/O mode flag for P1.1 (PM1.1) must be set to output mode ("1")
- Output latch value for P1.1 must be set to "0"

In summary, the operational sequence required to output a TC0-generated clock signal to the TCLO0 pin is as follows:

1. Load your reference value to TREF0
2. Set the clock frequency in TMOD0
3. Initiate TC0 clock output to TCLO0 (TMOD0.2 = "1")
4. Set port 1.1 mode flag (PM1.1) to "1"
5. Set P1.1 output latch to "0"
6. Set TOE0 flag to "1"

Each time the contents of TCNT0 and TREF0 coincide and an interrupt request is generated, the state of the output latch TOL0 is inverted and the TC0-generated clock signal is output to the TCLO0 pin.

 **PROGRAMMING TIP — TC0 Signal Output to the TCLO0 Pin**

Output a 30 ms pulse width signal to the TCLO0 pin:

```

BITS      EMB
SMB       15
LD        EA,#79H
LD        TREF0,EA
LD        EA,#4CH
LD        TMOD0,EA
LD        EA,#20H
LD        PMG1,EA      ; P1.1 ← Output mode
BITR      P1.1         ; P1.1 clear
BITS      TOE0

```


TC0 EXTERNAL INPUT SIGNAL DIVIDER

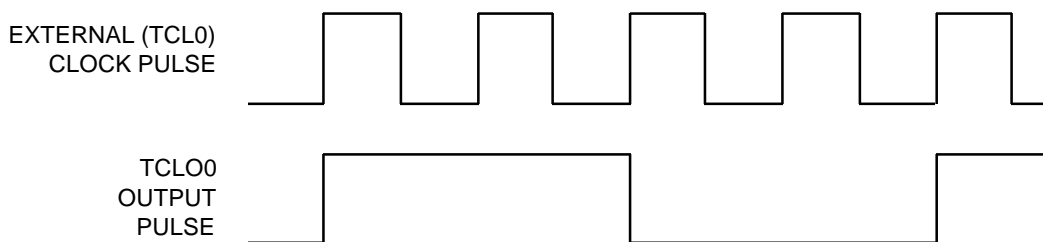
By selecting an external clock source and loading a reference value into the TC0 reference register, TREF0, you can divide the incoming clock signal by the TREF0 value and then output this modified clock frequency to the TCLO0 pin. The sequence of operations used to divide external clock input may be summarized as follows:

1. Load a signal divider value to the TREF0 buffer register
2. Clear TMOD0.6 to "0" to enable external clock input at the TCL0 pin
3. Set TMOD0.5 and TMOD0.4 to desired TCL0 signal edge detection
4. Set port 1.1 mode flag (PM1.1) to output ("1")
5. Set P1.1 output latch to "0"
6. Set TOE0 flag to "1" to enable output of the divided frequency

Divided clock signals are then output to the TCLO0 pin.

PROGRAMMING TIP — External TCL0 Clock Output to the TCLO0 Pin

Output external TCL0 clock pulse to the TCLO0 pin (divide by four):



```

BITS      EMB
SMB       15
LD        EA,#01H
LD        TREF0,EA
LD        EA,#0CH
LD        TMOD0,EA
LD        EA,#20H
LD        PMG1,EA      ; P1.1 ← Output mode
BITR      P1.1         ; P1.1 clear
BITS      TOE0
  
```

TC0 MODE REGISTER (TMOD0)

TMOD0 is the 8-bit mode control register for the timer/counter. It is located at the RAM addresses F90H–F91H and is addressable by 8-bit write instructions. One bit, TMOD0.3, is also 1-bit writeable. A RESET clears all TMOD0 bits to logic zero and disables TC0 operations.

F90H	TMOD0.3	TMOD0.2	"0"	"0"
F91H	"0"	TMOD0.6	TMOD0.5	TMOD0.4

TMOD0.2 is the enable/disable bit for the timer/counter. When TMOD0.3 is set to "1", the contents of TCNT0, IRQT0, and TOL0 are cleared, counting starts from 00H, and TMOD0.3 is automatically reset to "0" for normal TC0 operation. When TC0 operation stops (TMOD0.2 = "0"), the contents of the TC0 counter register, TCNT0, are retained until TC0 is re-enabled.

Use TMOD0.6, TMOD0.5, and TMOD0.4 bit settings together to select the TC0 clock source. This selection involves two variables:

- Synchronization of timer/counter operations with either the rising edge or the falling edge of the clock signal input at the TCL0 pin, and
- Selection of one of four frequencies, based on the division of the incoming system clock frequency, for use in internal TC0 operation.

Table 11-7. TC0 Mode Register (TMOD0) Organization

Bit Name	Setting	Resulting TC0 Function	Address
TMOD0.7	0	MSB value always logic zero	F91H
TMOD0.6 TMOD0.5 TMOD0.4	0,1	Specify input clock edge and internal frequency	
TMOD0.3	1	Clear TCNT0, IRQT0, and TOL0 and resume counting immediately (This bit is automatically cleared to logic zero immediately after counting resumes.)	
TMOD0.2	0	Disable timer/counter; retain TCNT0 contents	F90H
	1	Enable timer/counter	
TMOD0.1	0	Value always logic zero	
TMOD0.0	0	LSB value always logic zero	

Table 11-8. TMOD0.6, TMOD0.5, and TMOD0.4 Bit Settings

TMOD0.6	TMOD0.5	TMOD0.4	Resulting Counter Source and Clock Frequency
0	0	0	External clock input (TCL0) on rising edges
0	0	1	External clock input (TCL0) on falling edges
1	0	0	$f_x/2^{10} = 4.09 \text{ kHz}$
1	0	1	$f_x/2^6 = 65.5 \text{ kHz}$
1	1	0	$f_x/2^4 = 262 \text{ kHz}$
1	1	1	$f_x = 4.19 \text{ MHz}$

NOTE: 'fx' = system clock

PROGRAMMING TIP — Restarting TC0 Counting Operation

1. Set TC0 timer interval to 4.09 kHz:

```

BITS      EMB
SMB      15
LD       EA,#4CH
LD       TMOD0,EA
EI
BITS      IET0

```

2. Clear TCNT0, IRQT0, and TOL0 and restart TC0 counting operation:

```

BITS      EMB
SMB      15
BITS      TMOD0.3

```

TC0 COUNTER REGISTER (TCNT0)

The 8-bit counter register for the timer/counter, TCNT0, is mapped to the RAM addresses F94H–F95H. It is read-only and can be addressed by 8-bit RAM control instructions. A RESET sets all TCNT0 register values to logic zero (00H).

Whenever TMOD0.3 are enabled, TCNT0 is cleared to logic zero and counting begins. The TCNT0 register value is incremented each time an incoming clock signal that matches the signal edge and frequency setting of the TMOD0 register (specifically, TMOD0.6, TMOD0.5, and TMOD0.4) is detected.

Each time TCNT0 is incremented, the new value is compared to the reference value stored in the TC0 reference register, TREF0. When $TCNT0 = TREF0$, an overflow occurs in the TCNT0 register, the interrupt request flag, IRQT0, is set to logic one, and an interrupt request is generated to indicate that the specified timer/counter interval has elapsed.

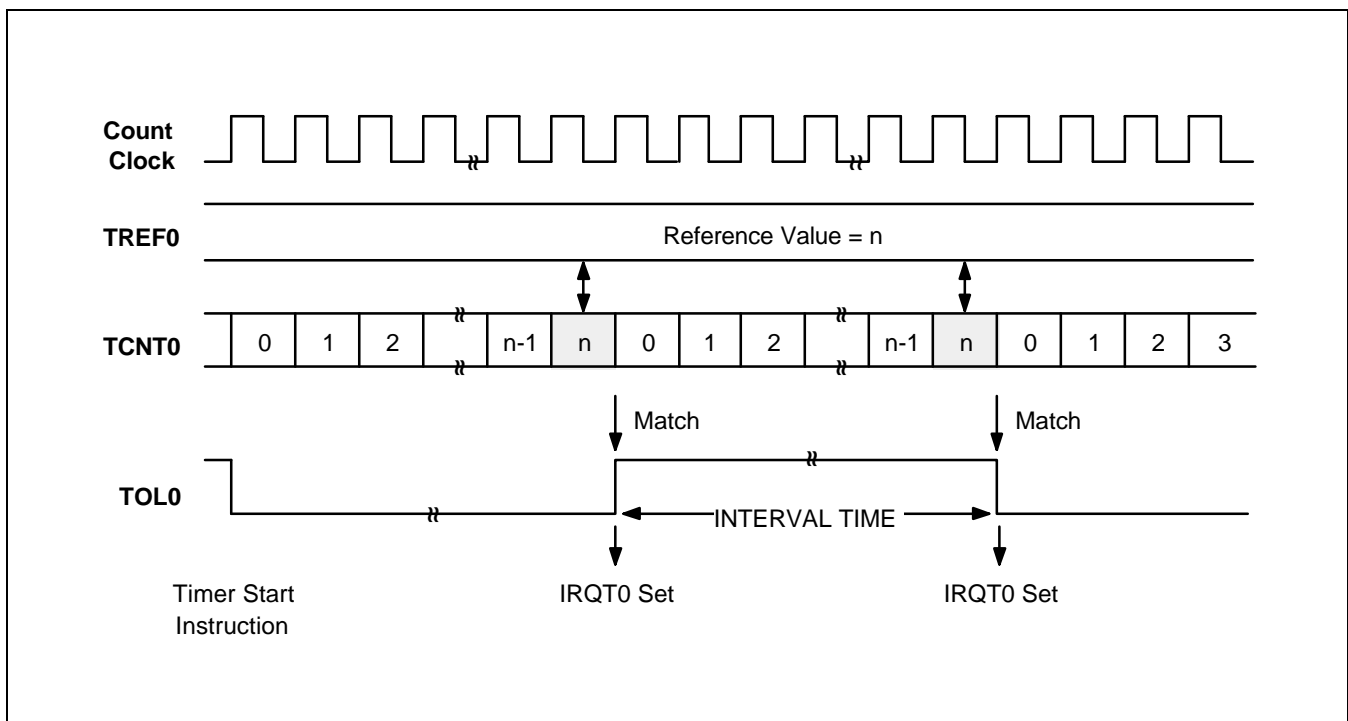


Figure 11-3. TC0 Timing Diagram

TC0 REFERENCE REGISTER (TREF0)

The TC0 reference register TREF0 is an 8-bit write-only register that is mapped to the RAM locations F96H and F97H. It is addressable by 8-bit RAM control instructions. A RESET initializes the TREF0 value to 'FFH'.

TREF0 is used to store a reference value to be compared to the incrementing TCNT0 register in order to identify an elapsed time interval. Reference values will differ depending upon the specific function that TC0 is being used to perform — as a programmable timer/counter, event counter, clock signal divider, or arbitrary frequency output source.

During timer/counter operation, the value loaded into the reference register is compared to the TCNT0 value. When TCNT0 = TREF0, the TC0 output latch (TOL0) is inverted and an interrupt request is generated to signal the interval or event.

The TREF0 value, together with the TMOD0 clock frequency selection, determines the specific TC0 timer interval. Use the following formula to calculate the correct value to load to the TREF0 reference register:

$$\text{TC0 timer interval} = (\text{TREF0 value} + 1) \times \frac{1}{\text{TMOD0 frequency setting}}$$

(assuming a TREF0 value $\neq 0$)

TC0 OUTPUT ENABLE FLAG (TOE0)

The 8-bit timer/counter 0 output enable flag TOE0 controls output from timer/counter 0 to the TCLO0 pin. TOE0 is mapped to RAM location F92H.2 and is addressable by 1/4-bit read or write instructions.

	Bit 3	Bit 2	Bit 1	Bit 0
F92H	0	TOE0	0	0

When you set the TOE0 flag to "1", the contents of TOL0 can be output to the TCLO0 pin. Whenever a RESET occurs, TOE0 is automatically set to logic zero, disabling all TC0 output. Even when the TOE0 flag is disabled, the timer/counter can continue to output an internally-generated clock frequency, via TOL0.

TC0 OUTPUT LATCH (TOL0)

TOL0 is the output latch for the timer/counter. When the 8-bit comparator detects a correspondence between the value of the counter register TCNT0 and the reference value stored in the TREF0 buffer, the TOL0 value is inverted — the latch toggles high-to-low or low-to-high.

Whenever the state of TOL0 is switched, the TC0 signal is output. TC0 output may be directed to the TCLO0 pin at P1.1.

Assuming TC0 is enabled, when bit 3 of the TMOD0 register is set to "1", the TOL0 latch is cleared to logic zero, along with the counter register TCNT0 and the interrupt request flag, IRQT0, and counting resumes immediately. When TC0 is disabled (TMOD0.2 = "0"), the contents of the TOL0 latch are retained and can be read, if necessary.

PROGRAMMING TIP — Setting a TC0 Timer Interval

To set a 30 ms timer interval for TC0, given $f_x = 4.19$ MHz, follow these steps.

1. Select the timer/counter mode register with a maximum setup time of 62.5 ms (assume the TC0 counter clock = $f_x/2^{10}$, and TREF0 is set to FFH):
2. Calculate the TREF0 value:

$$30 \text{ ms} = \frac{\text{TREF0 value} + 1}{4.09 \text{ kHz}}$$

$$\text{TREF0} + 1 = \frac{30 \text{ ms}}{244 \mu\text{s}} = 122.9 = 7AH$$

$$\text{TREF0 value} = 7AH - 1 = 79H$$

3. Load the value 79H to the TREF0 register:

BITS	EMB
SMB	15
LD	EA,#79H
LD	TREF0,EA
LD	EA,#4CH
LD	TMOD0,EA

12 BUZZER

OVERVIEW

Buzzer signal output is used to generate 4 specific frequency signals for buzzer sound. This function is controlled by the buzzer mode register, BUMOD. By writing an appropriate value to the buzzer mode register, the signal can be generated to the buzzer output pin. The signal frequency is selected by the buzzer mode register. The buzzer output circuit has the following components:

- 4-bit buzzer mode register (BUZMOD)
- Buzzer output pin (BUZ)

FUNCTION DESCRIPTION

The buzzer signal output circuit can generate a steady 0.5 kHz, 1 kHz, 2 kHz, or 4 kHz signal to the BUZ pin. To generate the buzzer frequency, load an appropriate value to the BUZMOD. Then the frequency selected according to the value of BUZMOD is generated to BUZ pin to actuate an external buzzer sound when the buzzer signal frequency is generated, P1.3 shared with BUZ pin must be assigned to output port and the value of output latch must be logic zero.

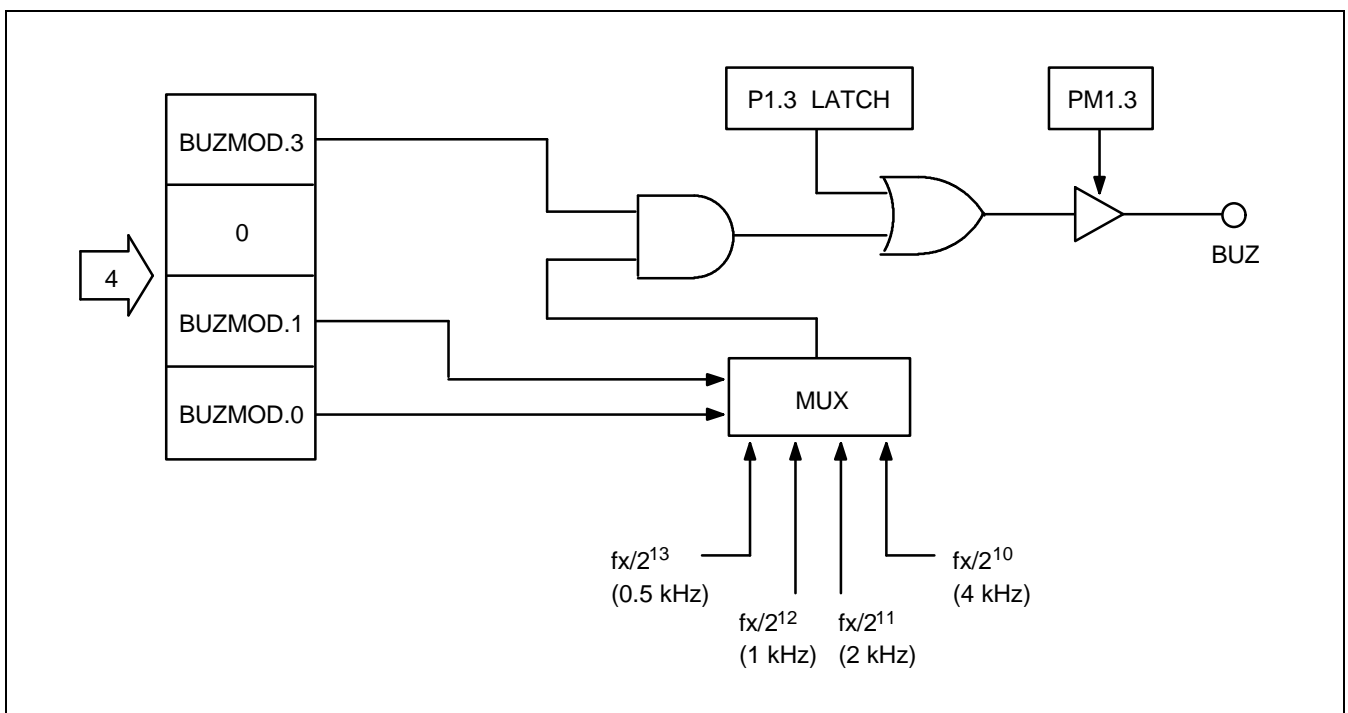


Figure 12-1. Buzzer Circuit Diagram

BUZZER MODE REGISTER (BUZMOD)

F88H	BUZMOD.3	"0"	BUZMOD.1	BUZMOD.0
------	----------	-----	----------	----------

Buzzer signal output is controlled by the buzzer mode register, BUZMOD. It is a 4-bit write-only addressable register. Bit position BUZMOD. 3 enables or disables buzzer output operation. If BUZMOD.3 is set to logic one, buzzer output operation. If BUZMOD.3 is set to logic one, buzzer output is enabled. BUZMOD.0–1 is used to select the signal frequency for buzzer sound. A RESET clears all BUZMOD bits to logic zero and the buzzer output operation is disabled.

Table 12-1. Buzzer Mode Register (BUZMOD) Organization

Bit Name	Values		Function	Address
BUZMOD.3	0		Disable buzzer (BUZ) signal output	F88H
	1		Enable buzzer (BUZ) signal output	
BUZMOD.2	"0"		Always logic zero	
BUZMOD.1–.0	0	0	0.5 kHz buzzer (BUZ) signal output	
	0	1	1 kHz buzzer (BUZ) signal output	
	1	0	2 kHz buzzer (BUZ) signal output	
	1	1	4 kHz buzzer (BUZ) signal output	

NOTE: System clock frequency (fx) is assumed to be 4.19 MHz.

 **PROGRAMMING TIP — BUZ Signal Output to the BUZ Pin**

Output a 1 kHz buzzer signal to the BUZ pin

```

BITS      EMB
SMB      15
LD       EA, #80H
LD       PMG1, EA      ; P1.3 ← output mode
BITR     P1.3          ; P1.3 clear
LD       A, #9H
LD       BUZMOD, A     ; 1 kHz buzzer signal output

```


13

D/A CONVERTER

OVERVIEW

The 8-bit D/A Converter (DAC) module uses successive approximation logic to convert 8-bit digital values to equivalent analog levels between $V_{DD} (1 - \frac{1}{256})$ and V_{SS} .

This D/A Converter consists of R–2R array structure. The D/A Converter has the following components:

- R–2R array structure
- Digital-to-analog converter mode register (DAMOD)
- Digital-to-analog converter data register (DADATA)
- Digital-to-analog converter output pin (DAO)

FUNCTION DESCRIPTION

To initiate a digital-to-analog conversion procedure, you should set the digital-to-analog converter enable bit (DAMOD.0).

The DAMOD register is an 1/4-bit write-only register located at the RAM address FD5H. You should write the digital value calculated to digital-to-analog converter data register (DADATA).

The DADATA register is an 8-bit write-only register located at the RAM address FD6H–FD7H.

NOTE

If the chip enters to power-down mode, STOP or IDLE, in conversion process, there will be current path in D/A Converter block. So, it is necessary to cut off the current path before the instruction execution enters power-down mode.

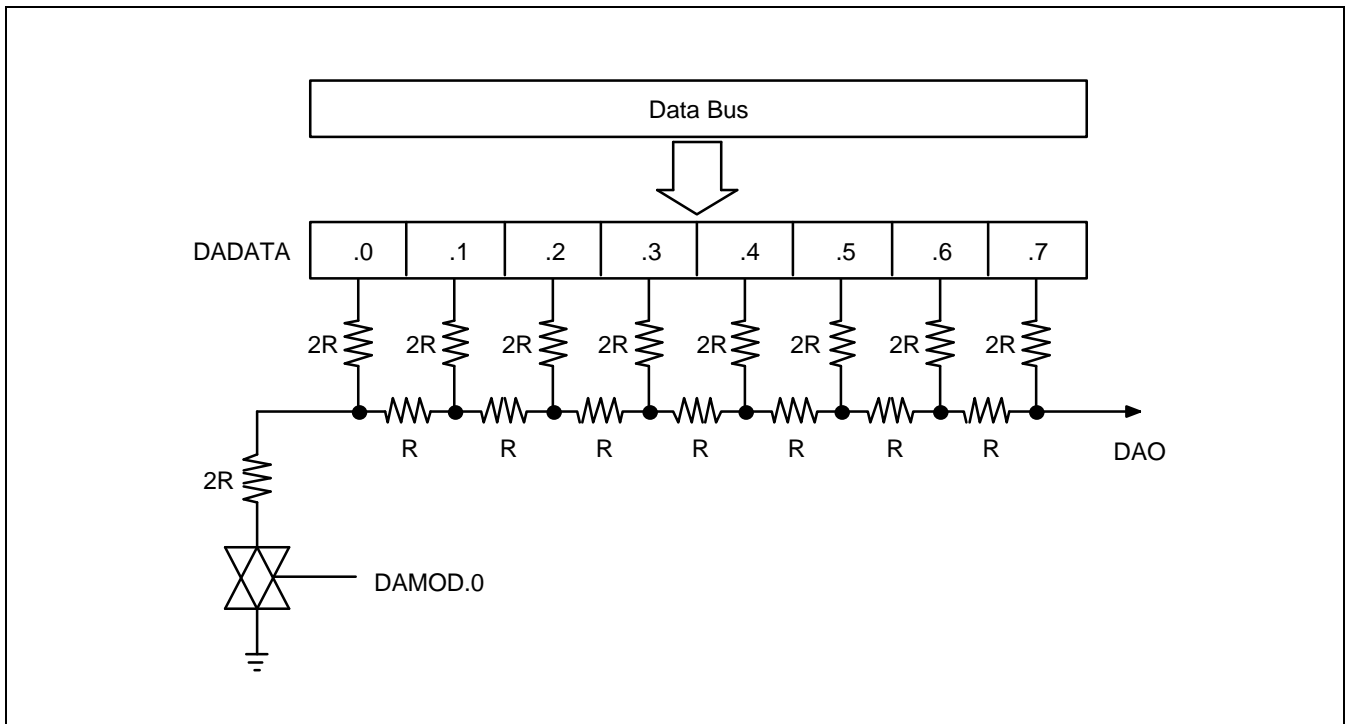


Figure 13-1. DAC Circuit Diagram

D/A CONVERTER MODE REGISTER (DAMOD)

The Digital-to-Analog Converter (DAC) mode register, DAMOD is a 1/4-bit write-only register located at the RAM address FD5H. DAMOD controls enabling or disabling DAC. DAMOD values are set to logic "0H" after a RESET and this value disables DAC.

DAMOD – Digital-to-Analog mode register **FD5H**

Bit	3	2	1	0
Identifier	"0"	"0"	"0"	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W

DAMOD.0	Digital-to-Analog Converter Enable/Disable control
0	Disable Digital-to-Analog Converter
1	Enable Digital-to-Analog Converter

D/A CONVERTER DATA REGISTER (DADATA)

The DAC DATA register, DATATA, is an 8-bit write-only register located at the RAM address, FD6H–FD7H. DADATA specifies the digital data to generate analog voltage. A RESET initializes the DADATA value to “00H”. The D/A Converter output value, V_{DAO} , is calculated by the following formula.

$$V_{\text{DAO}} = V_{\text{DD}} \times \frac{n}{256} \quad (n = 0\text{--}255, \text{DADATA value})$$

Table 13-1. DADATA Setting to Generate Analog Voltage

DADATA.7	DADATA.6	DADATA.5	DADATA.4	DADATA.3	DADATA.2	DADATA.1	DADATA.0	V_{DAO}
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	$V_{\text{DD}}/2^1$
0	1	0	0	0	0	0	0	$V_{\text{DD}}/2^2$
0	0	1	0	0	0	0	0	$V_{\text{DD}}/2^3$
0	0	0	1	0	0	0	0	$V_{\text{DD}}/2^4$
0	0	0	0	1	0	0	0	$V_{\text{DD}}/2^5$
0	0	0	0	0	1	0	0	$V_{\text{DD}}/2^6$
0	0	0	0	0	0	1	0	$V_{\text{DD}}/2^7$
0	0	0	0	0	0	0	1	$V_{\text{DD}}/2^8$

NOTE: These are the values determined by setting just one-bit of DADATA.0–DADATA.7. Other values of DAO can be obtained with superimposition.

14 ELECTRICAL DATA

OVERVIEW

In this section, S3C7544 electrical characteristics are presented in tables and graphs. The information is arranged in the following order:

Standard Electrical Characteristics

- Absolute maximum ratings
- D.C. electrical characteristics
- Main system clock oscillator characteristics
- Subsystem clock oscillator characteristics
- I/O capacitance
- A.C. electrical characteristics
- Operating voltage range

Miscellaneous Timing Waveforms

- A.C timing measurement point
- Clock timing measurement at X_{iN}
- Clock timing measurement at XT_{iN}
- TCL timing
- Input timing for RESET
- Input timing for external interrupts
- Serial data transfer timing

Stop Mode Characteristics and Timing Waveforms

- RAM data retention supply voltage in stop mode
- Stop mode release timing when initiated by RESET
- Stop mode release timing when initiated by an interrupt request

Table 14-1. Absolute Maximum Ratings

(T_A = 25 °C)

Parameter	Symbol	Conditions	Rating	Units
Supply Voltage	V _{DD}	–	– 0.3 to + 6.5	V
Input Voltage	V _I	All I/O ports	– 0.3 to V _{DD} + 0.3	V
Output Voltage	V _O	–	– 0.3 to V _{DD} + 0.3	V
Output Current High	I _{OH}	One I/O port active	– 5	mA
		All I/O ports active	– 35	
Output Current Low	I _{OL}	One I/O port active	+ 30 (peak)	mA
			+ 15 (note)	
		All I/O ports active	+ 100 (peak)	
			+ 60 (note)	
Operating Temperature	T _A	–	– 40 to + 85	°C
Storage Temperature	T _{stg}	–	– 65 to + 150	°C

NOTE: The values for output current low (I_{OL}) are calculated as peak value × √Duty .

Table 14-2. D.C. Electrical Characteristics

(T_A = – 40 °C to + 85 °C, V_{DD} = 1.8 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Units
Input High Voltage	V _{IH1}	All input pins except V _{IH2} –V _{IH3}	0.7 V _{DD}	–	V _{DD}	V
	V _{IH2}	P0 and RESET	0.8 V _{DD}	–	V _{DD}	
	V _{IH3}	X _{IN} and X _{OUT}	V _{DD} – 0.1	–	V _{DD}	
Input Low Voltage	V _{IL1}	All input pins except V _{IH2} –V _{IH3}	–	–	0.3 V _{DD}	V
	V _{IL2}	P0 and RESET			0.2 V _{DD}	
	V _{IL3}	X _{IN} and X _{OUT}			0.1	

Table 14-2. D.C. Electrical Characteristics (Continued)

 $(T_A = -40\text{ }^\circ\text{C to } +85\text{ }^\circ\text{C}, V_{DD} = 1.8\text{ V to } 5.5\text{ V})$

Parameter	Symbol	Conditions	Min	Typ	Max	Units
Output High Voltage	V_{OH}	$V_{DD} = 4.5\text{ V to } 5.5\text{ V}$ $I_{OH} = -1\text{ mA}$	$V_{DD} - 1.0$	–	–	V
Output Low Voltage	V_{OL1}	$V_{DD} = 4.5\text{ V to } 5.5\text{ V}$ $I_{OL} = 15\text{ mA}$ Ports 4, 5	–	–	2	V
		$V_{DD} = 1.8\text{ V to } 5.5\text{ V}$ $I_{OL} = 1.6\text{ mA}$			0.4	
	V_{OL2}	$V_{DD} = 4.5\text{ V to } 5.5\text{ V}$ $I_{OL} = 4\text{ mA}$ All out ports except ports 4, 5			2	
		$V_{DD} = 1.8\text{ V to } 5.5\text{ V}$ $I_{OL} = 1.6\text{ mA}$			0.6	
Input High Leakage Current	I_{LH1}	$V_{IN} = V_{DD}$ All input pins except X_{IN} and X_{OUT}	–	–	3	μA
	I_{LH2}	$V_{IN} = V_{DD}$ X_{IN} and X_{OUT}			20	
Input Low Leakage Current	I_{LIL1}	$V_{IN} = 0\text{ V}$ All input pins except X_{IN} , X_{OUT} and RESET	–	–	–3	μA
	I_{LIL2}	$V_{IN} = 0\text{ V}$ X_{IN} and X_{OUT}			–20	
Output High Leakage Current	I_{LOH}	$V_O = V_{DD}$ All output pins	–	–	3	μA
Output Low Leakage Current	I_{LOL}	$V_O = 0\text{ V}$ All output pins	–	–	–3	μA
Pull-up Resistor	R_{L1}	$V_{DD} = 5\text{ V}; V_I = 0\text{ V}$ except RESET	25	45	100	$\text{k}\Omega$
		$V_{DD} = 3\text{ V}$	50	90	200	
	R_{L2}	$V_{DD} = 5\text{ V}; V_I = 0\text{ V}; \text{RESET}$	100	220	400	
		$V_{DD} = 3\text{ V}$	200	450	800	

Table 14-2. D.C. Electrical Characteristics (Concluded)

($T_A = -40\text{ }^\circ\text{C}$ to $+85\text{ }^\circ\text{C}$, $V_{DD} = 1.8\text{ V}$ to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Units	
Supply Current (1)	I_{DD1} (DAC on)	Run mode; $V_{DD} = 5.0\text{ V} \pm 10\%$	6.0MHz	-	3.4	10.0	mA
		Crystal oscillator; $C1 = C2 = 22\text{pF}$	4.19MHz		2.7	8.0	
	I_{DD2} (DAC off)	Run mode; $V_{DD} = 5.0\text{ V} \pm 10\%$	6.0MHz	-	2.3	8.0	mA
			Crystal oscillator; $C1 = C2 = 22\text{pF}$		4.19MHz	1.7	
		$V_{DD} = 3\text{ V} \pm 10\%$	6.0MHz		1.1	4.0	
			4.19MHz		0.8	3.0	
	I_{DD3}	Idle mode; $V_{DD} = 5.0\text{ V} \pm 10\%$	6.0MHz	-	0.7	2.5	mA
			Crystal oscillator; $C1 = C2 = 22\text{pF}$		4.19MHz	0.5	
		$V_{DD} = 3\text{ V} \pm 10\%$	6.0MHz		0.3	1.5	
			4.19MHz		0.2	1.0	
I_{DD4}	Stop mode; $V_{DD} = 5.0\text{ V} \pm 10\%$		-	0.2	3.0	μA	
	Stop mode; $V_{DD} = 3.0\text{ V} \pm 10\%$			0.1	2.0		

NOTES:

- D.C. electrical values for supply current (I_{DD1} to I_{DD3}) do not include the current drawn through internal pull-up resistors.
- I_{DD1} typical values are measured when DADATA register value is 055H.

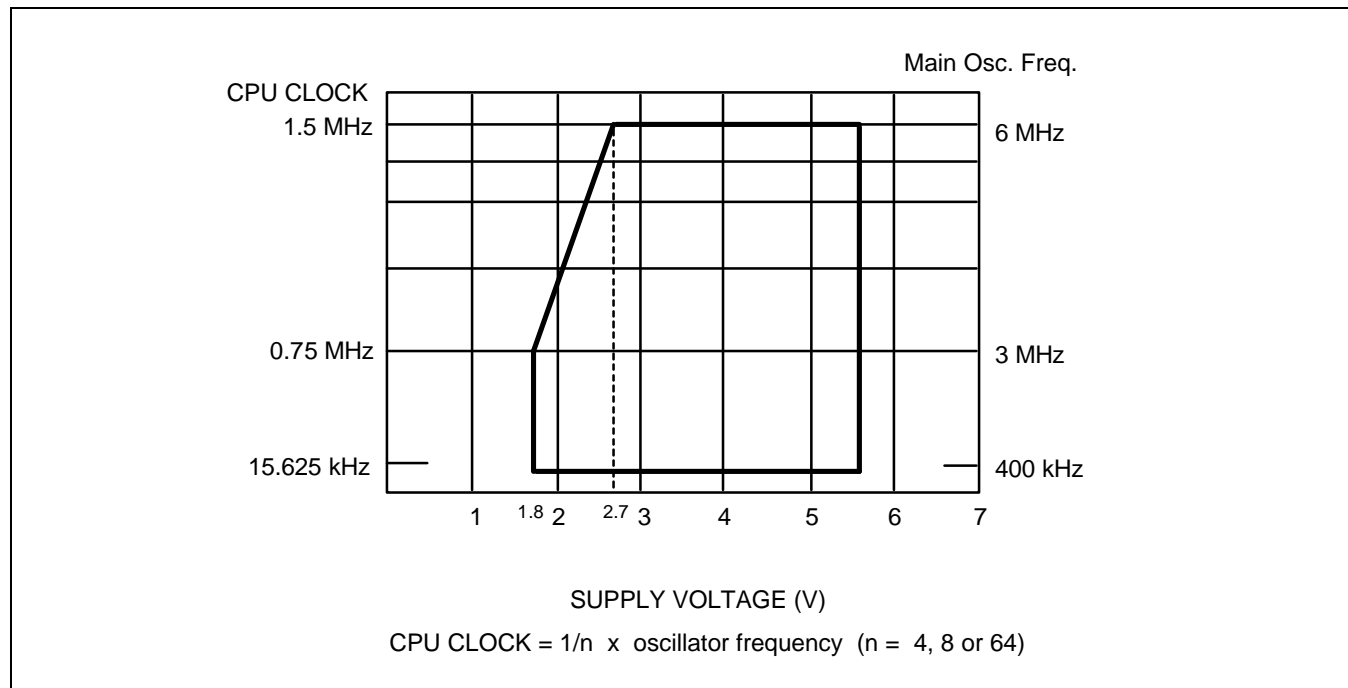
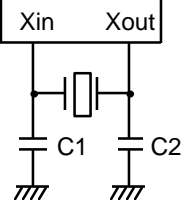
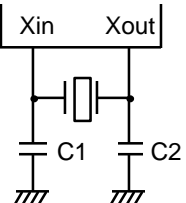
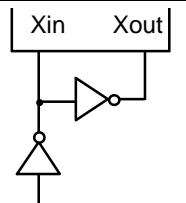


Figure 14-1. Standard Operating Voltage Range

Table 14-3. Oscillators Characteristics

(T_A = -40 °C + 85 °C, V_{DD} = 1.8 V to 5.5 V)

Oscillator	Clock Configuration	Parameter	Test Condition	Min	Typ	Max	Units
Ceramic Oscillator		Oscillation frequency ⁽¹⁾	V _{DD} = 2.7 V to 5.5 V	0.4	–	6.0	MHz
			V _{DD} = 1.8 V to 5.5 V	0.4	–	3	
		Stabilization time ⁽²⁾	V _{DD} = 3.0 V	–	–	4	ms
Crystal Oscillator		Oscillation frequency ⁽¹⁾	V _{DD} = 2.7 V to 5.5 V	0.4	–	6.0	MHz
			V _{DD} = 1.8 V to 5.5 V	0.4	–	3	
		Stabilization time ⁽²⁾	V _{DD} = 3.0 V	–	–	10	ms
External Clock		X _{IN} input frequency ⁽¹⁾	V _{DD} = 2.7 V to 5.5 V	0.4	–	6.0	MHz
			V _{DD} = 1.8 V to 5.5 V	0.4	–	3	
		X _{IN} input high and low level width (t _{XH} , t _{XL})	–	83.3	–	1250	ns

NOTES:

- Oscillation frequency and X_{IN} input frequency data are for oscillator characteristics only.
- Stabilization time is the interval required for oscillating stabilization after a power-on occurs, or when stop mode is terminated.

Table 14-4. Recommended Oscillator Constants

(T_A = -40 °C + 85 °C, V_{DD} = 1.8 V to 5.5 V)

Manufacturer	Series Number ⁽¹⁾	Frequency Range	Load Cap (pF)		Oscillator Voltage Range (V)		Remarks
			C1	C2	MIN	MAX	
TDK	FCR 05M5	3.58 MHz–6.0 MHz	33	33	2.0	5.5	Leaded Type
	FCR 05MC5	3.58 MHz–6.0 MHz	(2)	(2)	2.0	5.5	On-chip C Leaded Type
	CCR 05MC3	3.58 MHz–6.0 MHz	(3)	(3)	2.0	5.5	On-chip C SMD Type

NOTES:

1. Please specify normal oscillator frequency.
2. On-chip C: 30pF built in.
3. On-chip C: 38pF built in.

Table 14-5. Input/Output Capacitance

(T_A = 25 °C, V_{DD} = 0 V)

Parameter	Symbol	Condition	Min	Typ	Max	Units
Input Capacitance	C _{IN}	f = 1 MHz; Unmeasured pins are returned to V _{SS}	–	–	15	pF
Output Capacitance	C _{OUT}				15	pF
I/O Capacitance	C _{IO}				15	pF

Table 14-6. D/A Converter Electrical Characteristics

(T_A = -40 °C to +85 °C, V_{DD} = 3.5 V to 5.5 V, V_{SS} = 0 V)

Parameter	Symbol	Condition	Min	Typ	Max	Units
Resolution	–	–	–	–	8	bits
Absolute Accuracy	–		–3	–	3	LSB
Differential Linearity Error	DLE		–1	–	1	LSB
Setup Time	t _{su}		–	–	5	μs
Output Resistance	R _O		4.5	5	5.5	KΩ

Table 14-7. A.C. Electrical Characteristics

(T_A = -40 °C to +85 °C, V_{DD} = 1.8 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Units
Instruction Cycle Time	t _{CY}	V _{DD} = 2.7 V to 5.5 V	0.67	–	64	μs
		V _{DD} = 1.8 V to 5.5 V	1.33			
TCL0 Input Frequency	f _{TI}	V _{DD} = 2.7 V to 5.5 V	0	–	1.5	MHz
		V _{DD} = 1.8 V to 5.5 V			1	MHz
TCL0 Input High, Low Width	t _{TIH} , t _{TIL}	V _{DD} = 2.7 V to 5.5 V	0.48	–	–	μs
		V _{DD} = 1.8 V to 5.5 V	1.8			
Interrupt Input High, Low Width	t _{INTH} , t _{INTL}	INT0, INT1, KS0–KS1	10	–	–	μs
RESET Input Low Width	t _{RSL}	Input	–	–	10	μs

Table 14-8. RAM Data Retention Supply Voltage in Stop Mode

(T_A = -40 °C to +85 °C)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V _{DDDR}	–	1.8	–	5.5	V
Data retention supply current	I _{DDDR}	V _{DDDR} = 1.8 V	–	0.1	10	μA
Release signal set time	t _{SREL}	–	0	–	–	μs
Oscillator stabilization wait time ⁽¹⁾	t _{WAIT}	Released by RESET	–	2 ¹⁷ /fx	–	ms
		Released by interrupt	–	(2)	–	ms

NOTES:

- During oscillator stabilization wait time, all CPU operations must be stopped to avoid instability during oscillator start-up.
- Use the basic timer mode register (BMOD) interval timer to delay the execution of CPU instructions during the wait time.

TIMING WAVEFORMS

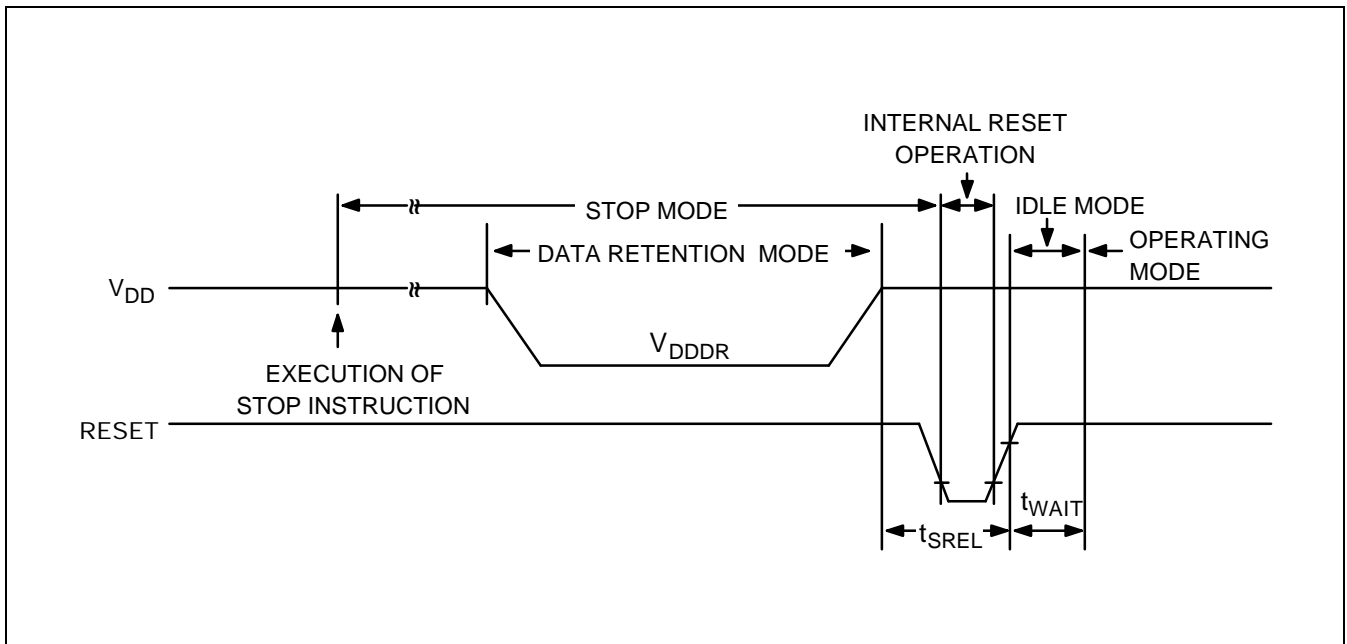


Figure 14-2. Stop Mode Release Timing When Initiated by RESET

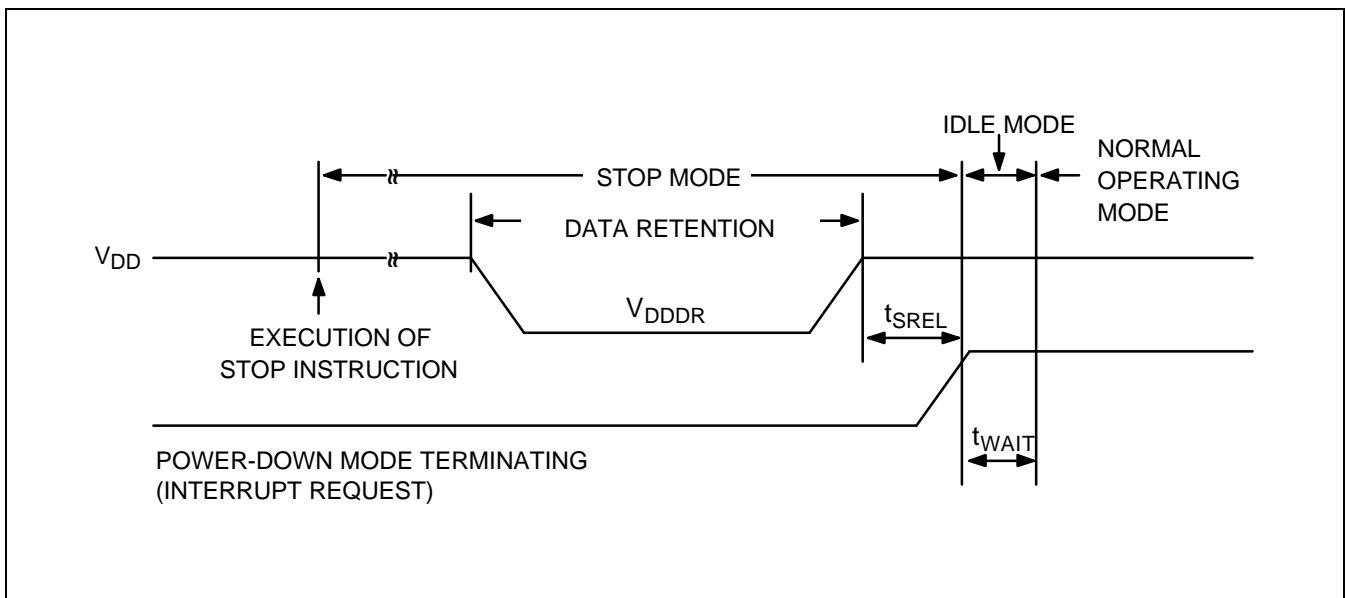


Figure 14-3. Stop Mode Release Timing When Initiated by Interrupt Request

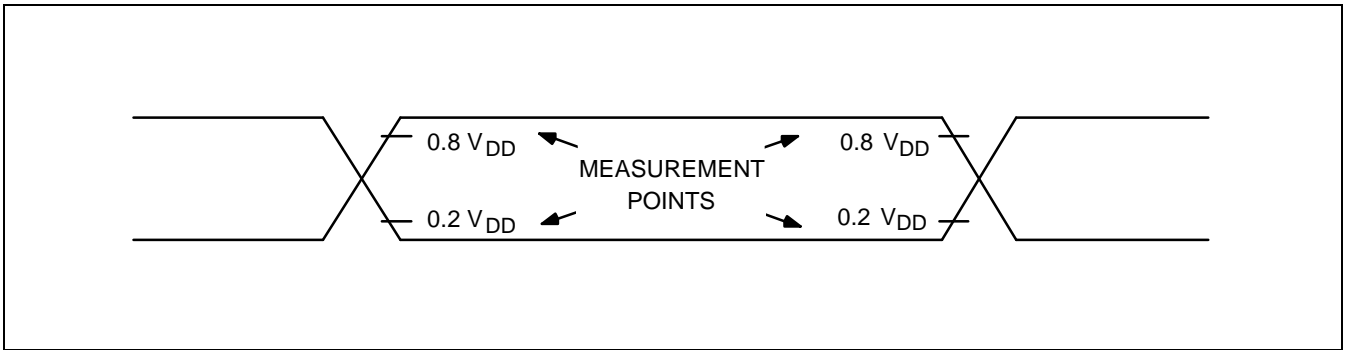


Figure 14-4. A.C. Timing Measurement Points (Except for X_{IN})

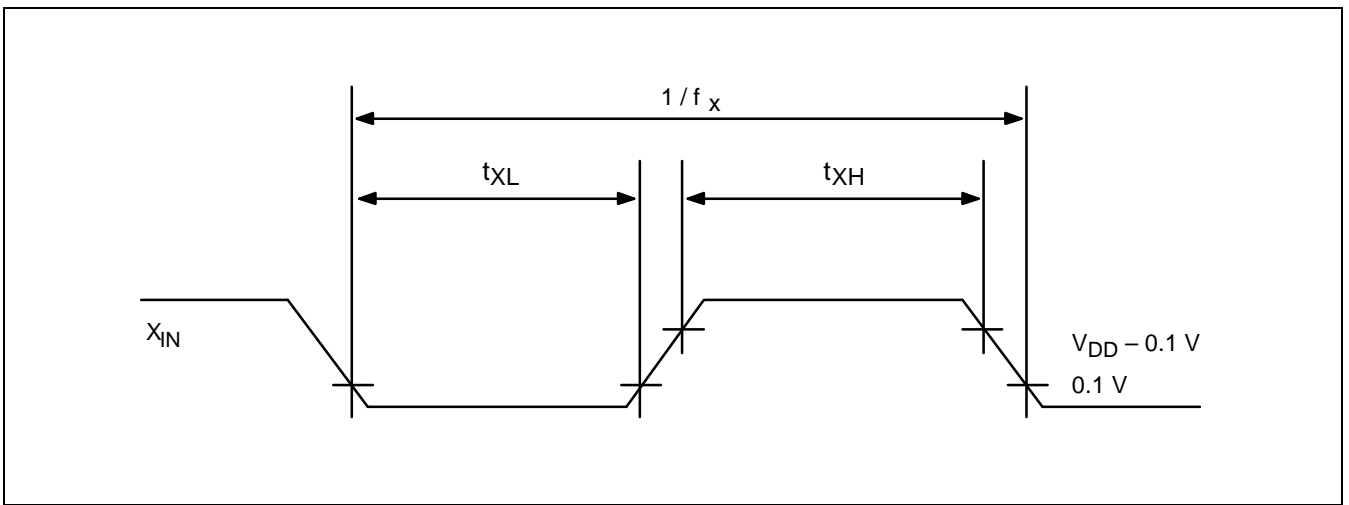


Figure 14-5. Clock Timing Measurement at X_{IN}

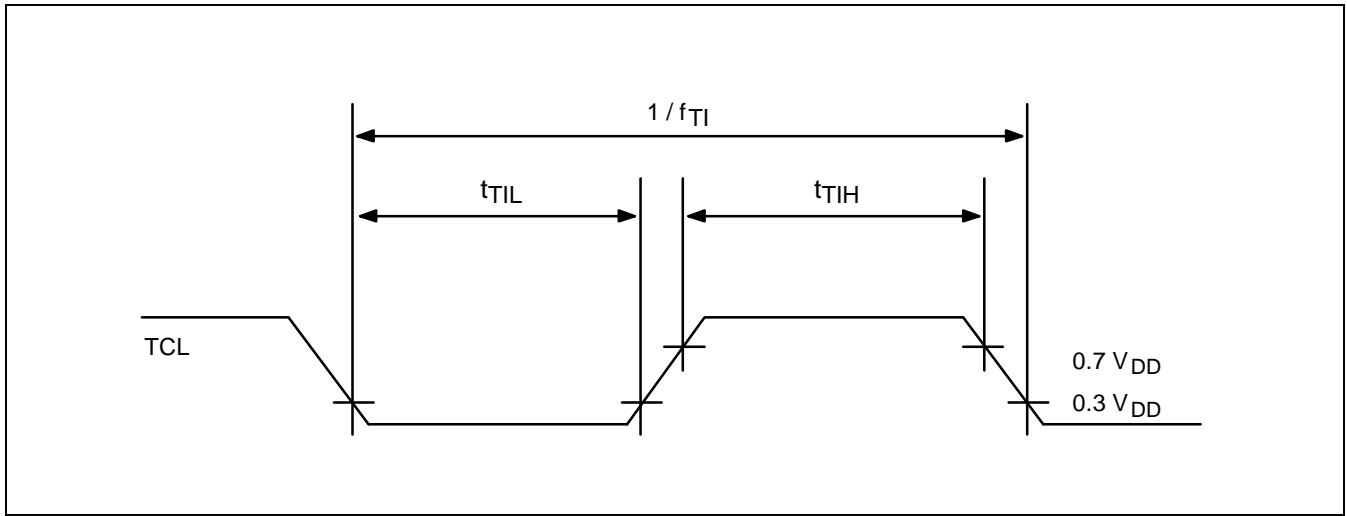


Figure 14-6. TCL Timing

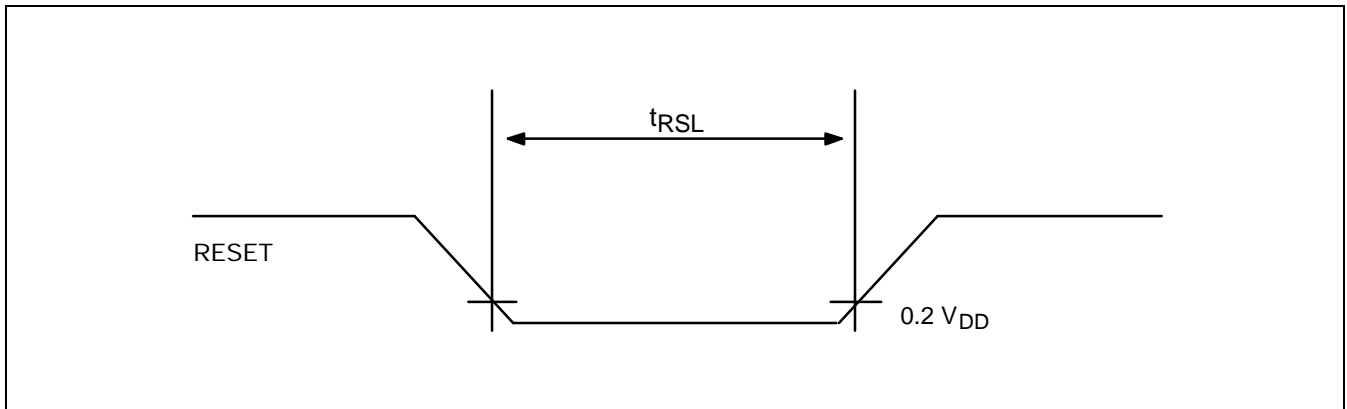


Figure 14-7. Input Timing for RESET Signal

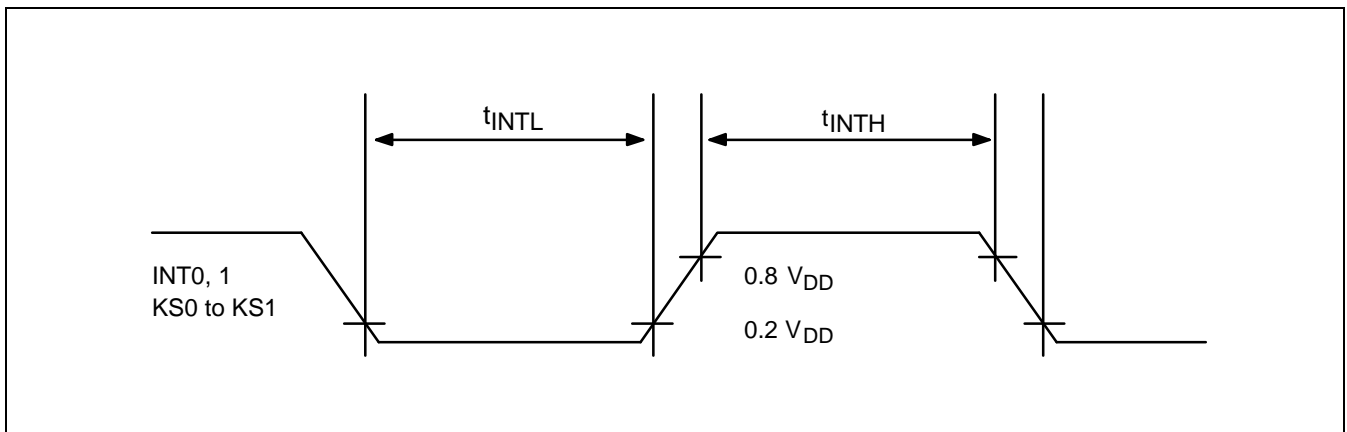


Figure 14-8. Input Timing for External Interrupts

15 MECHANICAL DATA

This section contains the following information about the device package:

- Package dimensions in millimeters
- Pad diagram
- Pad/pin coordinate data table

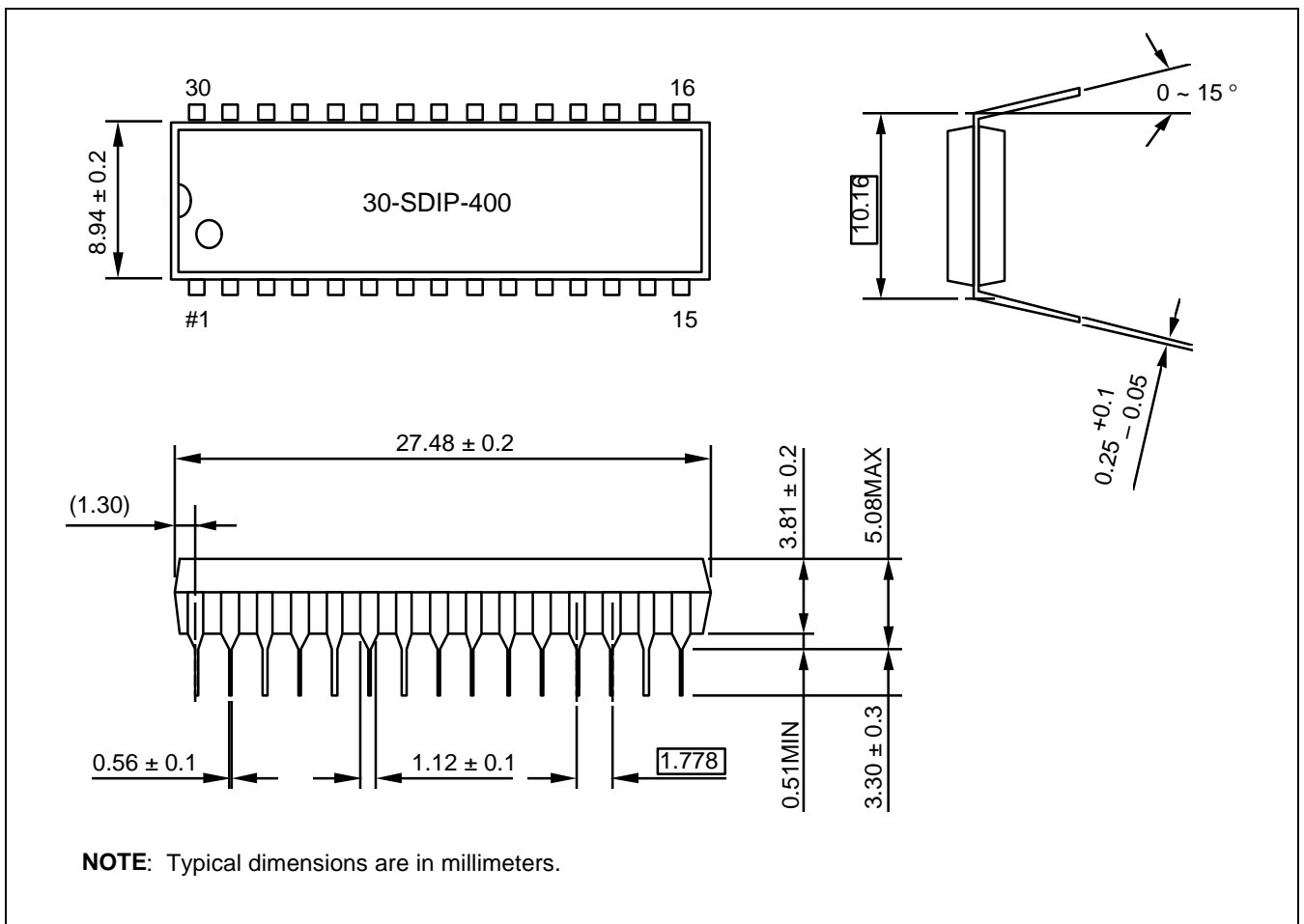


Figure 15-1. 30-SDIP-400 Package Dimensions

16

S3P7544 OTP

OVERVIEW

The S3P7544 single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the S3C7544 microcontroller. It has an on-chip OTP ROM instead of masked ROM. The EPROM is accessed by serial data format.

The S3P7544 is fully compatible with the S3C7544, both in function and in pin configuration. Because of its simple programming requirements, the S3P7544 is ideal for use as an evaluation chip for the S3C7544.

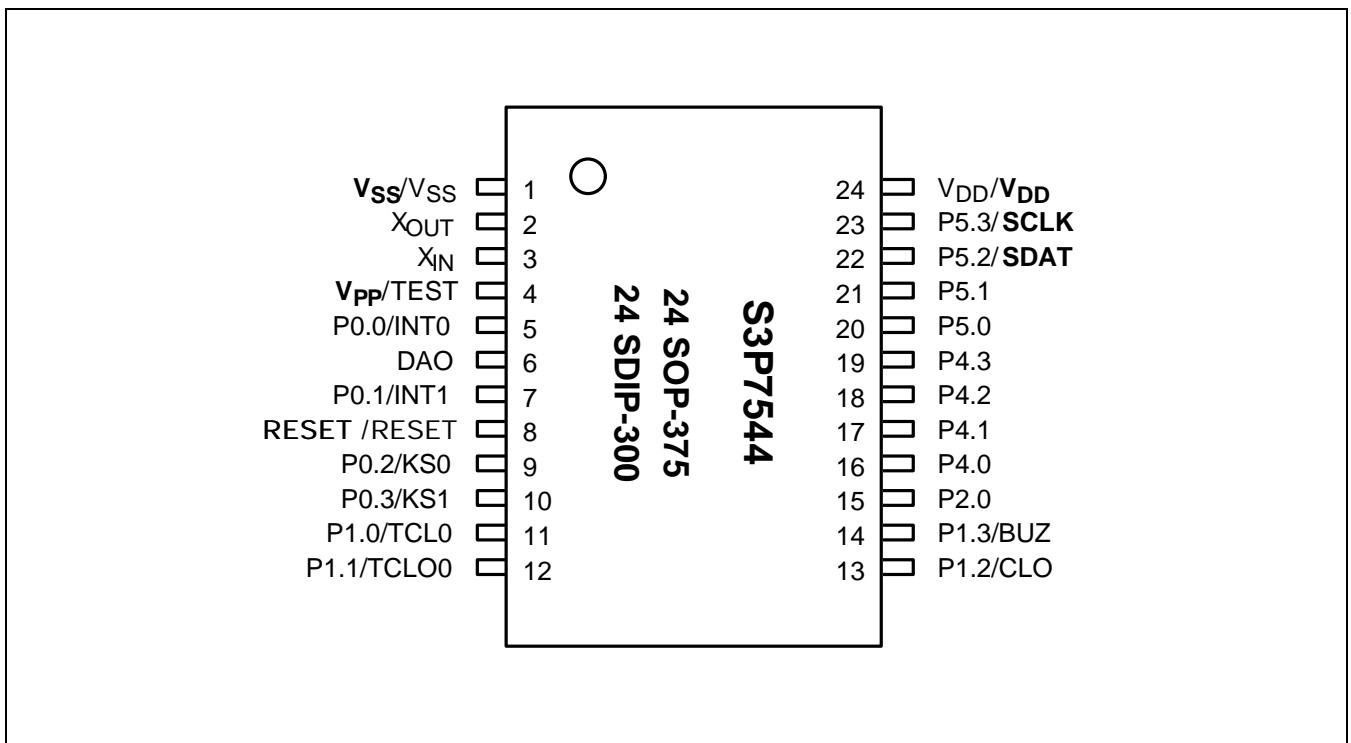


Figure 16-1. S3P7544 Pin Assignments (24 SOP-375, 24 SDIP-300 Package)

Table 16-1. Descriptions of Pins Used to Read/Write the EPROM

Main Chip Pin Name	During Programming			
	Pin Name	Pin No.	I/O	Function
P5.2	SDAT	22	I/O	Serial data pin. Output port when reading and input port when writing. Can be assigned as a Input / push-pull output port.
P5.3	SCLK	23	I/O	Serial clock pin. Input only pin.
TEST	TEST	4	I	Power supply pin for EPROM cell writing (indicates that OTP enters into the writing mode). When 12.5 V is applied, OTP is in writing mode and when 5 V is applied, OTP is in reading mode. (Option) Hold GND when OTP is operating.
RESET	RESET	8	I	Chip initialization
V _{DD} /V _{SS}	V _{DD} /V _{SS}	24/1	–	Logic power supply pin. V _{DD} should be tied to +5 V during programming.

NOTE: () means the 32-SOP OTP pin number.

Table 16-2. Comparison of S3P7544 and S3C7544 Features

Characteristic	S3P7544	S3C7544
Program Memory	4 K-byte EPROM	4 K-byte mask ROM
Operating Voltage (V _{DD})	1.8 V (3 MHz) to 5.5 V	1.8 V (3 MHz) to 5.5 V
OTP Programming Mode	V _{DD} = 5 V, V _{PP} (TEST) = 12.5 V	–
Pin Configuration	24 SOP, 24 SDIP	24 SOP, 24 SDIP
EPROM Programmability	User Program one time	Programmed at the factory

OPERATING MODE CHARACTERISTICS

When 12.5 V is supplied to the V_{PP}(TEST) pin of the S3P7544, the EPROM programming mode is entered. The operating mode (read, write, or read protection) is selected according to the input signals to the pins listed in Table 16-3 below.

Table 16-3. Operating Mode Selection Criteria

V _{DD}	V _{PP} (TEST)	REG/ MEM	Address (A15-A0)	R/W	Mode
5 V	5 V	0	0000H	1	EPROM read
	12.5 V	0	0000H	0	EPROM program
	12.5 V	0	0000H	1	EPROM verify
	12.5 V	1	0E3FH	0	EPROM read protection

NOTE: "0" means Low level; "1" means High level.

OTP ELECTRICAL DATA

Table 16-4. Absolute Maximum Ratings

(T_A = 25 °C)

Parameter	Symbol	Conditions	Rating	Units
Supply Voltage	V _{DD}	–	– 0.3 to + 6.5	V
Input Voltage	V _I	All I/O ports	– 0.3 to V _{DD} + 0.3	V
Output Voltage	V _O	–	– 0.3 to V _{DD} + 0.3	V
Output Current High	I _{OH}	One I/O port active	– 5	mA
		All I/O ports active	– 35	
Output Current Low	I _{OL}	One I/O port active	+ 30 (peak)	mA
			+ 15 (note)	
		All I/O ports active	+ 100 (peak)	
			+ 60 (note)	
Operating Temperature	T _A	–	– 40 to + 85	°C
Storage Temperature	T _{stg}	–	– 65 to + 150	°C

NOTE: The values for output current low (I_{OL}) are calculated as peak value × $\sqrt{\text{Duty}}$.

Table 16-5. D.C. Electrical Characteristics

(T_A = – 40 °C to + 85 °C, V_{DD} = 1.8 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Units
Input High Voltage	V _{IH1}	All input pins except V _{IH2} –V _{IH3}	0.7 V _{DD}	–	V _{DD}	V
	V _{IH2}	P0 and RESET	0.8 V _{DD}	–	V _{DD}	
	V _{IH3}	X _{IN} and X _{OUT}	V _{DD} – 0.1	–	V _{DD}	
Input Low Voltage	V _{IL1}	All input pins except V _{IH2} –V _{IH3}	–	–	0.3 V _{DD}	V
	V _{IL2}	P0 and RESET			0.2 V _{DD}	
	V _{IL3}	X _{IN} and X _{OUT}			0.1	

Table 16-5. D.C. Electrical Characteristics (Continued)

 $(T_A = -40\text{ }^\circ\text{C}$ to $+85\text{ }^\circ\text{C}$, $V_{DD} = 1.8\text{ V}$ to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Units
Output High Voltage	V_{OH}	$V_{DD} = 4.5\text{ V}$ to 5.5 V $I_{OH} = -1\text{ mA}$	$V_{DD} - 1.0$	–	–	V
Output Low Voltage	V_{OL1}	$V_{DD} = 4.5\text{ V}$ to 5.5 V $I_{OL} = 15\text{ mA}$ Ports 4, 5	–	–	2	V
		$V_{DD} = 1.8\text{ V}$ to 5.5 V $I_{OL} = 1.6\text{ mA}$			0.4	
	V_{OL2}	$V_{DD} = 4.5\text{ V}$ to 5.5 V $I_{OL} = 4\text{ mA}$ All out ports except ports 4, 5			2	
		$V_{DD} = 1.8\text{ V}$ to 5.5 V $I_{OL} = 1.6\text{ mA}$			0.6	
Input High Leakage Current	I_{LIH1}	$V_{IN} = V_{DD}$ All input pins except X_{IN} and X_{OUT}	–	–	3	μA
	I_{LIH2}	$V_{IN} = V_{DD}$ X_{IN} and X_{OUT}			20	
Input Low Leakage Current	I_{LIL1}	$V_{IN} = 0\text{ V}$ All input pins except X_{IN} , X_{OUT} and RESET	–	–	–3	μA
	I_{LIL2}	$V_{IN} = 0\text{ V}$ X_{IN} and X_{OUT}			–20	
Output High Leakage Current	I_{LOH}	$V_O = V_{DD}$ All output pins	–	–	3	μA
Output Low Leakage Current	I_{LOL}	$V_O = 0\text{ V}$ All output pins	–	–	–3	μA
Pull-up Resistor	R_{L1}	$V_{DD} = 5\text{ V}$; $V_I = 0\text{ V}$ except RESET	25	50	100	k Ω
		$V_{DD} = 3\text{ V}$	50	100	200	
	R_{L2}	$V_{DD} = 5\text{ V}$; $V_I = 0\text{ V}$; RESET	100	250	400	
		$V_{DD} = 3\text{ V}$	200	500	800	

Table 16-5. D.C. Electrical Characteristics (Concluded) $(T_A = -40\text{ }^\circ\text{C to } +85\text{ }^\circ\text{C, } V_{DD} = 1.8\text{ V to } 5.5\text{ V})$

Parameter	Symbol	Conditions	Min	Typ	Max	Units	
Supply Current (1)	I_{DD1} (DAC on)	Run mode; $V_{DD} = 5.0\text{ V} \pm 10\%$	6.0MHz	-	3.4	10.0	mA
		Crystal oscillator; $C1 = C2 = 22\text{pF}$	4.19MHz		2.7	8.0	
	I_{DD2} (DAC off)	Run mode; $V_{DD} = 5.0\text{ V} \pm 10\%$	6.0MHz	-	2.3	8.0	mA
			Crystal oscillator; $C1 = C2 = 22\text{pF}$		4.19MHz	1.7	
		$V_{DD} = 3\text{ V} \pm 10\%$	6.0MHz		1.1	4.0	
			4.19MHz		0.8	3.0	
	I_{DD3}	Idle mode; $V_{DD} = 5.0\text{ V} \pm 10\%$	6.0MHz	-	0.7	2.5	mA
			Crystal oscillator; $C1 = C2 = 22\text{pF}$		4.19MHz	0.5	
		$V_{DD} = 3\text{ V} \pm 10\%$	6.0MHz		0.3	1.5	
			4.19MHz		0.2	1.0	
I_{DD4}	Stop mode; $V_{DD} = 5.0\text{ V} \pm 10\%$		-	0.2	3.0	μA	
	Stop mode; $V_{DD} = 3.0\text{ V} \pm 10\%$			0.1	2.0		

NOTES:

- D.C. electrical values for supply current (I_{DD1} to I_{DD3}) do not include the current drawn through internal pull-up resistors.
- I_{DD1} typical values are measured when DADATA register value is 055H .

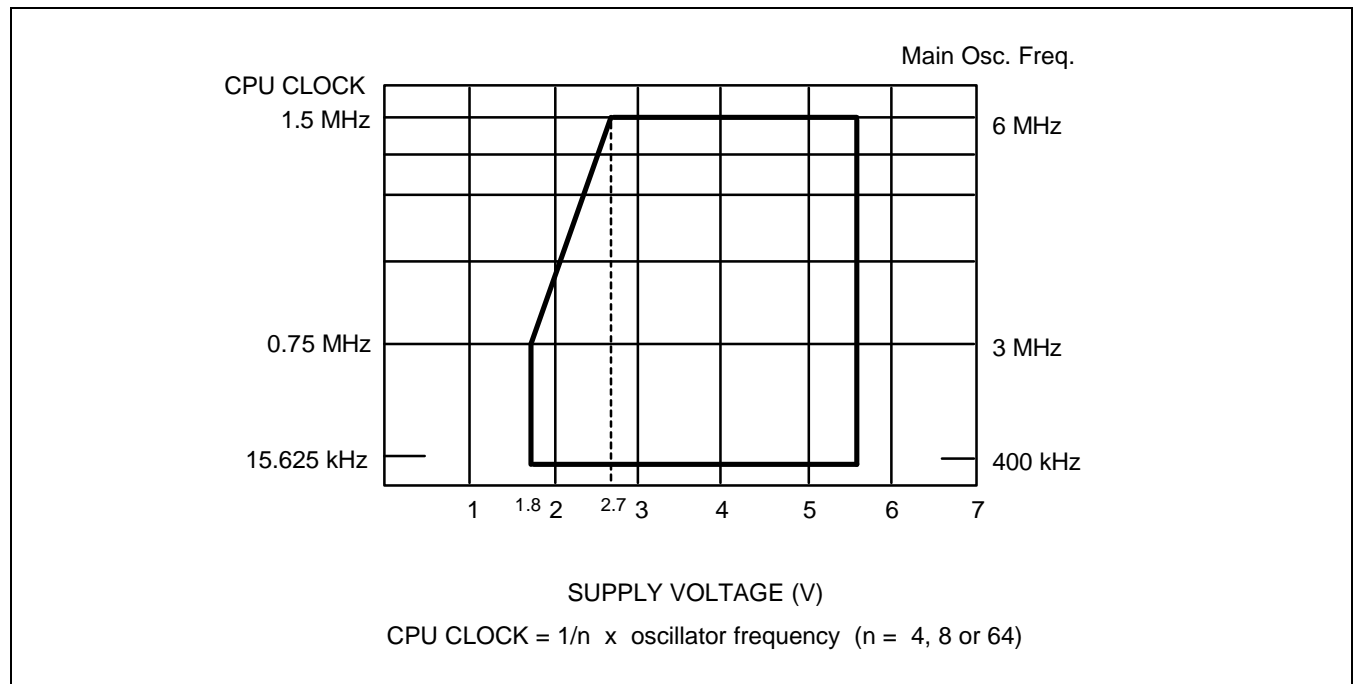
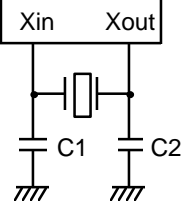
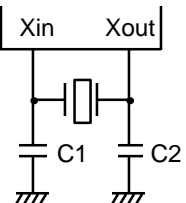
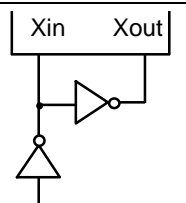
**Figure 16-2. Standard Operating Voltage Range**

Table 16-6. Oscillators Characteristics

(T_A = -40 °C + 85 °C, V_{DD} = 1.8 V to 5.5 V)

Oscillator	Clock Configuration	Parameter	Test Condition	Min	Typ	Max	Units
Ceramic Oscillator		Oscillation frequency ⁽¹⁾	V _{DD} = 2.7 V to 5.5 V	0.4	–	6.0	MHz
			V _{DD} = 1.8 V to 5.5 V	0.4	–	3	
		Stabilization time ⁽²⁾	V _{DD} = 3.0 V	–	–	4	ms
Crystal Oscillator		Oscillation frequency ⁽¹⁾	V _{DD} = 2.7 V to 5.5 V	0.4	–	6.0	MHz
			V _{DD} = 1.8 V to 5.5 V	0.4	–	3	
		Stabilization time ⁽²⁾	V _{DD} = 3.0 V	–	–	10	ms
External Clock		X _{IN} input frequency ⁽¹⁾	V _{DD} = 2.7 V to 5.5 V	0.4	–	6.0	MHz
			V _{DD} = 1.8 V to 5.5 V	0.4	–	3	
		X _{IN} input high and low level width (t _{XH} , t _{XL})	–	83.3	–	1250	ns

NOTES:

- Oscillation frequency and X_{IN} input frequency data are for oscillator characteristics only.
- Stabilization time is the interval required for oscillating stabilization after a power-on occurs, or when stop mode is terminated.

Table 16-7. Input/Output Capacitance

(T_A = 25 °C, V_{DD} = 0 V)

Parameter	Symbol	Condition	Min	Typ	Max	Units
Input Capacitance	C _{IN}	f = 1 MHz; Unmeasured pins are returned to V _{SS}	–	–	15	pF
Output Capacitance	C _{OUT}				15	pF
I/O Capacitance	C _{IO}				15	pF

Table 16-8. Comparator Electrical Characteristics

(T_A = –40 °C to +85 °C, V_{DD} = 3.5 V to 5.5 V, V_{SS} = 0 V)

Parameter	Symbol	Condition	Min	Typ	Max	Units
Resolution	–	–	–	–	8	bits
Absolute Accuracy	–		–3	–	3	LSB
Differential Linearity Error	DLE		–1	–	1	LSB
Setup Time	t _{su}		–	–	5	μs
Output Resistance	R _O		4.5	5	5.5	KΩ

Table 16-9. A.C. Electrical Characteristics

(T_A = –40 °C to +85 °C, V_{DD} = 1.8 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Units
Instruction Cycle Time	t _{CY}	V _{DD} = 2.7 V to 5.5 V	0.67	–	64	μs
		V _{DD} = 1.8 V to 5.5 V	1.33			
TCL0 Input Frequency	f _{TI}	V _{DD} = 2.7 V to 5.5 V	0	–	1.5	MHz
		V _{DD} = 1.8 V to 5.5 V			1	MHz
TCL0 Input High, Low Width	t _{TIH} , t _{TIL}	V _{DD} = 2.7 V to 5.5 V	0.48	–	–	μs
		V _{DD} = 1.8 V to 5.5 V	1.8			
Interrupt Input High, Low Width	t _{INTH} , t _{INTL}	INT0, INT1, KS0–KS1	10	–	–	μs
RESET Input Low Width	t _{RSL}	Input	–	–	10	μs

Table 16-10. RAM Data Retention Supply Voltage in Stop Mode $(T_A = -40\text{ }^\circ\text{C to } +85\text{ }^\circ\text{C})$

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V_{DDDR}	–	1.8	–	5.5	V
Data retention supply current	I_{DDDR}	$V_{\text{DDDR}} = 1.8\text{ V}$	–	0.1	10	μA
Release signal set time	t_{SREL}	–	0	–	–	μs
Oscillator stabilization wait time ⁽¹⁾	t_{WAIT}	Released by RESET	–	$2^{17}/f_x$	–	ms
		Released by interrupt	–	(2)	–	ms

NOTES:

1. During oscillator stabilization wait time, all CPU operations must be stopped to avoid instability during oscillator start-up.
2. Use the basic timer mode register (BMOD) interval timer to delay the execution of CPU instructions during the wait time.

17

DEVELOPMENT TOOLS

OVERVIEW

Samsung provides a powerful and easy-to-use development support system in turnkey form. The development support system is configured with a host system, debugging tools, and support software. For the host system, any standard computer that operates with MS-DOS as its operating system can be used. One type of debugging tool including hardware and software is provided: the sophisticated and powerful in-circuit emulator, SMDS2+, for S3C7, S3C9, S3C8 families of microcontrollers. The SMDS2+ is a new and improved version of SMDS2. Samsung also offers support software that includes debugger, assembler, and a program for setting options.

SHINE

Samsung Host Interface for In-Circuit Emulator, SHINE, is a multi-window based debugger for SMDS2+. SHINE provides pull-down and pop-up menus, mouse support, function/hot keys, and context-sensitive hyper-linked help. It has an advanced, multiple-windowed user interface that emphasizes ease of use. Each window can be sized, moved, scrolled, highlighted, added, or removed completely.

SAMA ASSEMBLER

The Samsung Arrangeable Microcontroller (SAM) Assembler, SAMA, is a universal assembler, and generates object code in standard hexadecimal format. Assembled program code includes the object code that is used for ROM data and required SMDS program control data. To assemble programs, SAMA requires a source file and an auxiliary definition (DEF) file with device specific information.

SASM57

The SASM57 is a relocatable assembler for Samsung's S3C7-series microcontrollers. The SASM57 takes a source file containing assembly language statements and translates into a corresponding source code, object code and comments. The SASM57 supports macros and conditional assembly. It runs on the MS-DOS operating system. It produces the relocatable object code only, so the user should link object file. Object files can be linked with other object files and loaded into memory.

HEX2ROM

HEX2ROM file generates ROM code from HEX file which has been produced by assembler. ROM code must be needed to fabricate a microcontroller which has a mask ROM. When generating the ROM code (.OBJ file) by HEX2ROM, the value 'FF' is filled into the unused ROM area up to the maximum ROM size of the target device automatically.

TARGET BOARDS

Target boards are available for all S3C7-series microcontrollers. All required target system cables and adapters are included with the device-specific target board.

OTPs

One time programmable microcontroller (OTP) for the S3C77544 microcontroller and OTP programmer (Gang) are now available.

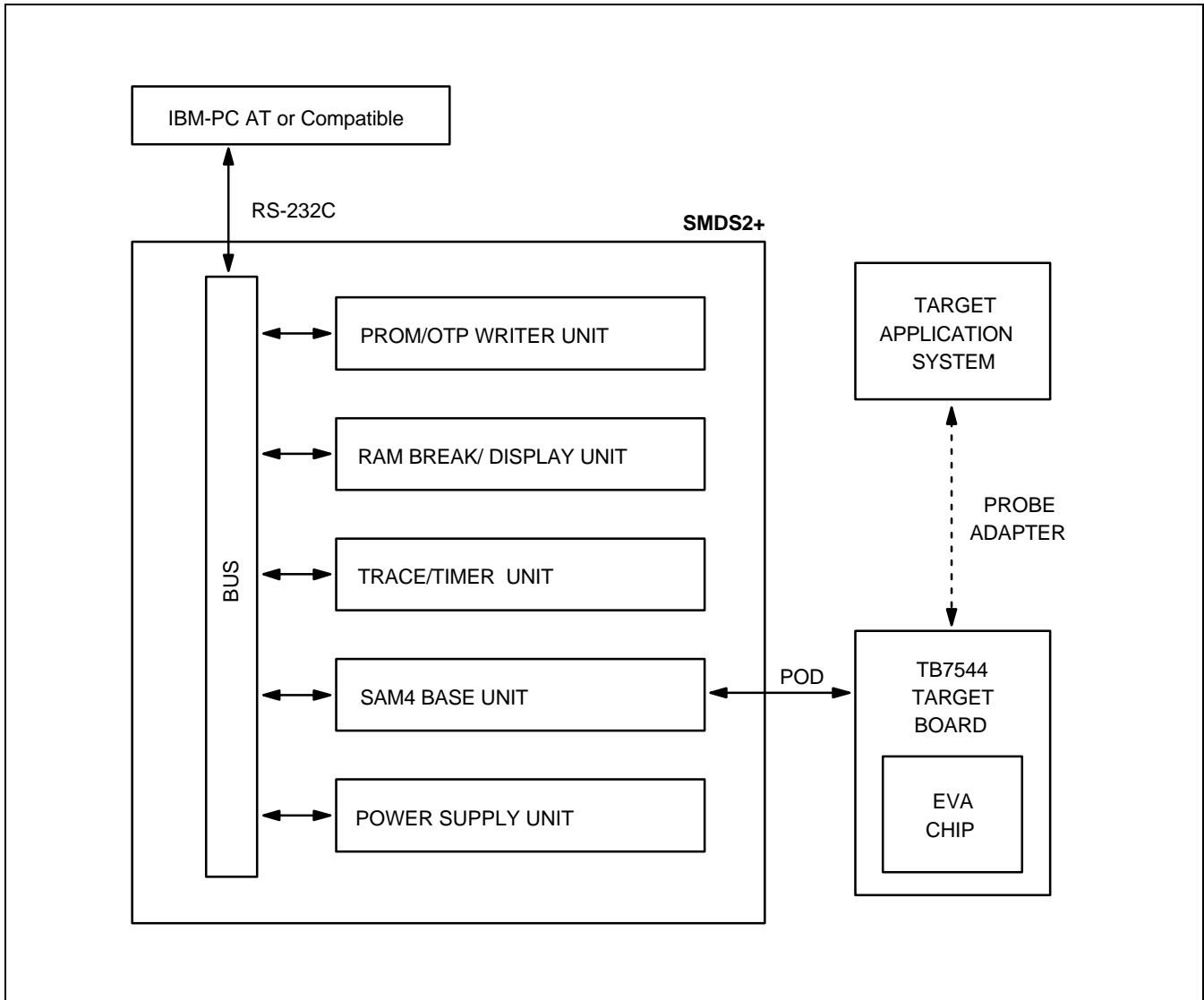


Figure 17-1. SMDS Product Configuration (SMDS2+)

TB7544 TARGET BOARD

The TB7544 target board is used for the S3C77544/P7544 microcontroller. It is supported by the SMDS2+ development system.

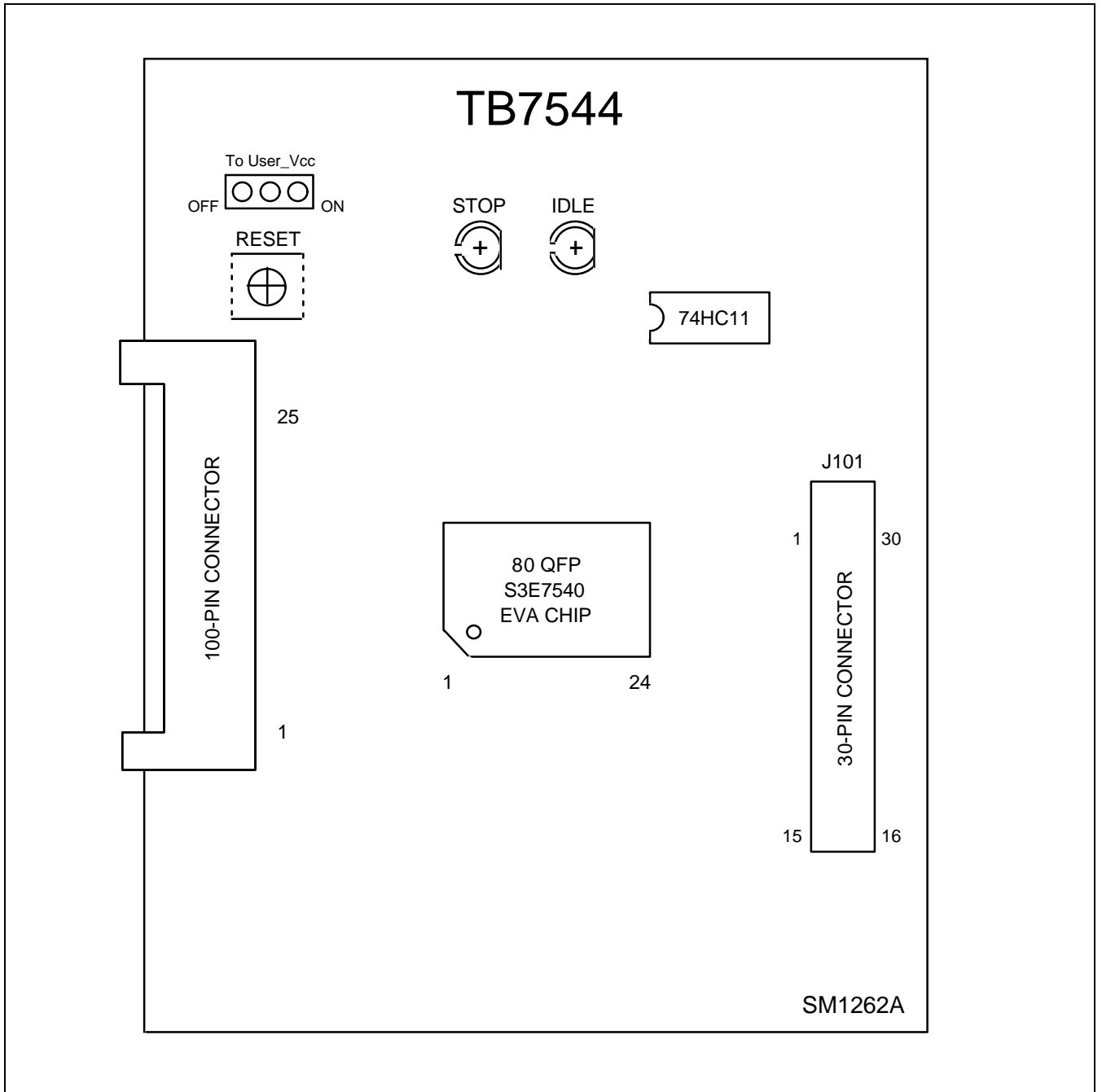
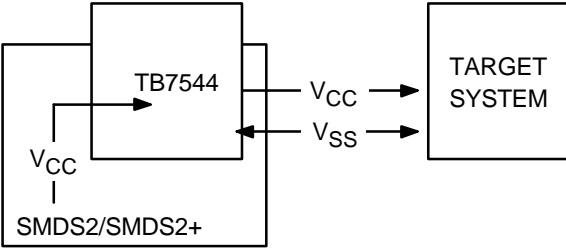
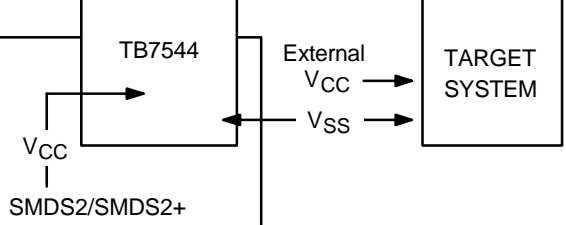


Figure 17-2. TB7544 Target Board Configuration

Table 17-1. Power Selection Settings for TB7544

'To User_Vcc' Settings	Operating Mode	Comments
<p>To User_Vcc OFF <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> ON</p>	 <p>The diagram shows a TB7544 chip. An arrow labeled V_{CC} points from the SMDS2/SMDS2+ block to the chip. Two arrows labeled V_{SS} point from the chip to the TARGET SYSTEM block.</p>	<p>The SMDS2/SMDS2+ supplies V_{CC} to the target board (evaluation chip) and the target system.</p>
<p>To User_Vcc OFF <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/> ON</p>	 <p>The diagram shows a TB7544 chip. An arrow labeled V_{CC} points from the SMDS2/SMDS2+ block to the chip. Two arrows labeled External V_{CC} and V_{SS} point from the TARGET SYSTEM block to the chip.</p>	<p>The SMDS2/SMDS2+ supplies V_{CC} only to the target board (evaluation chip). The target system must have its own power supply.</p>

IDLE LED

This LED is ON when the evaluation chip (S3E7540) is in idle mode.

STOP LED

This LED is ON when the evaluation chip (S3E7540) is in stop mode.

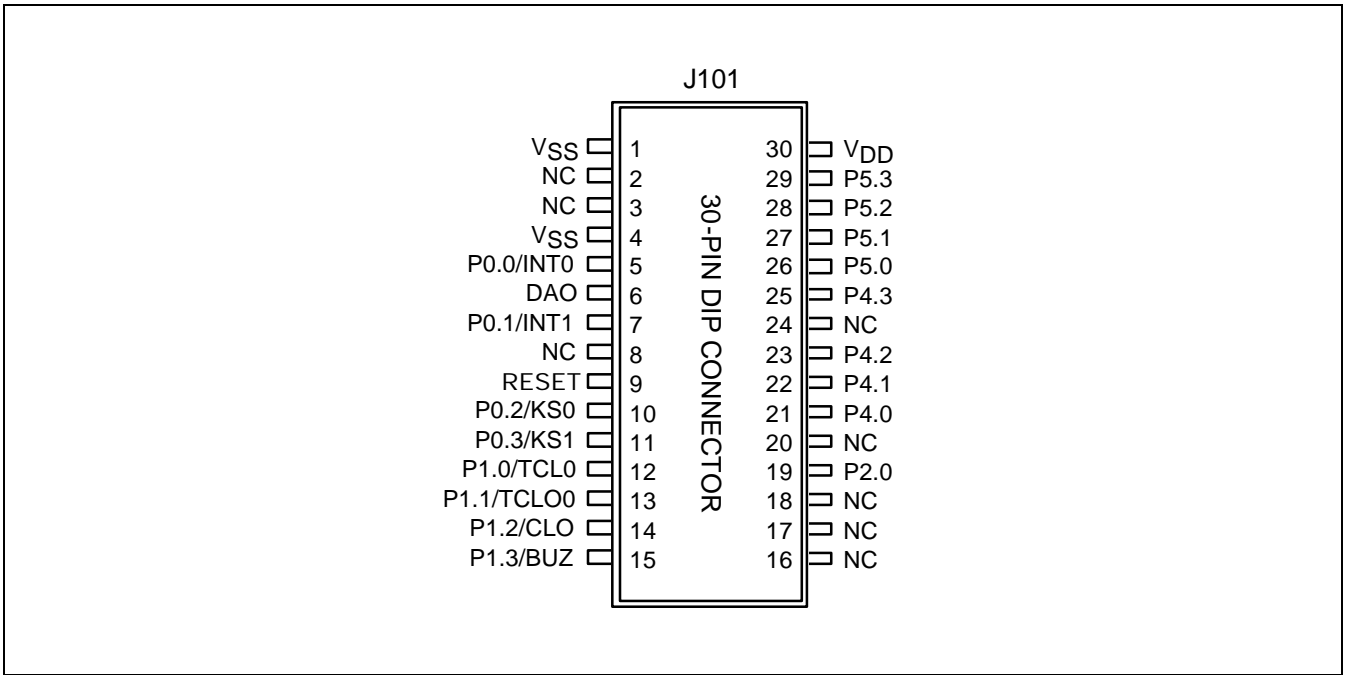


Figure 17-3. 30-Pin Connector for TB7544

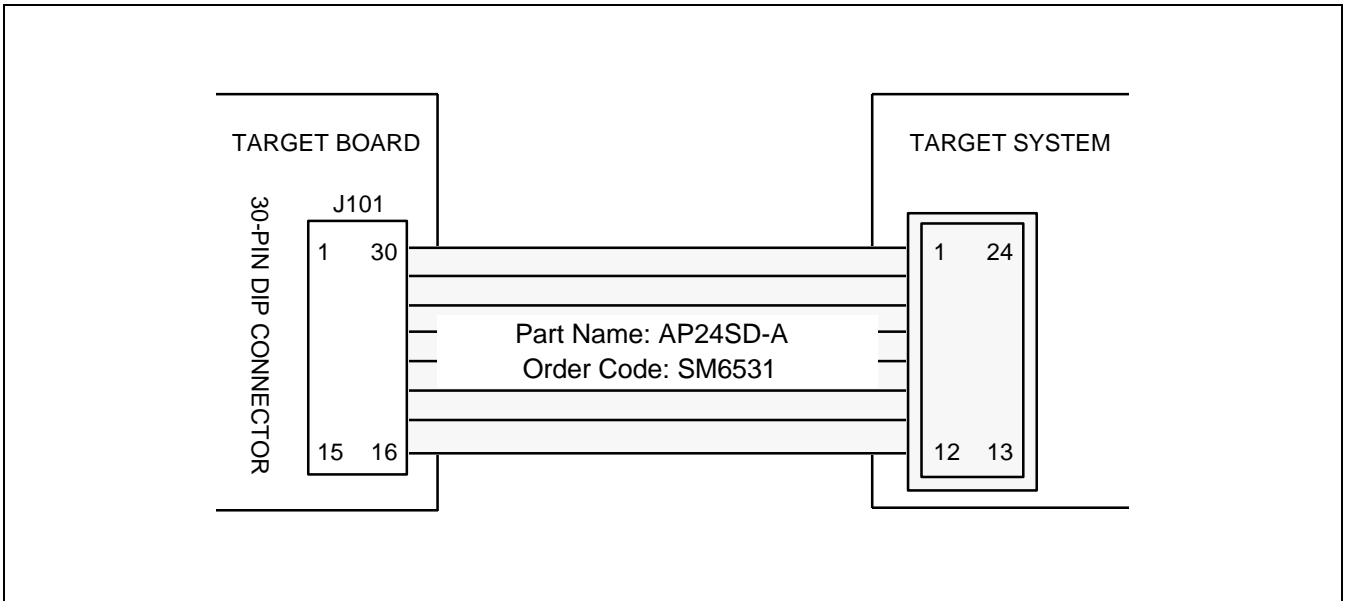


Figure 17-4. TB7544 Adapter Cable for 24-SDIP Package (S3C77544/P7544)