

1 PRODUCT OVERVIEW

SAM87RI PRODUCT FAMILY

Samsung's SAM88RCRI family of 8-bit single-chip CMOS microcontrollers offer fast and efficient CPU, a wide range of integrated peripherals, and support OTP device.

A dual address/data bus architecture and bit- or nibble-configurable I/O ports provide a flexible programming environment for applications with varied memory and I/O requirements. Timer/counters with selectable operating modes are included to support real-time operations.

S3C9664 MICROCONTROLLER

The S3C9664 microcontroller with USB function can be used in a wide range of general purpose applications. It is especially suitable for joystick, game pad controller or mouse and is available in 20-pin DIP, 24-pin SDIP and 20, 24-pin SOP. The S3C9664 single-chip 8-bit microcontroller is fabricated using an advanced CMOS process. It is built around the powerful SAM88RCRI CPU core.

Stop and Idle power-down modes were implemented to reduce power consumption. To increase on-chip register space, the size of the internal register file was logically expanded. The S3C9664 has 4 K bytes of program memory on-chip, and 208 bytes of RAM including 16 bytes of working register. Using the SAM88RCRI design approach, the following peripherals were integrated with the SAM88RCRI core:

- Two configurable I/O ports (14 I/O pins on 20 pin package, 18 I/O pins on 24pin package)
- Analog-to-digital converter with six input channel and 10-bit resolution
- One 8-bit basic timer for watchdog function
- One 8 bit timer/counter with three operating modes (Timer 0)
- One 8 bit timer (Timer 1)

OTP

The S3C9664 microcontroller is also available in OTP (One Time Programmable) version, S3P9664. S3P9664 microcontroller has an on-chip 4 K byte one-time-programmable EPROM instead of masked ROM. The S3P9664 is compatible to S3C9664, both in function and in pin configuration.

FEATURES

CPU

- SAM88RCRI CPU core

Memory

- 4-K byte internal program memory
- 208-byte general purpose register area
- 16 bytes of working register

Instruction Set

- 41 instructions
- IDLE and STOP instructions added for power-down modes

Instruction Execution Time

- 0.66 μ s at 6 MHz f_{OSC}

Interrupts

- 28 interrupt sources and one vector (24 pins)
- 24 interrupt sources and one vector (20 pin)
- One interrupt level

General I/O

- Three I/O ports (total 18 I/O pins at 24 SOP/SDIP)
- Three I/O ports (total 14 I/O pins at 20 SOP/DIP)

Timer/Counter

- One 8-bit basic timer for watchdog function
- One 8 bit timer/counter with three operating modes(Match, capture, PWM)
- One 8-bit Timer

USB

- Compatible to USB low speed (1.5 Mbps) device 1.1 specification.
- Serial bus interface engine (SIE)
 - Packet decoding/generation
 - CRC generation and checking
 - NRZI encoding/decoding and bit-stuffing
- Two 8-byte receive/transmit USB buffer

A/D Converter

- Six analog input pins
- 10-bit conversion resolution

Low Voltage Reset

- Low voltage reset
- Power on Reset

Sub Oscillator

- Internal RC sub oscillator
- Auto interrupt wake-up

Oscillator Frequency

- 6 MHz crystal/ceramic oscillator
- External clock source (6 MHz)

Operating Temperature Range

- -40°C to $+85^{\circ}\text{C}$

Operating Voltage Range

- 4.0 V to 5.25 V

Package Types

- 24-pin SOP/SDIP
- 20-pin SOP/DIP

BLOCK DIAGRAM

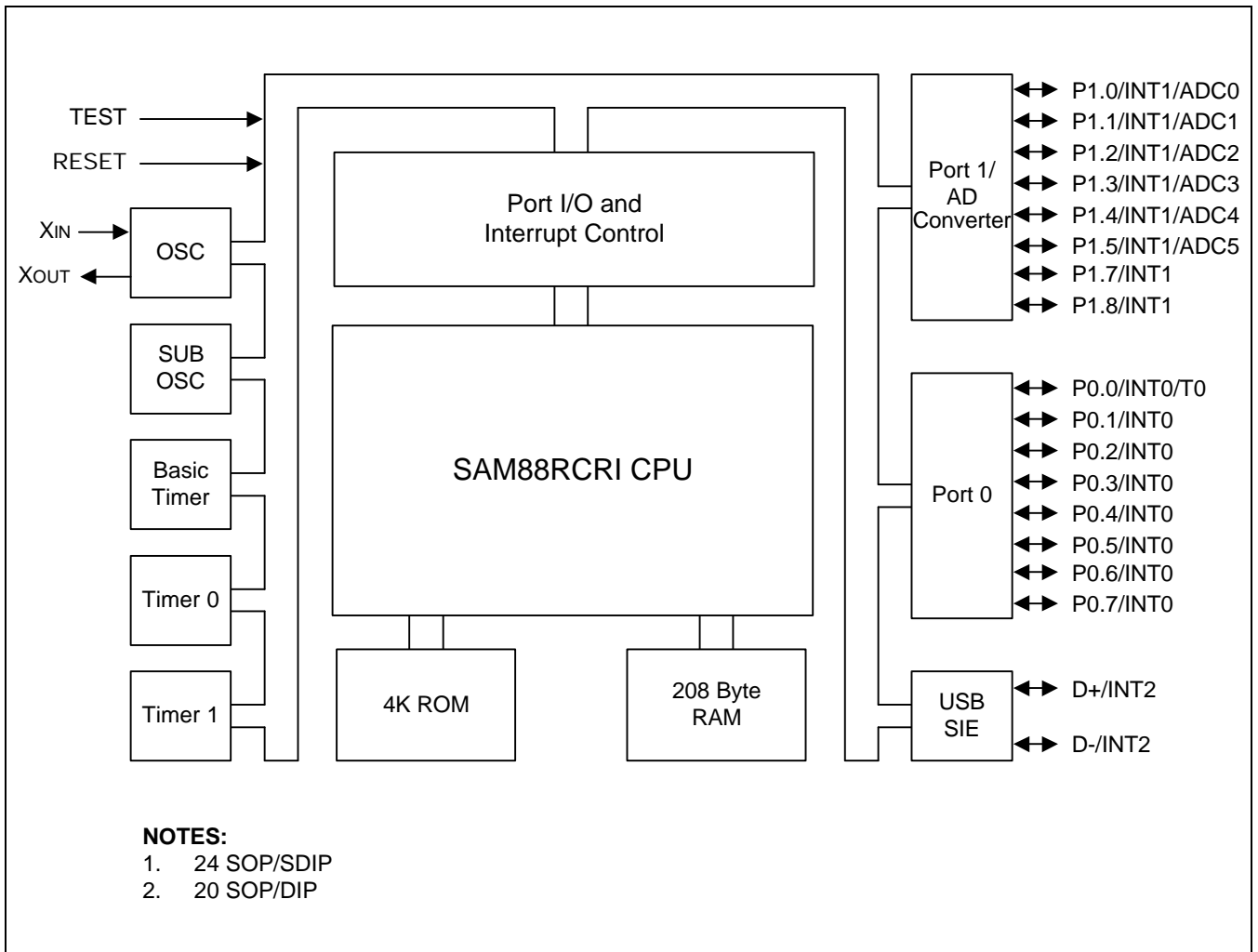


Figure 1-1. Block Diagram

PIN ASSIGNMENTS

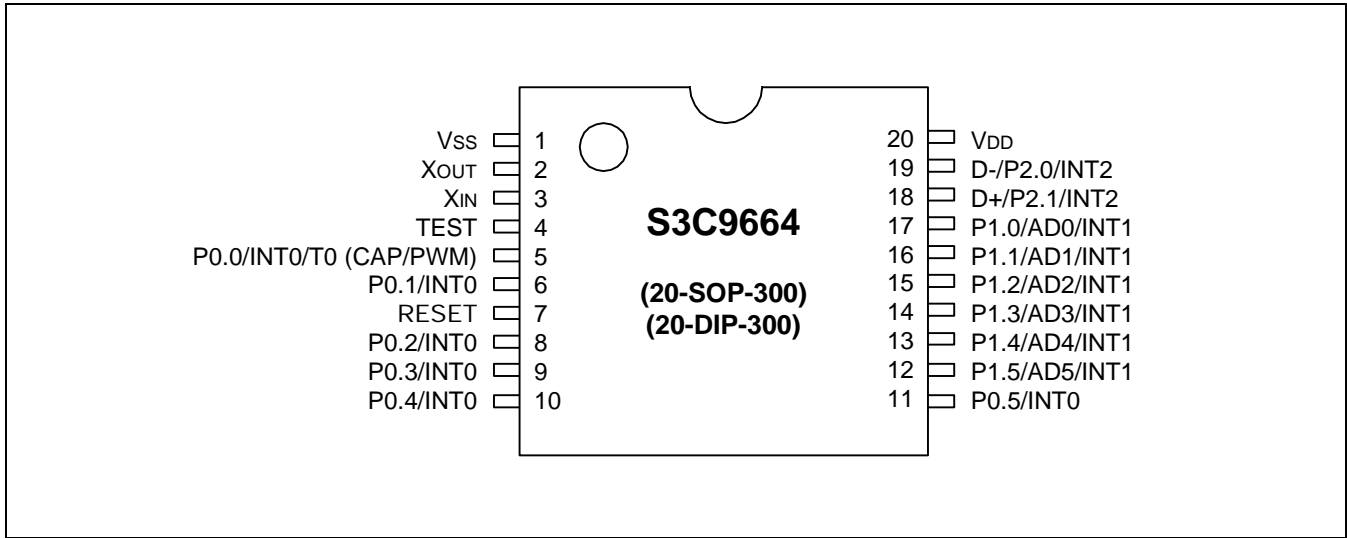


Figure 1-2. Pin Assignment (20 Pin)

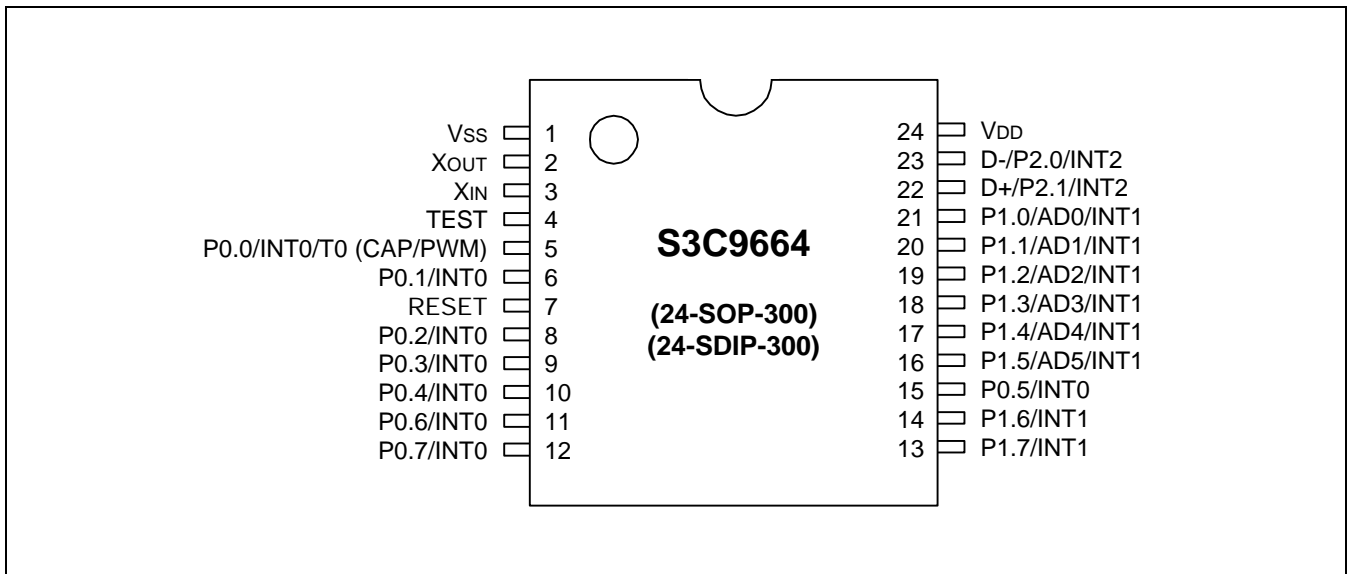


Figure 1-3. Pin Assignment (24 Pin)

Table 1-1. Pin Descriptions

Pin Name	In/Out	Pin Description	Pin Type	Pin Numbers	Share Pins
P0.0–P0.7	I/O	Bit-programmable I/O port for Schmitt trigger input , push-pull output and N-Ch open drain output. Pull-up/ pull-down resistors are assignable by software. Port1 pins can also be used as external interrupt.	G	5,6,8–11 (5,6,8–12,15)	T0, INT0
P1.0–P1.5	I/O	Bit-programmable I/O port for Schmitt trigger input, Schmitt trigger input with pull-up and N-Ch open drain output. Port1 pins can also be used as A/D converter Channel.	F	12–17 (16–21)	AD0–5 INT1
P1.6–P1.7	I/O	Bit-programmable I/O port for Schmitt trigger input , Schmitt trigger input with pull-up and N-Ch open drain output and Push-pull output.	E	(13–14)	INT1
P2.0/D– – P2.1/D+	I/O	Bit-programmable I/O port for Schmitt trigger input , Schmitt trigger input with pull-up and N-Ch open drain output and Push-pull output. Port 2 can be individually configured as external interrupt inputs. Also it can be configured as an USB ports	E	18–19 (22–23)	INT2
X _{IN} , X _{OUT}	–	Crystal/ceramic oscillator signal for system clock.	–	2–3 (2–3)	–
RESET	I	System reset signal input pin.	B	7 (7)	–
TEST	I	Test signal input pin(for factory use only; muse be connected to V _{SS})	–	4 (4)	–
V _{DD} , V _{SS}	–	Voltage input pin and ground	–	1,20 (1,24)	–
T0	I/O	Timer 0 capture input or PWM output pin	G	6 (6)	P0.0
INT0	I	External interrupt input	G	5,6,8–11 (5,6,8–12,15)	P0.0–P0.7
INT1	I	External interrupt input	F,E	12–17 (13–14, 16–21)	P1.0–P1.7
INT2	I	External interrupt input	E	18–19	P2.0–P2.1
AD0–AD5	I	A/D converter input	F	12–17 (16–21)	P1.0–P1.5

NOTE: Pin numbers show in parentheses "()" are for the 24-pin package

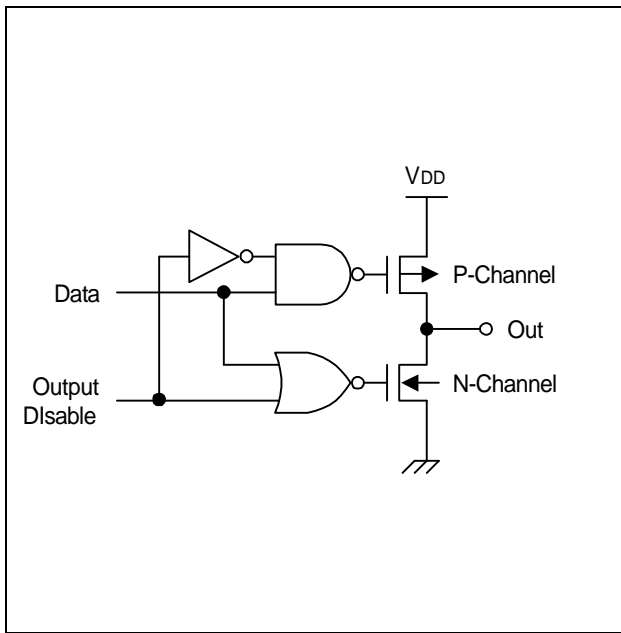


Figure 1-4. Pin Circuit Type C

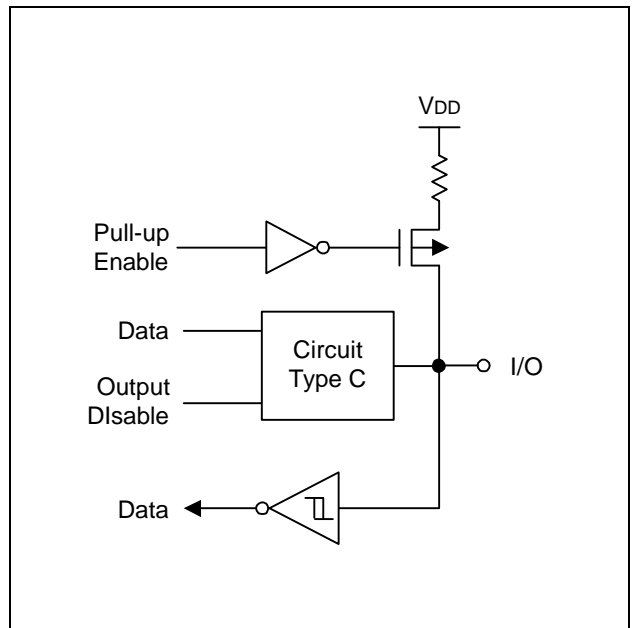


Figure 1-5. Pin Circuit Type D

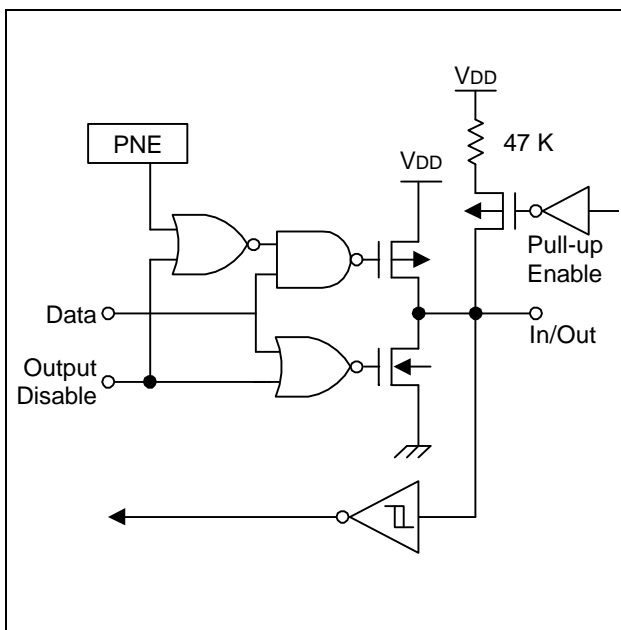


Figure 1-6. Pin Circuit Type E

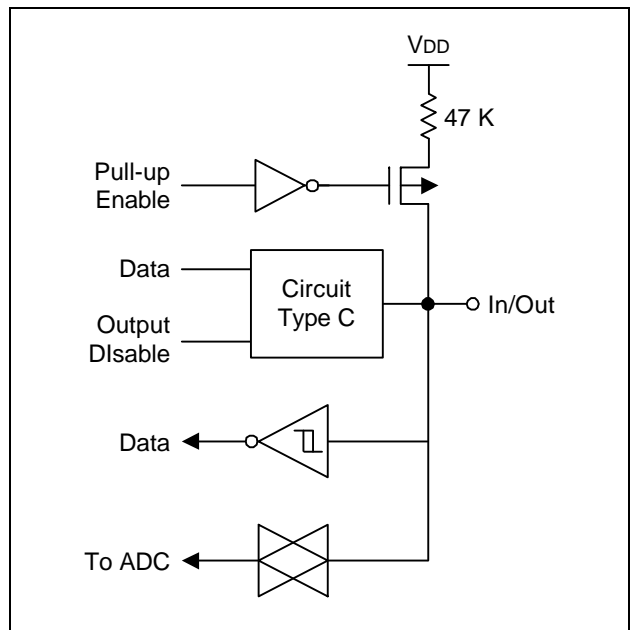


Figure 1-7. Pin Circuit Type F

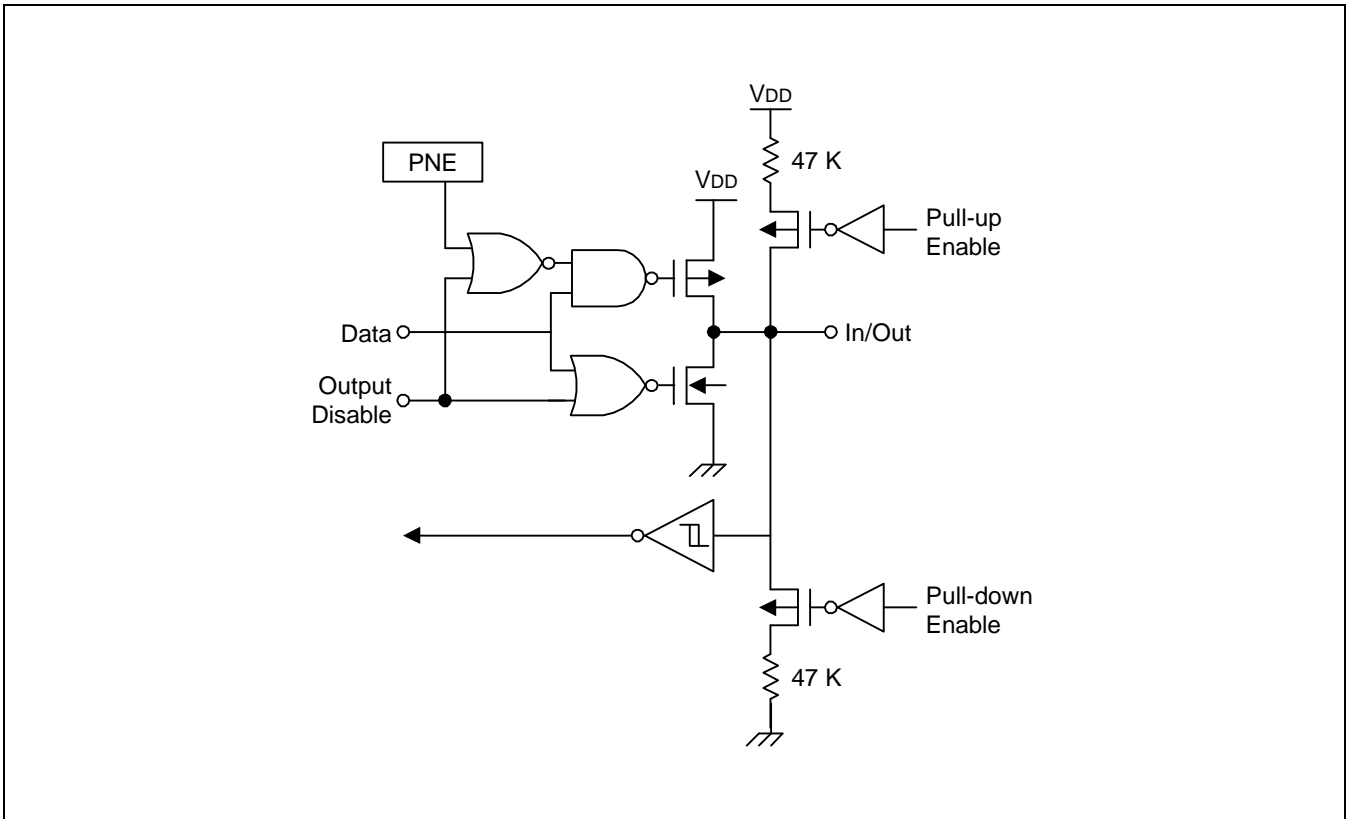


Figure 1-8. Pin Circuit Type G

2 ADDRESS SPACES

OVERVIEW

The S3C9664 microcontroller has two kinds of address space:

- Program memory (ROM)
- Internal register file

A 13-bit address bus supports both program memory. Special instructions and related internal logic determine when the 13-bit bus carries addresses for program memory. A separate 8-bit register bus carries addresses and data between the CPU and the internal register file.

The S3C9664 has 4 K bytes of mask-programmable program memory on-chip. The S3C9664 microcontroller has 192 bytes general-purpose registers in its internal register file. Forty-nine bytes in the register file are mapped for system and peripheral control functions.

PROGRAM MEMORY (ROM)

NORMAL OPERATING MODE (INTERNAL ROM)

The S3C9664 has 4 K bytes of internal mask-programmable program memory. The first 2 bytes of the ROM (0000H–0001H) are an interrupt vector address. The program reset address in the ROM is 0100H.

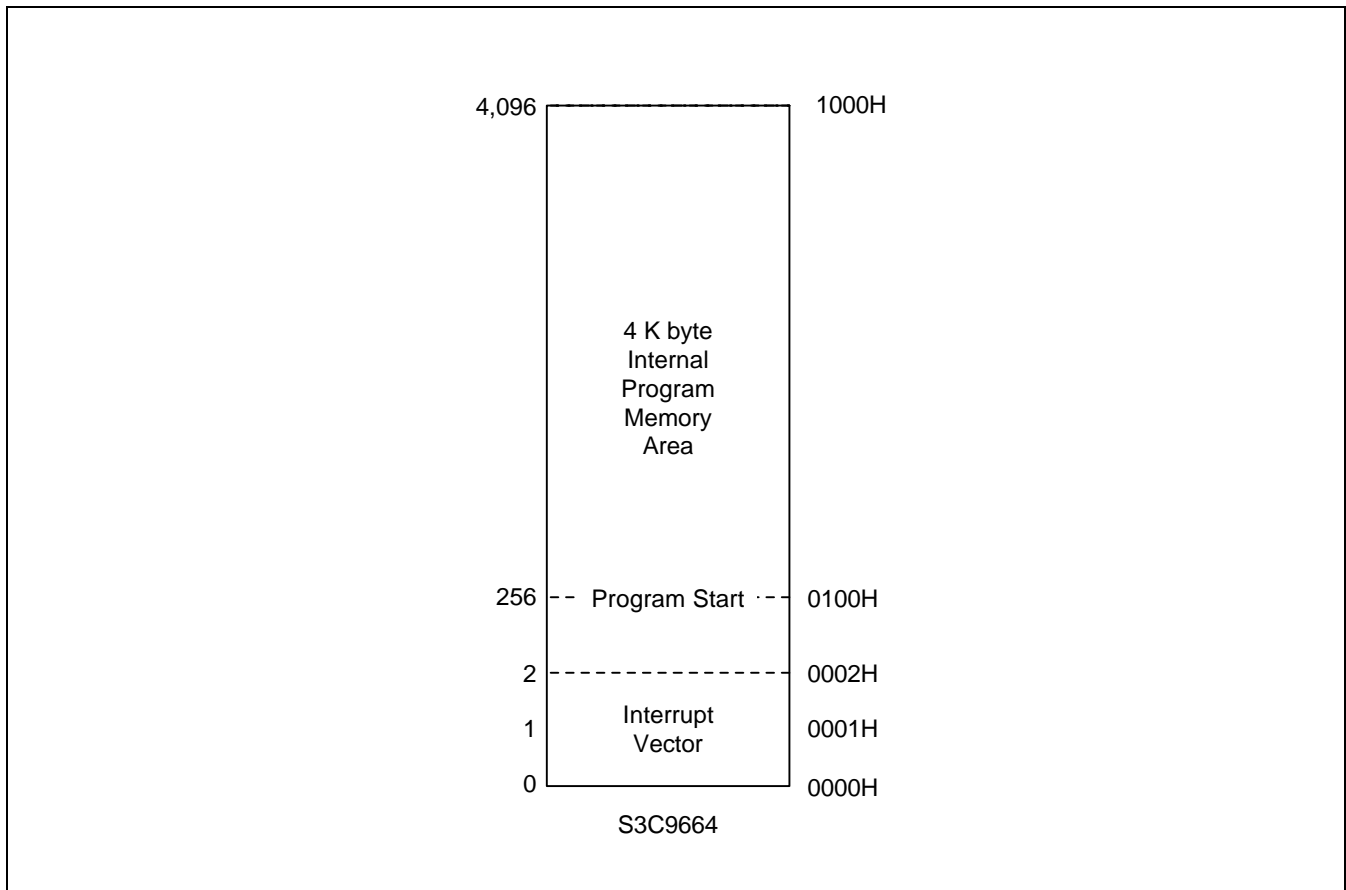


Figure 2-1. S3C9664 Program Memory Address Space

REGISTER ARCHITECTURE

The upper 64 bytes (page 0) and the expanded one byte (FFH, page 1) of the S3C9664's internal register file are addressed as working registers, system control registers and peripheral control registers. The lower 192 bytes of internal register file (00H–BFH) is called the general purpose register space.

For many SAM88RCRI microcontrollers, the addressable area of the internal register file is further expanded by the additional of one or more register pages at general purpose register space (00H–BFH). This register file expansion is not implemented in the S3C9664.

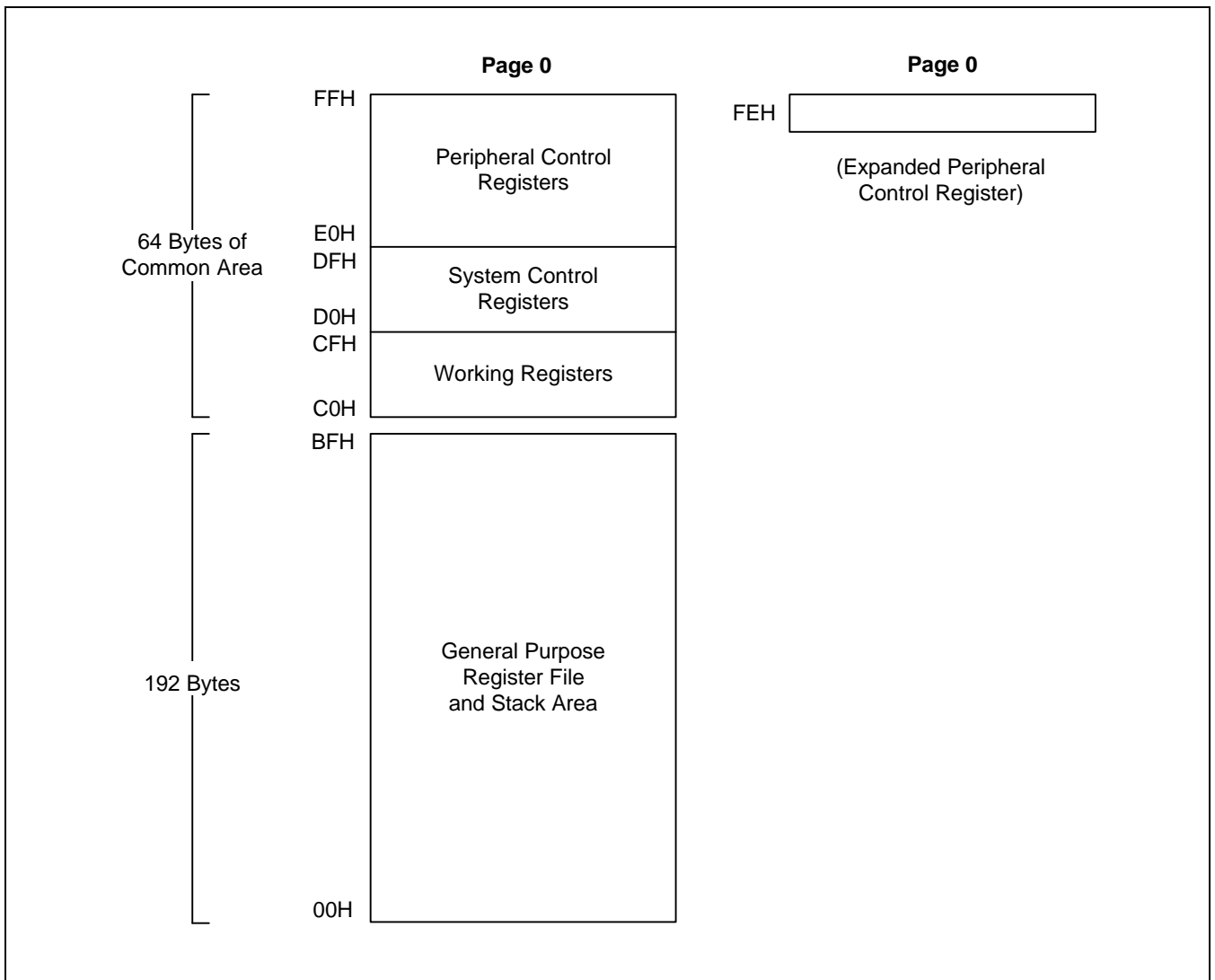


Figure 2-2. Internal Register File Organization

COMMON WORKING REGISTER AREA (C0H–CFH)

The SAM88RCRI register architecture provides an efficient method of working register addressing that takes full advantage of shorter instruction formats to reduce execution time.

This 16-byte address range is called common area. That is, locations in this area can be used as working registers by operations that address any location on any page in the register file. Typically, these working registers serve as temporary buffers for data operations between different pages. However, because the S3C9664 uses only page 0, you can use the common area for any internal data operation.

The Register (R) addressing mode can be used to access this area

Registers are addressed either as a single 8-bit register or as a paired 16-bit register. In 16-bit register pairs, the address of the first 8-bit register is always an even number and the address of the next register is an odd number. The most significant byte of the 16-bit data is always stored in the even-numbered register; the least significant byte is always stored in the next (+ 1) odd-numbered register.

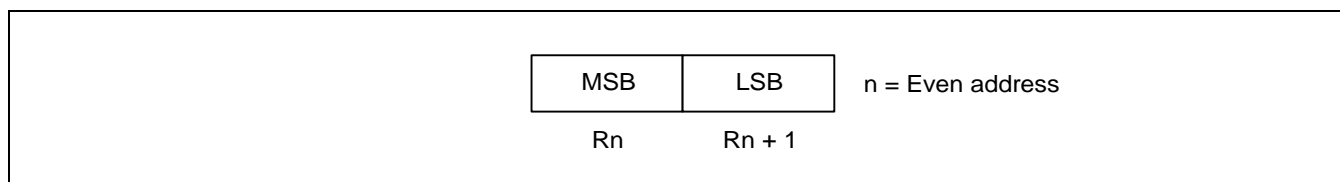


Figure 2-3. 16-Bit Register Pairs

PROGRAMMING TIP — Addressing the Common Working Register Area

As the following examples show, you should access working registers in the common area, locations C0H–CFH, using working register addressing mode only.

- Examples:
1. LD 0C2H,40H ; Invalid addressing mode!
Use working register addressing instead:
LD R2,40H ; R2 (C2H) ← the value in location 40H
 2. ADD 0C3H,#45H ; Invalid addressing mode!
Use working register addressing instead:
ADD R3,#45H ; R3 (C3H) ← R3 + 45H

SYSTEM STACK

S3C9-series microcontrollers use the system stack for subroutine calls and returns and to store data. The PUSH and POP instructions are used to control system stack operations. The S3C9664 architecture supports stack operations in the internal register file.

STACK OPERATIONS

Return addresses for procedure calls and interrupts and data are stored on the stack. The contents of the PC are saved to stack by a CALL instruction and restored by the RET instruction. When an interrupt occurs, the contents of the PC and the FLAGS register are pushed to the stack. The IRET instruction then pops these values back to their original locations. The stack address is always decremented before a push operation and incremented after a pop operation. The stack pointer (SP) always points to the stack frame stored on the top of the stack, as shown in Figure 2-4.

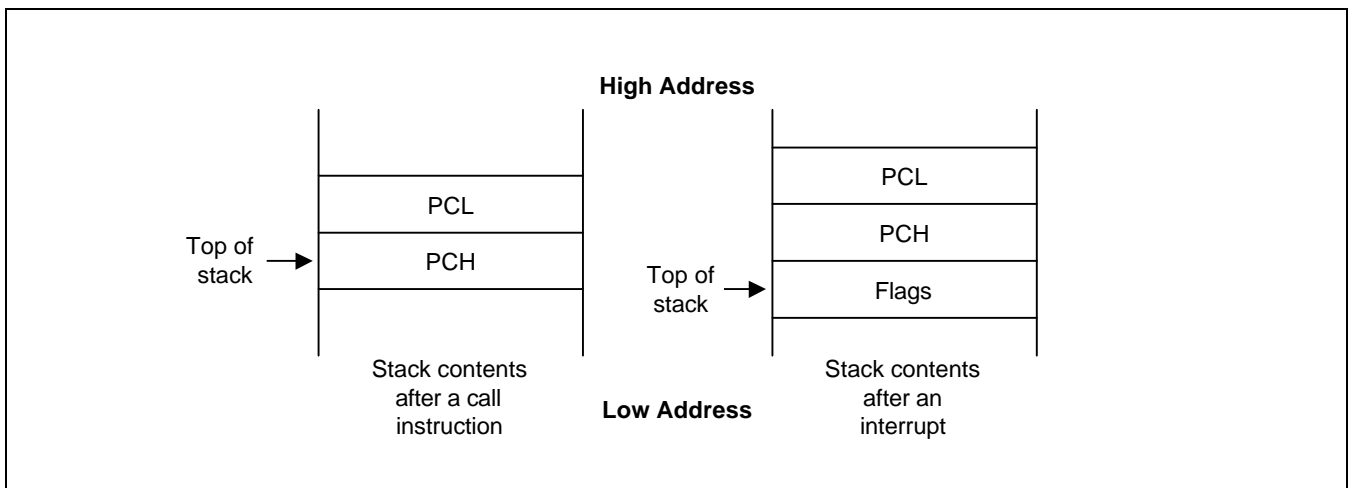


Figure 2-4. Stack Operations

STACK POINTER (SP)

Register location D9H contains the 8-bit stack pointer (SP) that is used for system stack operations. After a reset, the SP value is undetermined.

Because only internal memory space is implemented in the KS86C6504/P6504, the SP must be initialized to an 8-bit value in the range 00H–BFH.

NOTE

In case a Stack Pointer is initialized to 00H, it is decreased to FFH when stack operation starts. This means that a Stack Pointer access invalid stack area.

 PROGRAMMING TIP — Standard Stack Operations Using PUSH and POP

The following example shows you how to perform stack operations in the internal register file using PUSH and POP instructions:

```
LD      SP,#0C0H      ; SP ← C0H (Normally, the SP is set to 0C0H by the
                      ; initialization routine)
.
.
PUSH   SYM            ; Stack address 0BFH ← SYM
PUSH   20H            ; Stack address 0BDH ← 20H
PUSH   R3             ; Stack address 0BCH ← R3
.
.
POP    R3             ; R3 ← Stack address 0BCH
POP    20H            ; 20H ← Stack address 0BDH
POP    SYM            ; SYM ← Stack address 0BFH
```

3 ADDRESSING MODES

OVERVIEW

Instructions that are stored in program memory are fetched for execution using the program counter. Instructions indicate the operation to be performed and the data to be operated on. Addressing mode is the method used to determine the location of the data operand. The operands specified in SAM88RCRI instructions may be condition codes, immediate data, or a location in the register file, program memory, or data memory.

The SAM88RCRI instruction set supports six explicit addressing modes. Not all of these addressing modes are available for each instruction. The addressing modes and their symbols are as follows:

- Register (R)
- Indirect Register (IR)
- Indexed (X)
- Direct Address (DA)
- Relative Address (RA)
- Immediate (IM)

REGISTER ADDRESSING MODE (R)

In Register addressing mode, the operand is the content of a specified register (see Figure 3-1). Working register addressing differs from Register addressing because it uses a 16-byte working register space in the register file and a 4-bit register within that space (see Figure 3-2).

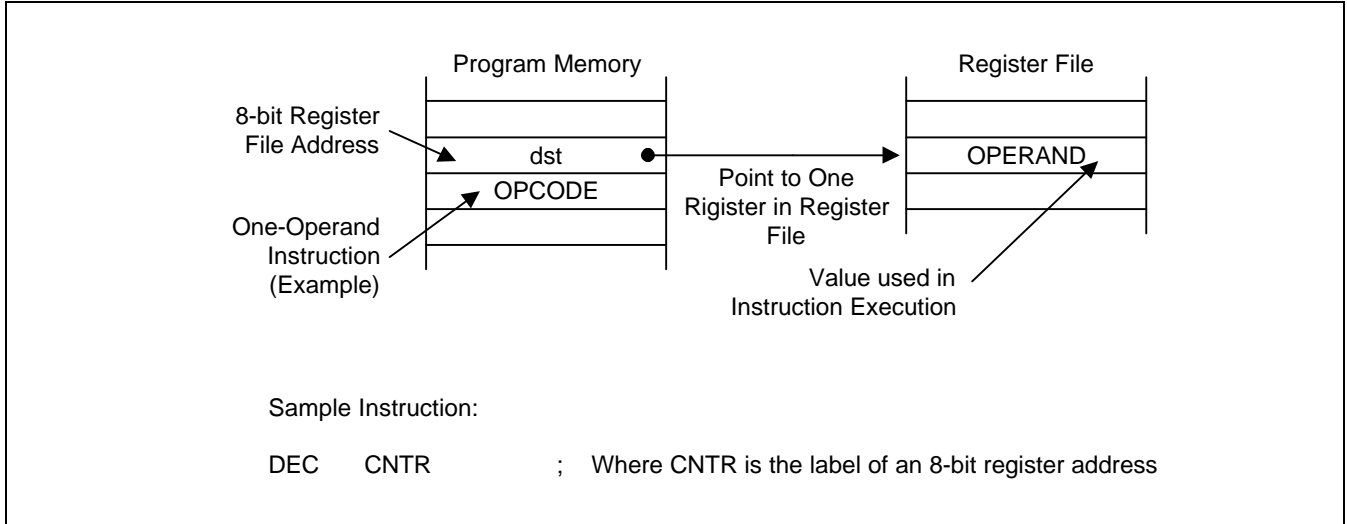


Figure 3-1. Register Addressing

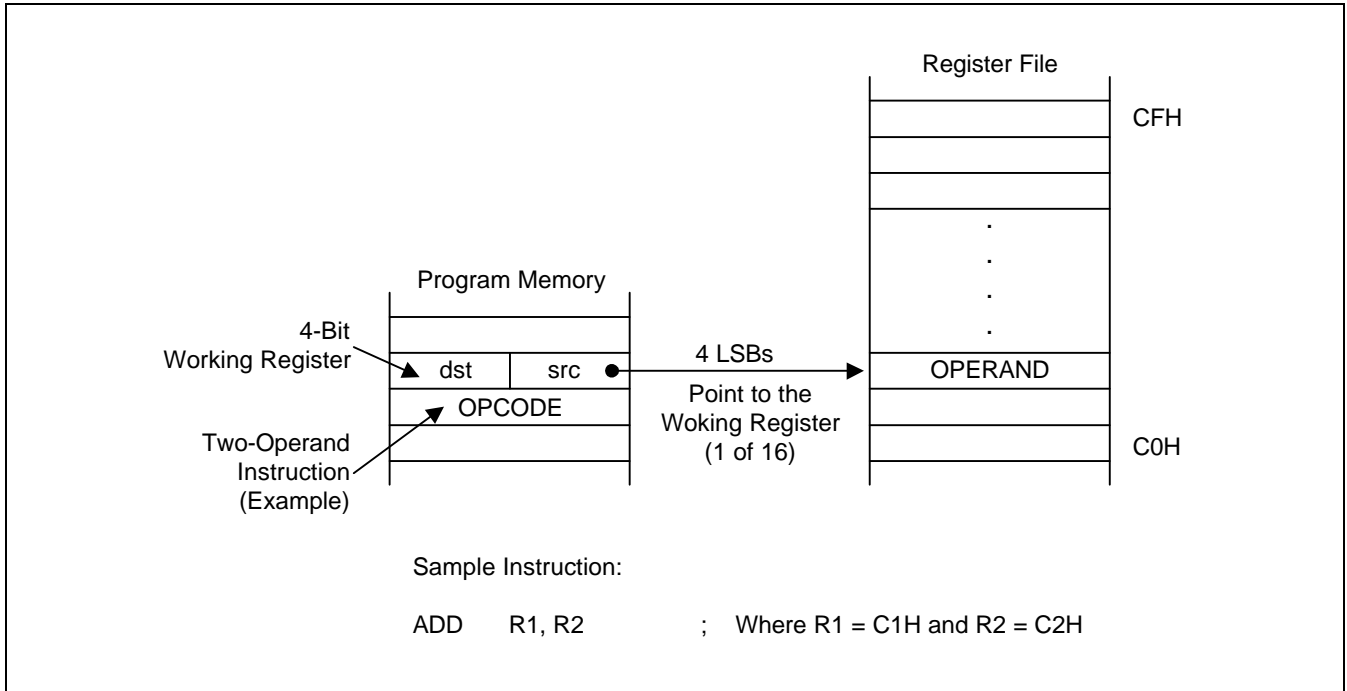


Figure 3-2. Working Register Addressing

INDIRECT REGISTER ADDRESSING MODE (IR)

In Indirect Register (IR) addressing mode, the content of the specified register or register pair is the address of the operand. Depending on the instruction used, the actual address may point to a register in the register file, to program memory (ROM), or to an external memory space (see Figures 3-3 through 3-6).

You can use any 8-bit register to indirectly address another register. Any 16-bit register pair can be used to indirectly address another memory location.

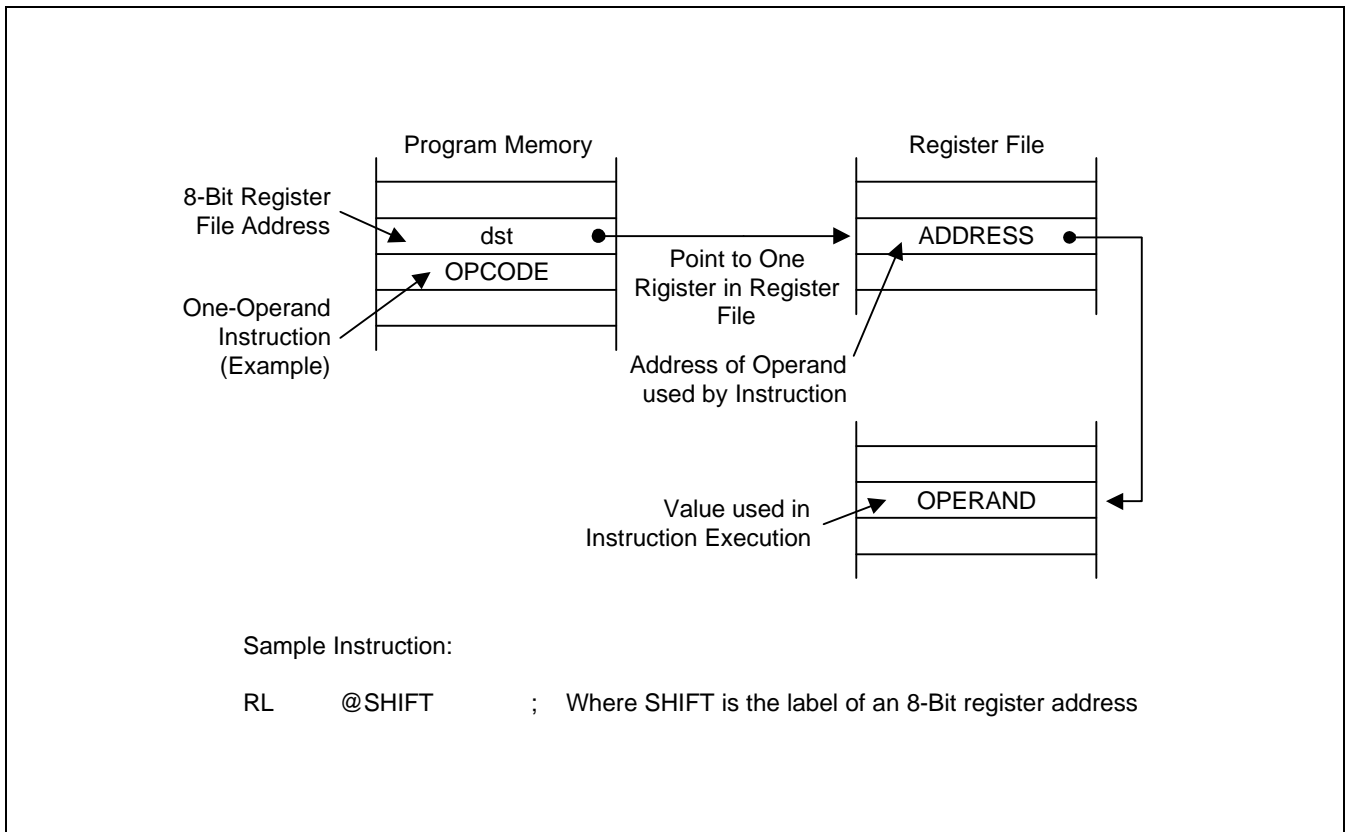


Figure 3-3. Indirect Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Continued)

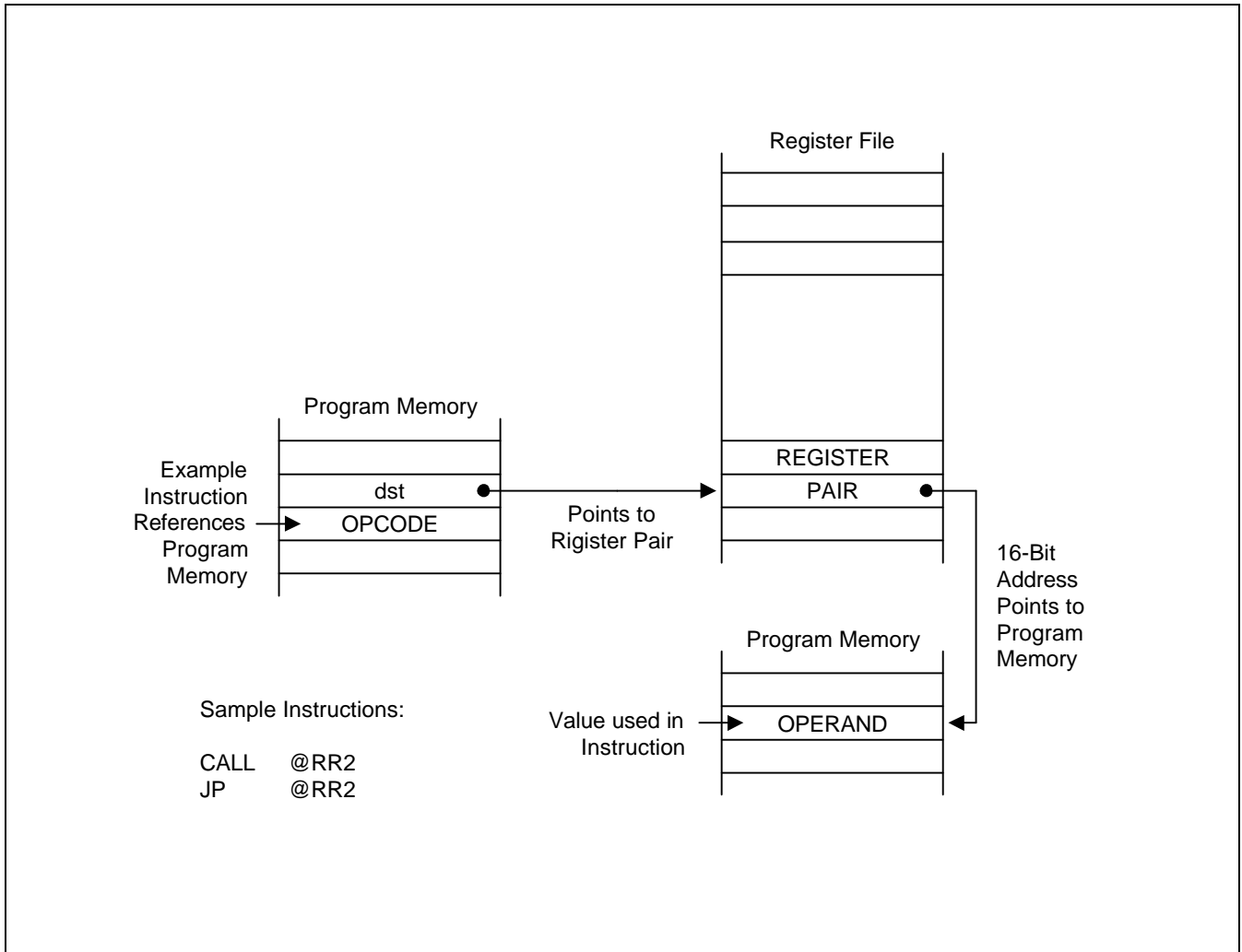


Figure 3-4. Indirect Register Addressing to Program Memory

INDIRECT REGISTER ADDRESSING MODE (Continued)

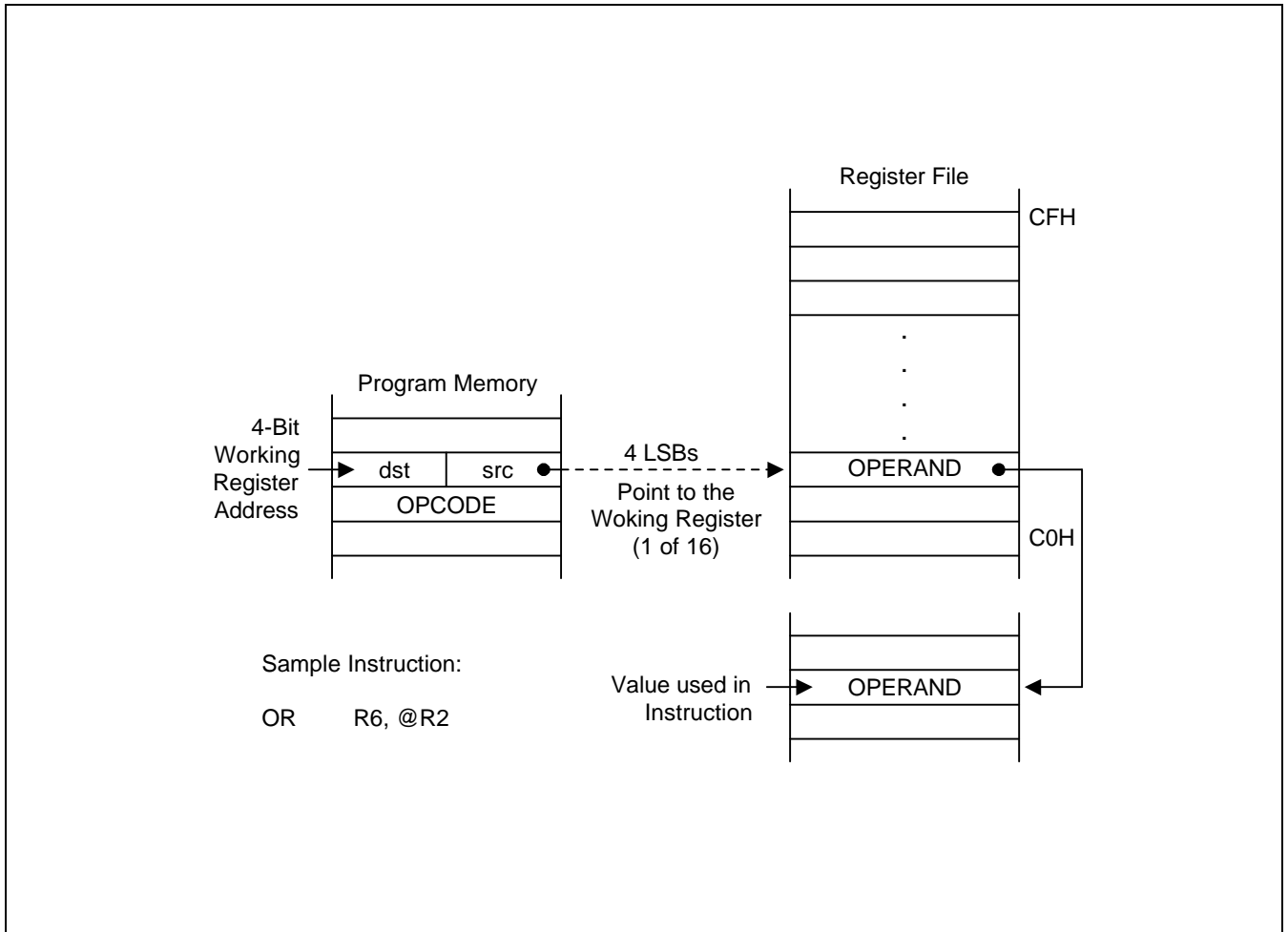


Figure 3-5. Indirect Working Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Concluded)

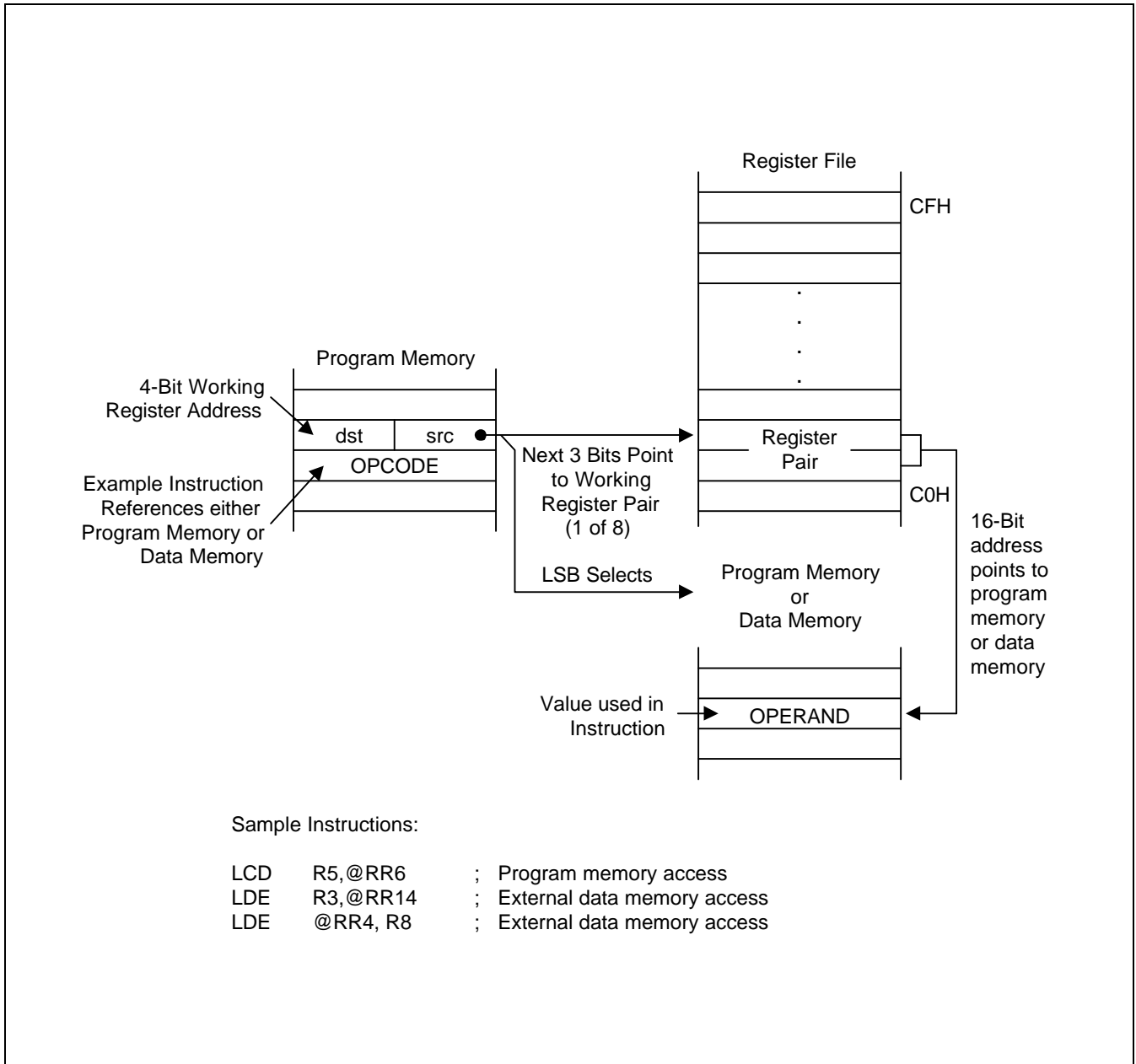


Figure 3-6. Indirect Working Register Addressing to Program or Data Memory

INDEXED ADDRESSING MODE (X)

Indexed (X) addressing mode adds an offset value to a base address during instruction execution in order to calculate the effective operand address (see Figure 3-7). You can use Indexed addressing mode to access locations in the internal register file or in external memory.

In short offset Indexed addressing mode, the 8-bit displacement is treated as a signed integer in the range of -128 to $+127$. This applies to external memory accesses only (see Figure 3-8).

For register file addressing, an 8-bit base address provided by the instruction is added to an 8-bit offset contained in a working register. For external memory accesses, the base address is stored in the working register pair designated in the instruction. The 8-bit or 16-bit offset given in the instruction is then added to the base address (see Figure 3-9).

The only instruction that supports Indexed addressing mode for the internal register file is the Load instruction (LD). The LDC and LDE instructions support Indexed addressing mode for internal program memory, external program memory, and for external data memory, when implemented.

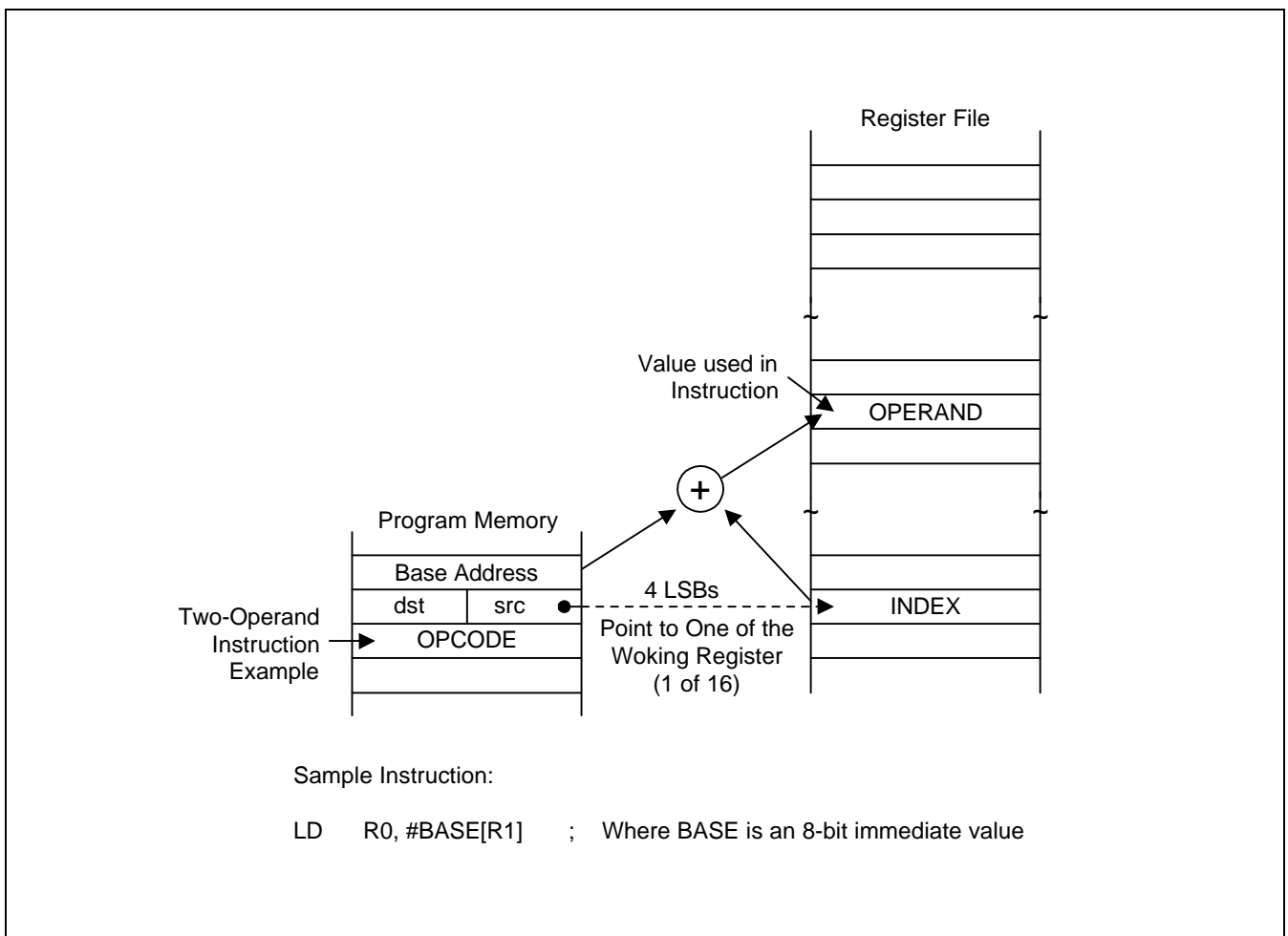


Figure 3-7. Indexed Addressing to Register File

INDEXED ADDRESSING MODE (Continued)

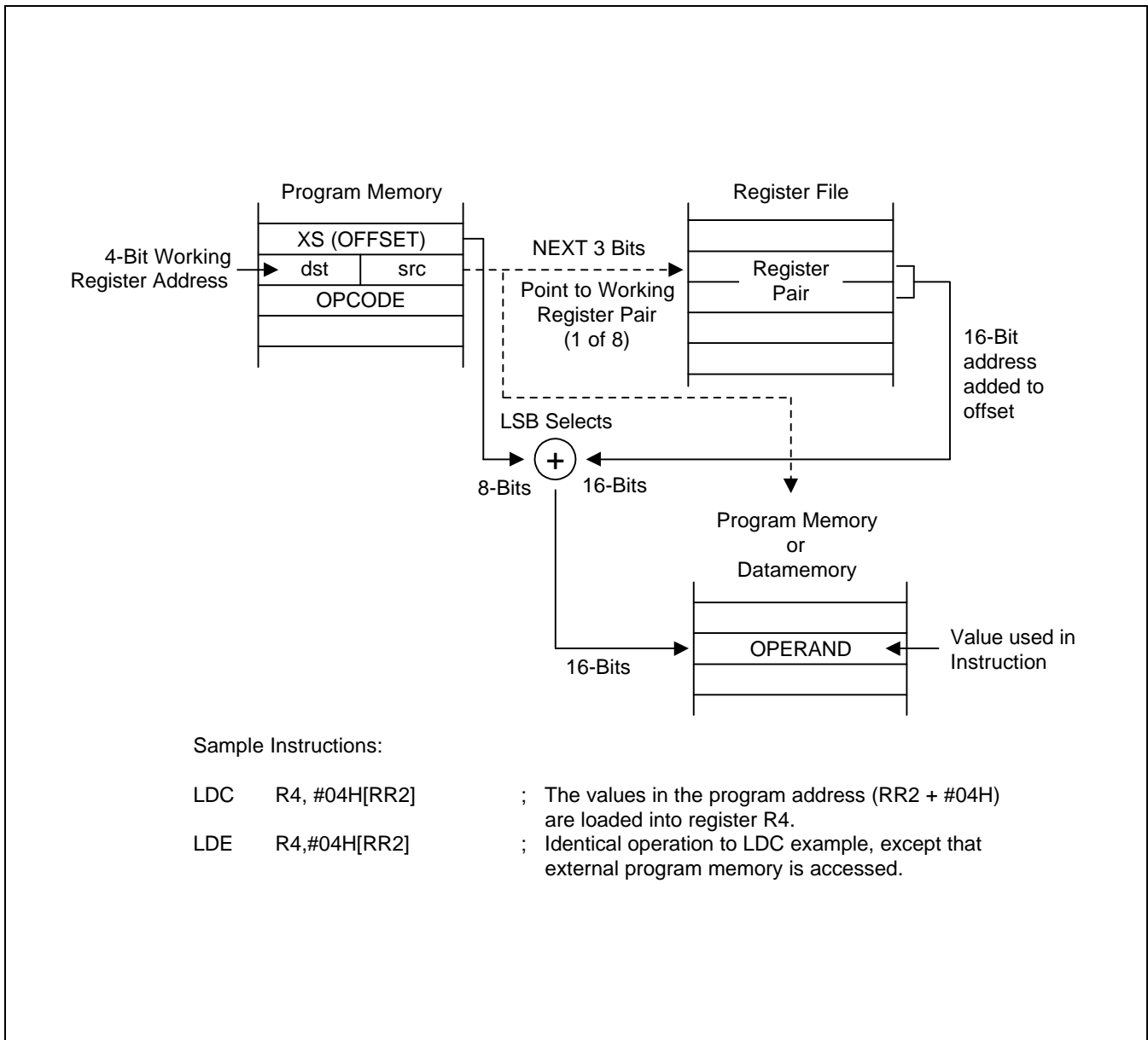


Figure 3-8. Indexed Addressing to Program or Data Memory with Short Offset

INDEXED ADDRESSING MODE (Concluded)

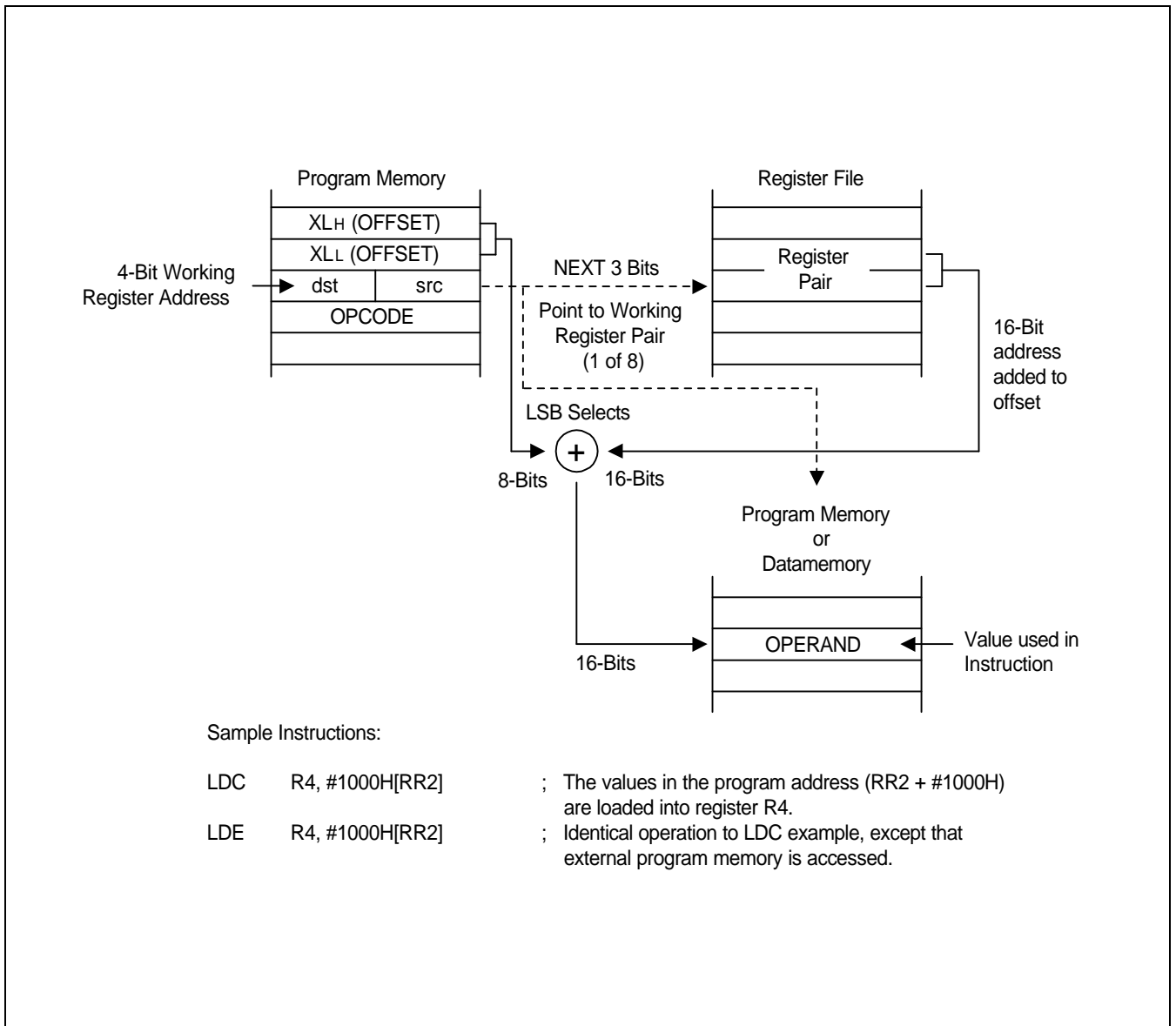


Figure 3-9. Indexed Addressing to Program or Data Memory with Long Offset

DIRECT ADDRESS MODE (DA)

In Direct Address (DA) mode, the instruction provides the operand's 16-bit memory address. Jump (JP) and Call (CALL) instructions use this addressing mode to specify the 16-bit destination address that is loaded into the PC whenever a JP or CALL instruction is executed.

The LDC and LDE instructions can use Direct Address mode to specify the source or destination address for Load operations to program memory (LDC) or to external data memory (LDE), if implemented.

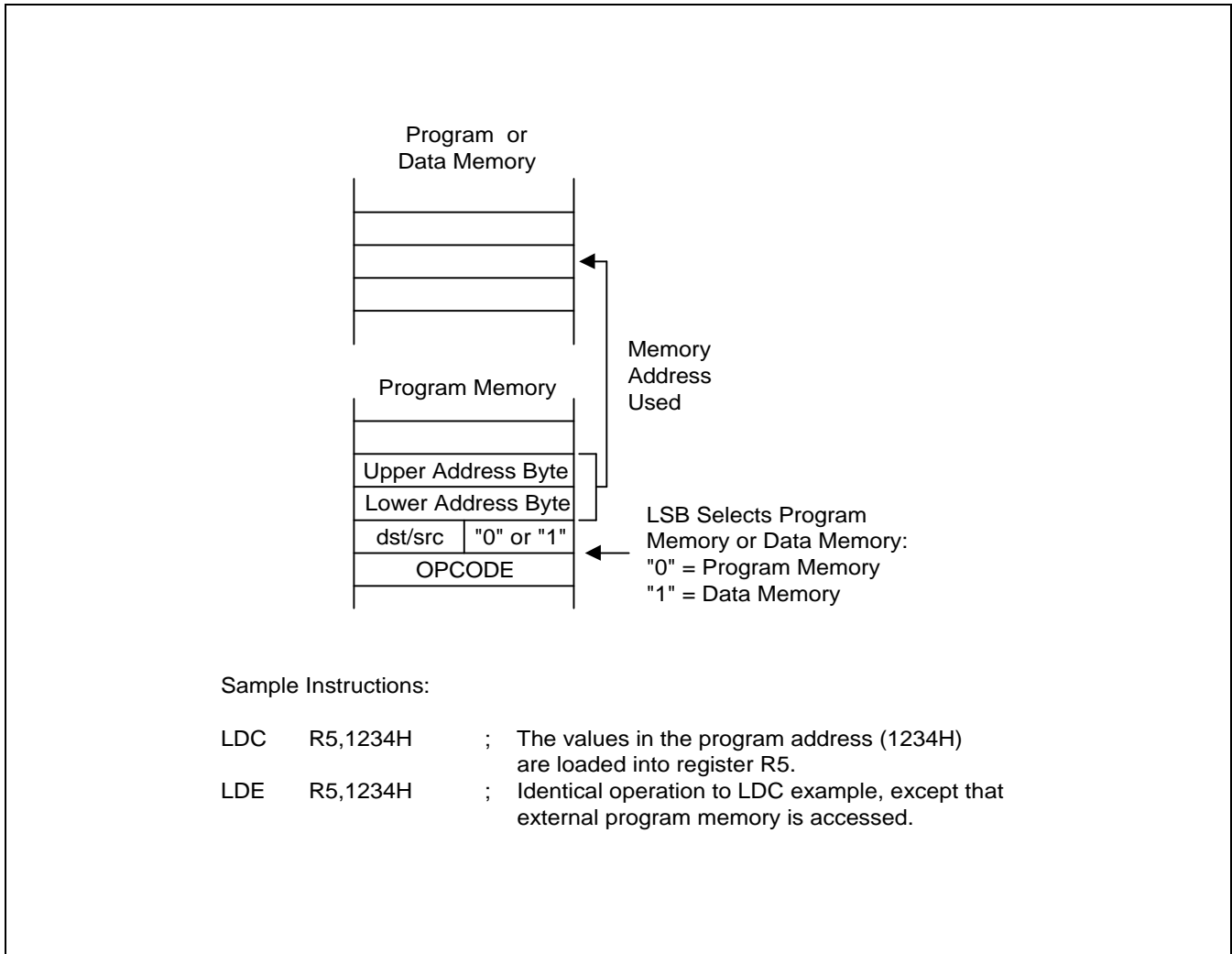


Figure 3-10. Direct Addressing for Load Instructions

DIRECT ADDRESS MODE (Continued)

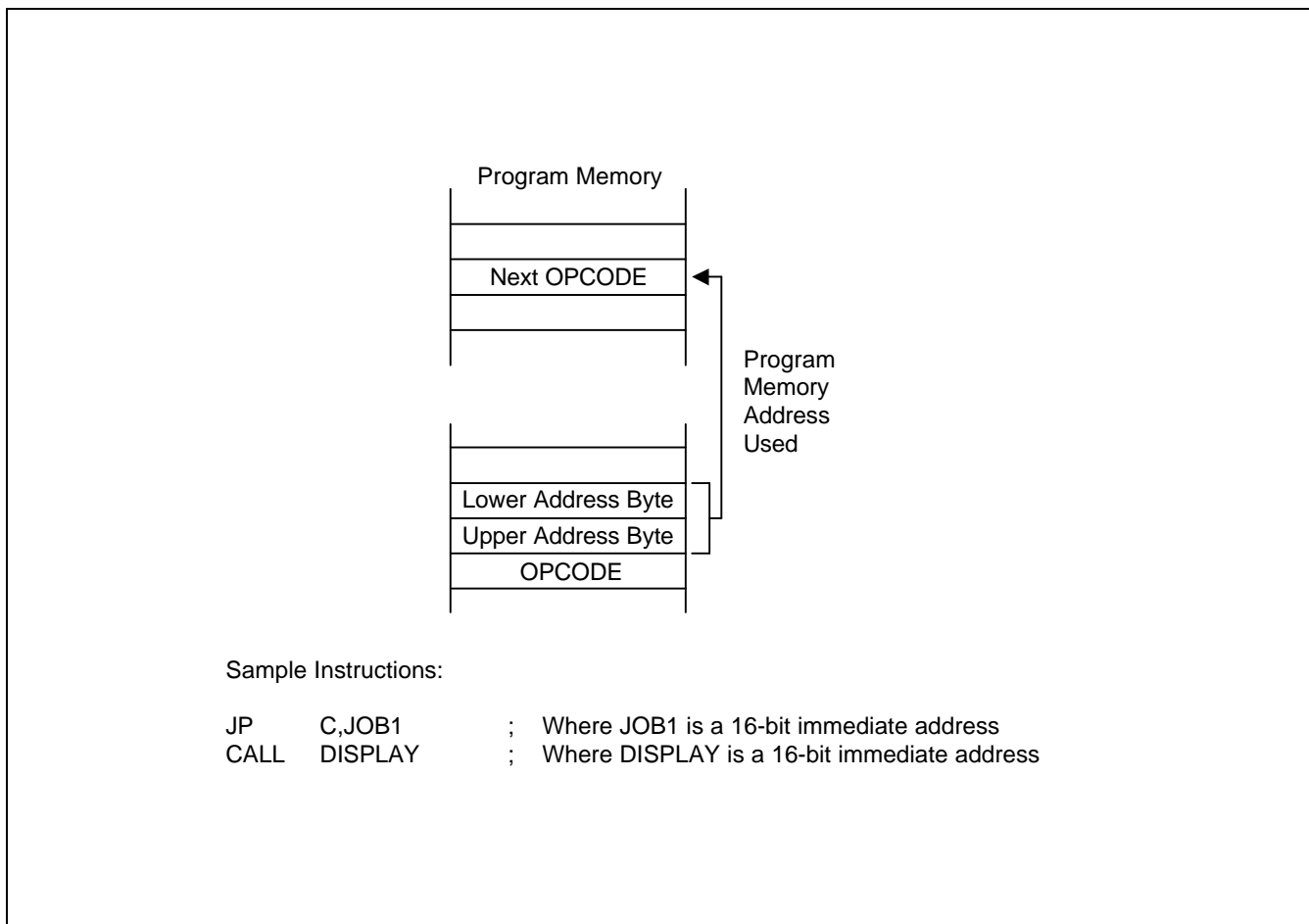


Figure 3-11. Direct Addressing for Call and Jump Instructions

RELATIVE ADDRESS MODE (RA)

In Relative Address (RA) mode, a two's-complement signed displacement between - 128 and + 127 is specified in the instruction. The displacement value is then added to the current PC value. The result is the address of the next instruction to be executed. Before this addition occurs, the PC contains the address of the instruction immediately following the current instruction.

The instructions that support RA addressing is JR.

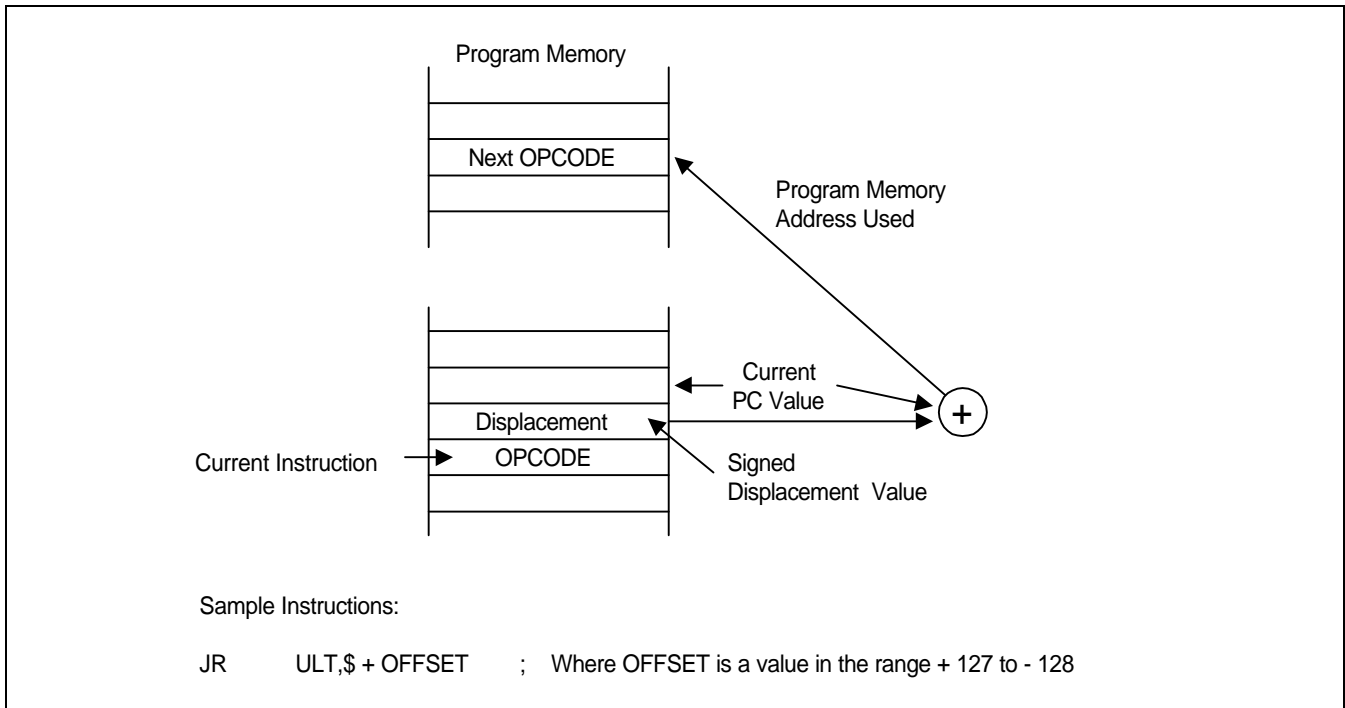


Figure 3-12. Relative Addressing

IMMEDIATE MODE (IM)

In Immediate (IM) addressing mode, the operand value used in the instruction is the value supplied in the operand field itself. Immediate addressing mode is useful for loading constant values into registers.

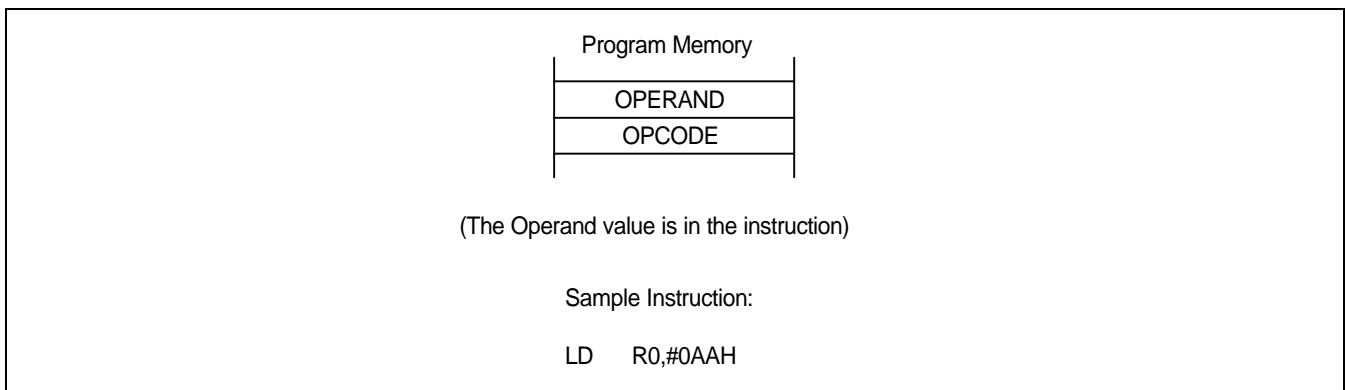


Figure 3-13. Immediate Addressing

4 CONTROL REGISTERS

OVERVIEW

In this section, detailed descriptions of the S3C9664 control registers are presented in an easy-to-read format. These descriptions will help familiarize you with the mapped locations in the register file. You can also use them as a quick-reference source when writing application programs.

System and peripheral registers are summarized in Table 4-1. Figure 4-1 illustrates the important features of the standard register description format.

Control register descriptions are arranged in alphabetical order according to register mnemonic. More information about control registers is presented in the context of the various peripheral hardware descriptions in Part II of this manual.

Table 4-1. Register Map and Reset Status (Page 0)

Register Name	Mnemonic	Hex	R/W
General purpose register file & Stack area	–	00-BFH	R/W
Working register area	–	C0H-CFH	R/W
Timer 0 counter register	T0CNT	D0H	R
Timer 0 data register	T0DATA	D1H	R/W
Timer 0 control register	T0CON	D2H	R/W
Timer 0 interrupt control register	T0INT	D3H	R/W
Clock control register	CLKCON	D4H	R/W
System flags register	FLAGS	D5H	R/W
A/D converter data register (Low byte)	ADDATAL	D6H	R
A/D converter data register (High byte)	ADDATAH	D7H	R
A/D control register	ADCON	D8H	R/W
Stack Pointer Register	SP	D9H	R/W
Port 2 data register	P2	DAH	R/W
Location DBH is not mapped.			
Basic timer control register	BTCON	DCH	R/W
Basic timer counter	BTCNT	DDH	R
Location DEH is not mapped.			
System Mode Register	SYM	DFH	R/W
Port 0 data register	P0	E0H	R/W
Port 1 data register	P1	E1H	R/W
Timer 1 counter register	T1CNT	E2H	R
Timer 1 control register	T1CON	E3H	R/W
Port 0 pull-up/down register (low byte)	P0PURL	E4H	R/W
Port 0 pull-up/down register (high byte)	P0PURH	E5H	R/W
Port 0 control register (low byte)	P0CONL	E6H	R/W
Port 0 control register (high byte)	P0CONH	E7H	R/W
Port 1 control register (low byte)	P1CONL	E8H	R/W
Port 1 control register (high byte)	P1CONH	E9H	R/W
Port 0 interrupt enable register	P0INT	EAH	R/W
Port 0 interrupt pending register	P0PND	EBH	R/W
Port 1 interrupt enable register	P1INT	ECH	R/W
Port 1 interrupt pending register	P1PND	EDH	R/W
Timer 1 data register	T1DATA	EEH	R/W
Port 2 control/interrupt and Pending register	P2CONINT	EFH	R/W

Table 4-1. Register Map and Reset Status (Page 0) (Continued)

Register Name	Mnemonic	Hex	R/W
USB function address register	FADDR	F0H	R/W
USB control endpoint status register	EP0CSR	F1H	R/W
USB interrupt endpoint 1 status register	EP1CSR	F2H	R/W
USB control endpoint byte count register	EP0BCNT	F3H	R/W
USB control endpoint FIFO register	EP0FIFO	F4H	R/W
USB interrupt endpoint 1 FIFO register	EP1FIFO	F5H	R/W
USB interrupt pending register	USBPND	F6H	R/W
USB interrupt enable register	USBINT	F7H	R/W
USB power management register	PWRMGR	F8H	R/W
USB interrupt endpoint 2 status register	EP2CSR	F9H	R/W
USB interrupt endpoint 2 FIFO register	EP2FIFO	FAH	R/W
Endpoint mode register	EPMODE	FBH	R/W
USB interrupt endpoint 1 byte count register	EP1BCNT	FCH	R/W
USB interrupt endpoint 2 byte count register	EP2BCNT	FDH	R/W
USB control register	USBCON	FEH	R/W
Sub control register	SUBCON	FFH	R/W

Table 4-2. Register Map and Reset Status (Page 1)

Register Name	Mnemonic	Hex	R/W
XCON register	XCON	FEH	W

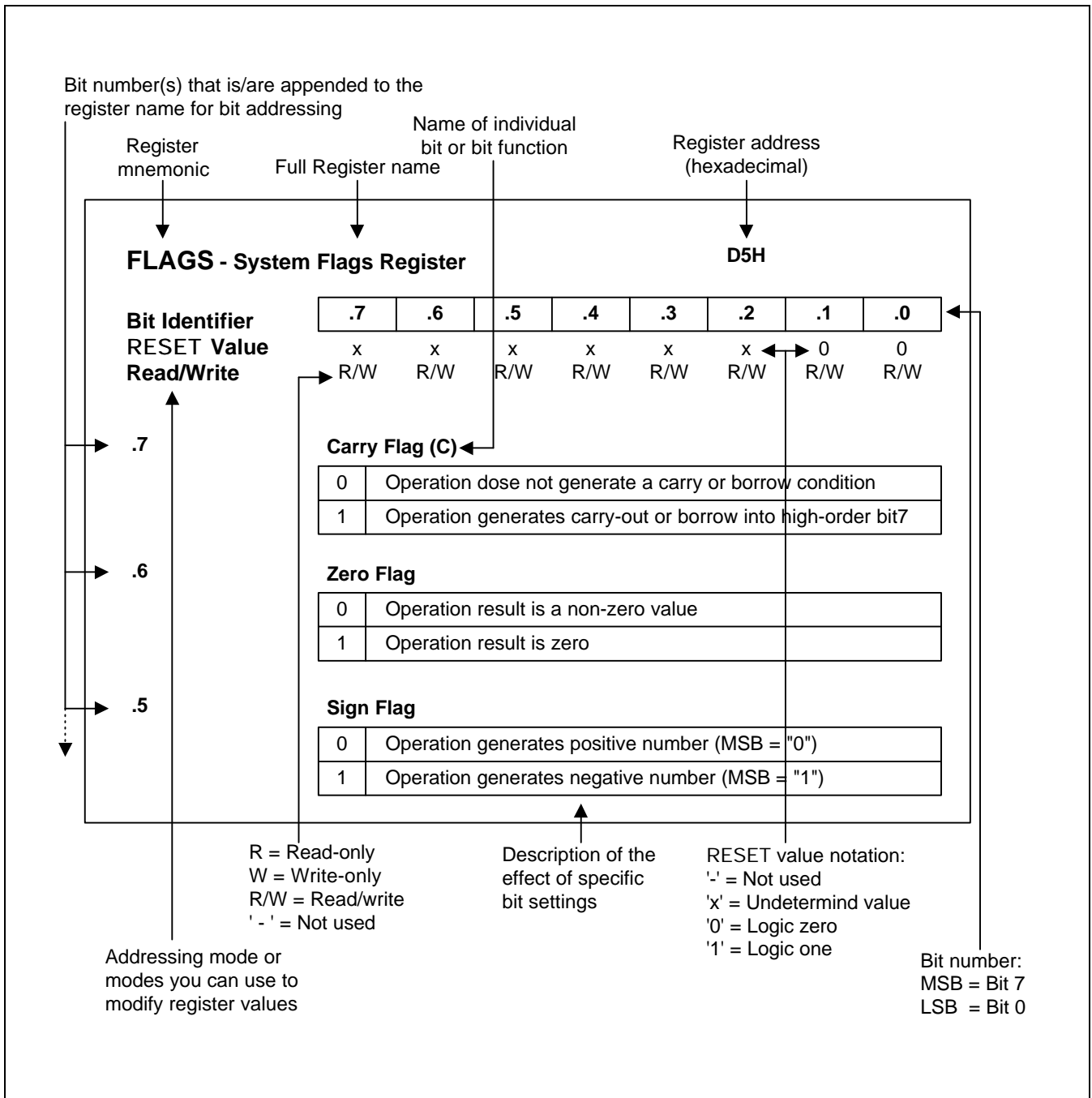


Figure 4-1. Register Description Format

ADCON — A/D Converter Control Register

Page 0, D8H

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

.7	Not used for the S3C9664/P9664
----	--------------------------------

.6–.4 Analog Input Pin Selection Bits

0	0	0	ADC0 (P1.0)
0	0	1	ADC1 (P1.1)
0	1	0	ADC2 (P1.2)
0	1	1	ADC3 (P1.3)
1	0	0	ADC4 (P1.4)
1	0	1	ADC5 (P1.5)
1	1	0	Not used for the S3C9664/P9664
1	1	1	Not used for the S3C9664/P9664

.3 End-of-Conversion Status Bit

0	A/D conversion not complete (when read)
1	A/D conversion is complete (when read)

.2–.1 Conversion Speed Selection Bits

0	0	$f_{OSC}/16$
0	1	$f_{OSC}/8$
1	0	$f_{OSC}/4$
1	1	f_{OSC}

.0 Conversion Start Bit

0	Automatically reset after conversion starting
1	Starting

NOTE: To configure an A/D converter input channel, you must also make the proper setting in the port 1 control register, P1CONL (ADC0–ADC3) or P1CONH (ADC4–ADC5). Only one input can be configured at one time.

BTCON — Basic Timer Control Register

Page 0, DCH

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

.7 – .4**Watchdog Timer Enable Bits**

1	0	1	0	Disable watchdog function
Any other value				Enable watchdog function

.3 – .2**Basic Timer Input Clock Selection Bits**

0	0	$f_{OSC}/4096$
0	1	$f_{OSC}/1024$
1	0	$f_{OSC}/128$
1	1	Non divided (f_{OSC})

.1**Basic Timer Counter Clear Bit** (note)

0	No effect
1	Clear BTCNT

.0**Basic Timer Divider Clear Bit** (note)

0	No effect
1	Clear both dividers

NOTE: When you write a "1" to BTCON.0 (or BTCON.1), the basic timer counter (or basic timer divider) is cleared. The bit is then cleared automatically to "0".

CLKCON — System Clock Control Register

Page 0, D4H

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

.7 Oscillator IRQ Wake-up Function Bit

0	Enable IRQ for main system oscillator wake-up in power down mode
1	Disable IRQ for main system oscillator wake-up in power down mode

.6 and .5 Not used for S3C9664**.4 and .3 CPU Clock (System Clock) Selection Bits**

0	0	Divide by 16 ($f_{OSC}/16$)
0	1	Divide by 8 ($f_{OSC}/8$)
1	0	Divide by 2 ($f_{OSC}/2$)
1	1	Non-divided clock (f_{OSC})

.2 – .0 Not used for S3C9664

EPOBCNT — Endpoint 0 Write Counter register

Page 0, F3H

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

.7 Data Toggle check bit

0	Data 0 transaction toggle
1	Data 1 transaction toggle

.6 Reserved**.5 Over 8 bytes Received bit**

0	Normal Operation
1	Indicates over 8 bytes received

.4 Enable Bit

0	Disable endpoint 0
1	Enable endpoint 0

.3 – .0 The Byte Counter of Data stored in endpoint 0

0	0	0	0	Minimum bytes stored in endpoint 0
1	0	0	0	Maximum bytes stored in endpoint 0

EP1BCNT — Endpoint 1 Write Counter register

Page 0, FCH

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

.7

Data Toggle check bit

0	Data 0 transaction toggle
1	Data 1 transaction toggle

.6

Reserved

.5

Over 8 bytes Received bit

0	Normal Operation
1	Indicates over 8 bytes received

.4

Enable Bit

0	Disable endpoint 1
1	Enable endpoint 1

.3 – .0

The Byte Counter of Data stored in endpoint 1

0	0	0	0	Minimum bytes can be stored in endpoint 1
1	0	0	0	Maximum bytes can be stored in endpoint 1

EP2BCNT — Endpoint 2 Write Counter register

Page 0, FDH

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

.7 Data Toggle check bit

0	Data 0 transaction toggle
1	Data 1 transaction toggle

.6 Reserved**.5 Over 8 bytes Received bit**

0	Normal Operation
1	Indicates over 8 bytes received

.4 Enable Bit

0	Disable endpoint 2
1	Enable endpoint 2

.3 – .0 The Byte Counter of Data stored in endpoint 2

0	0	0	0	Minimum bytes can be stored in endpoint 2
1	0	0	0	Maximum bytes can be stored in endpoint 2

EPOCSR — Control Endpoint Status Register

Page 0, F1H

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

.7**SETUP_END Clear Bit**

0	No effect (when write)
1	Clear SETUP_END (bit4) bit

.6**OUT_PKT_RDY Clear Bit**

0	No effect (when write)
1	Clear OUT_PKT_RDY (bit0) bit

.5**STALL Signal Sending Bit**

0	No effect (when write)
1	Send STALL signal to host

.4**Setup Transfer End Bit**

0	No effect (when write)
1	SIE sets this bit when a control transfer ends before DATA_END (bit3) is set

.3**Setup Data End Bit**

0	No effect (when write)
1	MCU set this bit after loading or unloading the last packet data into the FIFO

.2**STALL Signal Receive Bit**

0	MCU clear this bit to end the STALL condition
1	SIE sets this bit if a control transaction is ended due to a protocol violation

.1**In Packet Ready Bit**

0	SIE clear this bit once the packet has been successfully sent to the host
1	MCU sets this bit after writing a packet of data into Endpoint0 FIFO

.0**Out Packet Ready Bit**

0	No effect (when write)
1	SIE sets this bit once a valid token is written to the FIFO

EP1CSR — Interrupt Endpoint Status Register

Page 0, F2H

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

a) The belows are configured as IN mode.

.7 DATA_TOGGLE Clear Bit

0	No effect (when write)
1	Clears the data toggle sequence bit

.6 – .3 Maximum Packet Size Bits

0	No effect (when write)
1	Indicates the maximum packet size for interrupt endpoint

.2 FIFO Flush Bit

0	No effect (when write)
1	FIFO is flushed, and IN_PKT_RDY cleared

.1 Force STALL Bit

0	MCU clears this bit to end the STALL condition
1	Issues a STALL handshake to USB

.0 In Packet Ready Bit

0	SIE clear this bit once the packet has been successfully sent to the host
1	MCU sets this bit after writing a packet of data into Endpoint 1 FIFO

EP1CSR — Interrupt Endpoint Status Register

Page 0, F2H

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

b) The belows are configured as OUT mode.

.7	Reserved
----	----------

.6	OUT_PKT_RDY Clear Bit	
	0	No effect (when write)
	1	Clear OUT_PKT_RDY (bit0) bit

.5-.4	Reserved
-------	----------

.3	STALL Signal Receive Bit	
	0	MCU can clear this bit
	1	SIE sets this bit after sending stall packet

.2	FIFO Flush Bit	
	0	No effect (when write)
	1	FIFO is flushed, and IN_PKT_RDY cleared

.1	Force STALL Bit	
	0	MCU clears this bit to end the STALL condition
	1	Issues a STALL handshake to USB

.0	Out Packet Ready Bit	
	0	No effect (when write)
	1	SIE sets this bit once a valid data is written to the FIFO

EP2CSR — Interrupt Endpoint Status Register

Page 0, F9H

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

a) The belows are configured as IN mode.

.7 DATA_TOGGLE Clear Bit

0	No effect (when write)
1	Clears the data toggle sequence bit

.6 – .3 Maximum Packet Size Bits

0	No effect (when write)
1	Indicates the maximum packet size for interrupt endpoint

.2 FIFO Flush Bit

0	No effect (when write)
1	FIFO is flushed, and IN_PKT_RDY cleared

.1 Force STALL Bit

0	MCU clears this bit to end the STALL condition
1	Issues a STALL handshake to USB

.0 In Packet Ready Bit

0	SIE clear this bit once the packet has been successfully sent to the host
1	MCU sets this bit after writing a packet of data into Endpoint 2 FIFO

EP2CSR — Interrupt Endpoint Status Register

Page 0, F9H

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

b) The belows are configured as OUT mode.

.7 Reserved**.6** **OUT_PKT_RDY Clear Bit**

0	No effect (when write)
1	Clear OUT_PKT_RDY (bit0) bit

.5-.4 Reserved**.3** **STALL Signal Receive Bit**

0	MCU can clear this bit
1	SIE sets this bit after sending stall packet

.2 **FIFO Flush Bit**

0	No effect (when write)
1	FIFO is flushed, and IN_PKT_RDY cleared

.1 **Force STALL Bit**

0	MCU clears this bit to end the STALL condition
1	Issues a STALL handshake to USB

.0 **Out Packet Ready Bit**

0	No effect (when write)
1	SIE sets this bit once a valid data is written to the FIFO

EPMODE— Endpoint Mode register

Page 0, E7H

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

.7 and .6**Reset Length**

0	0	22.4 μ s + α
0	1	12.0 μ s + α
1	0	6.0 μ s + α
1	1	4.0 μ s + α

NOTE: $\alpha \cong \pm 0.66 \mu$ s**.5 - .2****Reserved****.1****Endpoint 2 Mode**

0	Endpoint 2 acts as an IN interrupt endpoint
1	Endpoint 2 acts as an OUT interrupt endpoint

.0**Endpoint 1 Mode**

0	Endpoint 1 acts as an IN interrupt endpoint
1	Endpoint 1 acts as an OUT interrupt endpoint

FLAGS — System Flags Register

Page 0, D5H

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

.7

Carry Flag (C)

0	Operation does not generate a carry or borrow condition
---	---

.6

Zero Flag (Z)

0	Operation result is a non-zero value
1	Operation result is zero

.5

Sign Flag (S)

0	Operation generates a positive number (MSB = "0")
1	Operation generates a negative number (MSB = "1")

.4

Overflow Flag (V)

0	Operation result is $\leq +127$ or ≥ -128
1	Operation result is $\geq +127$ or ≤ -128

.3 – .0

Not used for S3C9664

P0CONH — Port 0 Control High Byte Register

Page 0, E7H

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

.7 and .6**Port 0, P0.7/INT0 Configuration Bits**

0	0	Schmitt trigger input; interrupt on falling edges
0	1	Schmitt trigger input; interrupt on rising edges
1	0	N-CH open drain output
1	1	Push-pull output

.5 and .4**Port 0, P0.6/INT0 Configuration Bits**

0	0	Schmitt trigger input; interrupt on falling edges
0	1	Schmitt trigger input; interrupt on rising edges
1	0	N-CH open drain output
1	1	Push-pull output

.3 and .2**Port 0, P0.5/INT0 Configuration Bits**

0	0	Schmitt trigger input; interrupt on falling edges
0	1	Schmitt trigger input; interrupt on rising edges
1	0	N-CH open drain output
1	1	Push-pull output

.1 and .0**Port 0, P0.4/INT0 Configuration Bits**

0	0	Schmitt trigger input; interrupt on falling edges
0	1	Schmitt trigger input; interrupt on rising edges
1	0	N-CH open drain output
1	1	Push-pull output

P0CONL — Port 0 Control Low Byte Register

Page 0, E6H

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

.7 and .6**Port 0, P0.3/INT0 Configuration Bits**

0	0	Schmitt trigger input; interrupt on falling edges
0	1	Schmitt trigger input; interrupt on rising edges
1	0	N-CH open drain output
1	1	Push-pull output

.5 and .4**Port 0, P0.2/INT0 Configuration Bits**

0	0	Schmitt trigger input; interrupt on falling edges
0	1	Schmitt trigger input; interrupt on rising edges
1	0	N-CH open drain output
1	1	Push-pull output

.3 and .2**Port 0, P0.1/INT0 Configuration Bits**

0	0	Schmitt trigger input; interrupt on falling edges
0	1	Schmitt trigger input; interrupt on rising edges
1	0	N-CH open drain output
1	1	Push-pull output

.1 and .0**Port 0, P0.0/INT0 Configuration Bits**

0	0	Schmitt trigger input; interrupt on falling edges (or capture input)
0	1	Schmitt trigger input; interrupt on rising edges
1	0	N-CH open drain output
1	1	Alternative Function (T0 out : match or PWM)

POINT — Port 0 Interrupt Enable Register

Page 0, EAH

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

.7 P0.7/INT0 Interrupt Enable Bit

0	Disable P0.7 interrupt
1	Enable P0.7 interrupt

.6 P0.6/INT0 Interrupt Enable Bit

0	Disable P0.6 interrupt
1	Enable P0.6 interrupt

.5 P0.5/INT0 Interrupt Enable Bit

0	Disable P0.5 interrupt
1	Enable P0.5 interrupt

.4 P0.4/INT0 Interrupt Enable Bit

0	Disable P0.4 interrupt
1	Enable P0.4 interrupt

.3 P0.3/INT0 Interrupt Enable Bit

0	Disable P0.3 interrupt
1	Enable P0.3 interrupt

.2 P0.2/INT0 Interrupt Enable Bit

0	Disable P0.2 interrupt
1	Enable P0.2 interrupt

.1 P0.1/INT0 Interrupt Enable Bit

0	Disable P0.1 interrupt
1	Enable P0.1 interrupt

.0 P0.0/INT0 Interrupt Enable Bit

0	Disable P0.0 interrupt
1	Enable P0.0 interrupt

POPND — Port 0 interrupt Pending Register

Page 0, EBH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (note)	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 P0.7/INT0 Interrupt Pending Bit

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.6 P0.6/INT0 Interrupt Pending Bit

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.5 P0.5/INT0 Interrupt Pending Bit

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.4 P0.4/INT0 Interrupt Pending Bit

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.3 P0.3/INT0 Interrupt Pending Bit

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.2 P0.2/INT0 Interrupt Pending Bit

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.1 P0.1/INT0 Interrupt Pending Bit

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.0 P0.0/INT0 Interrupt Pending Bit

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

NOTE: To clear a port 0 interrupt pending condition, write a "0" to the corresponding POPND register bit location.

POPURH — Port0 Pull-up/down Enable Register (High Byte)

Page 0, E5H

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

.7 and .6**Port 0.7 Pull-up/down bits**

0	0	Enable Pull-up
0	1	Enable Pull-down
1	0	Disable Pull-up/down
1	1	Disable Pull-up/down

.5 and .4**Port 0.6 Pull-up/down bits**

0	0	Enable Pull-up
0	1	Enable Pull-down
1	0	Disable Pull-up/down
1	1	Disable Pull-up/down

.3 and .2**Port 0.5 Pull-up/down bits**

0	0	Enable Pull-up
0	1	Enable Pull-down
1	0	Disable Pull-up/down
1	1	Disable Pull-up/down

.1 and .0**Port 0.4 Pull-up/down bits**

0	0	Enable Pull-up
0	1	Enable Pull-down
1	0	Disable Pull-up/down
1	1	Disable Pull-up/down

NOTE: Pull –up/down resistor is to be automatically disabled on Push-Pull Output mode and Open-Drain output mode. Otherwise, which can be enabled in all input modes

POPURL — Port 0 Pull-up/down Enable Register (Low byte)

Page 0, E4H

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

.7 and .6**Port 0.3 Pull-up/down bits**

0	0	Enable Pull-up
0	1	Enable Pull-down
1	0	Disable Pull-up/down
1	1	Disable Pull-up/down

.5 and .4**Port 0.2 Pull-up/down bits**

0	0	Enable Pull-up
0	1	Enable Pull-down
1	0	Disable Pull-up/down
1	1	Disable Pull-up/down

.3 and .2**Port 0.1 Pull-up/down bits**

0	0	Enable Pull-up
0	1	Enable Pull-down
1	0	Disable Pull-up/down
1	1	Disable Pull-up/down

.1 and .0**Port 0.0 Pull-up/down bits**

0	0	Enable Pull-up
0	1	Enable Pull-down
1	0	Disable Pull-up/down
1	1	Disable Pull-up/down

NOTE: Pull –up/down resistor is to be automatically disabled on Push-Pull Output mode and Open drain Output mode . Otherwise, which can be enabled in all input modes.

P1CONH — Port 1 Control High Byte Register

Page 0, E9H

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

.7 and .6**Port 1, P1.7/INT1 Configuration Bits**

0	0	input, falling edge external interrupt with pull-up resistor
0	1	Input, rising edge external interrupt
1	0	N-CH open drain out
1	1	Push-pull output

.5 and .4**Port 1, P1.6/INT1 Configuration Bits**

0	0	input, falling edge external interrupt with pull-up resistor
0	1	Input, rising edge external interrupt
1	0	N-CH open drain out
1	1	Push-pull output

.3 and .2**Port 1, P1.5/AD5 /INT1 Configuration Bits**

0	0	input, falling edge external interrupt pull-up resistor enabled; A/D converter off
0	1	input; A/D converter off, rising edge external interrupt
1	0	A/D converter input (AD5); input off
1	1	Push-pull output

.1 and .0**Port 1, P1.4/AD4/INT1 Configuration Bits**

0	0	input, falling edge external interrupt pull-up resistor enabled; A/D converter off
0	1	input; A/D converter off, rising edge external interrupt
1	0	A/D converter input (AD4); input off
1	1	Push-pull output

P1CONL — Port 1 Control Low Byte Register

Page 0, E8H

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

.7 and .6**Port 1, P1.3/AD3/INT1 Configuration Bits**

0	0	Input, falling edge external interrupt pull-up resistor enabled; A/D converter off
0	1	Input; A/D converter off, rising edge external interrupt
1	0	A/D converter input (AD3); input off
1	1	Push-pull output

.5 and .4**Port 1, P1.2/AD2 /INT1 Configuration Bits**

0	0	Input, falling edge external interrupt pull-up resistor enabled; A/D converter off
0	1	Input; A/D converter off, rising edge external interrupt
1	0	A/D converter input (AD2); input off
1	1	Push-pull output

.3 and .2**Port 1, P1.1/AD1/INT1 Configuration Bits**

0	0	Input, falling edge external interrupt pull-up resistor enabled; A/D converter off
0	1	Input; A/D converter off, rising edge external interrupt
1	0	A/D converter input (AD1); input off
1	1	Push-pull output

.1 and .0**Port 1, P1.0/AD0/INT1 Configuration Bits**

0	0	input, falling edge external interrupt pull-up resistor enabled; A/D converter off
0	1	input; A/D converter off, rising edge external interrupt
1	0	A/D converter input (AD0); input off
1	1	Push-pull output

P1INT — Port 1 Interrupt Enable Register

Page 0, ECH

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

.7 P1.7/INT1 Interrupt Enable Bit

0	Disable P1.7 interrupt
1	Enable P1.7 interrupt

.6 P1.6/INT1 Interrupt Enable Bit

0	Disable P1.6 interrupt
1	Enable P1.6 interrupt

.5 P1.5/INT1 Interrupt Enable Bit

0	Disable P1.5 interrupt
1	Enable P1.5 interrupt

.4 P1.4/INT1 Interrupt Enable Bit

0	Disable P1.4 interrupt
1	Enable P1.4 interrupt

.3 P1.3/INT1 Interrupt Enable Bit

0	Disable P1.3 interrupt
1	Enable P1.3 interrupt

.2 P1.2/INT1 Interrupt Enable Bit

0	Disable P1.2 interrupt
1	Enable P1.2 interrupt

.1 P1.1/INT1 Interrupt Enable Bit

0	Disable P1.1 interrupt
1	Enable P1.1 interrupt

.0 P1.0/INT1 Interrupt Enable Bit

0	Disable P1.0 interrupt
1	Enable P1.0 interrupt

P1PND — Port 1 Interrupt Pending Register

Page 0, EDH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (note)	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7**P1.7/INT1 Interrupt Pending Bit**

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.6**P1.6/INT1 Interrupt Pending Bit**

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.5**P1.5/INT1 Interrupt Pending Bit**

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.4**P1.4/INT1 Interrupt Pending Bit**

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.3**P1.3/INT1 Interrupt Pending Bit**

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.2**P1.2/INT1 Interrupt Pending Bit**

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.1**P1.1/INT1 Interrupt Pending Bit**

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

.0**P1.0/INT1 Interrupt Pending Bit**

0	No interrupt pending (when bit is read)
1	Interrupt is pending (when bit is read)

NOTE: To clear a port 1 interrupt pending condition, write a "0" to the corresponding P1PND register bit location.

P2CONINT— Port 2 Control/Interrupt Register

Page 0, EFH

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

.7 and .6**Port 2, P2.1/INT2 Configuration Bits**

0	0	Input, rising edge external interrupt
0	1	Input, falling edge external interrupt with pull-up resistor
1	0	N-CH open drain out
1	1	Push-pull output

.5 and .4**Port 2, P2.0/INT2 Configuration Bits**

0	0	Input, rising edge external interrupt
0	1	Input, falling edge external interrupt with pull-up resistor
1	0	N-CH open drain out
1	1	Push-pull output

.3 and .2**P2.0-P2.1 Interrupt enable Bits**

0	External interrupt disable
1	External interrupt enable

.1 and .0**P2.0-P2.1 Interrupt Pending Bits**

0	No pending (when read)/clear pending bit
1	Pending (when read)/No effect (When write)

PWRMGR — USB Power Management Register

Page 0, F8H

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

.7 – .5 **Reserved**.4 **VPIN Value Bit**

0	Indicates the value of VPIN is low
1	Indicates the value of VPIN is high

.3 **VMIN Value Bit**

0	Indicates the value of VMIN is low
1	Indicates the value of VMIN is high

.2 **Clear Suspend Counter**

0	No effect
1	Clear suspend counter

.1 **RESUME Signal Sending Bit**

0	RESUME signal is ended
1	While in suspend state, if the MCU wants to initiate a resume, it writes a 1 to this register for 10 ms (maximum of 15 ms), and clears this register. In suspend mode, if this bit is set to "1", USB generates resume signaling.

.0 **SUSPEND Status Bit**

0	Cleared automatically when MCU writes a zero to RESUME signal sending bit or when function receives resume signal from the host while in suspend mode
1	This bit is set when SUSPEND interrupt occur

SUBCON — SUB_Oscillator Control

Page 0, EFH

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

.7 Sub_Oscillator Interrupt Enable Bit

0	Sub_oscillator interrupt disable
1	Sub_oscillator interrupt enable

.6 Sub_Oscillator Interrupt Pending Bit

0	No pending (when read)/clear pending (when write)
1	Pending (when read)/no effect (when write)

.5 Sub_Oscillator Counter clear Bit

0	No effect/ after counter cleared ,automatically this bit is then cleared to “0”
1	Clear Sub_Oscillator counter clear

Sub_Oscillator Enable Bit

0	Sub_oscillator disable
1	Sub_oscillator enable

Sub_Oscillator Counter Input Clock Selection Bits

.3 and .1	0	0	0	$f_{OSC}/2048$
	0	0	1	$f_{OSC}/3072$
	0	1	0	$f_{OSC}/4096$
	0	1	1	$f_{OSC}/6144$
	1	0	0	$f_{OSC}/8192$
	1	0	1	$f_{OSC}/12288$
	1	1	0	$f_{OSC}/16384$
	1	1	1	$f_{OSC}/24576$

.0	Not used for S3C9664
----	----------------------

NOTE: $f_{OSC} = 158.86 \text{ kHz} (\pm 10 \%)$

SYM — System Mode Register

Page 0, DFH

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

.7 – .4

Not used for S3C9664

.3

Global Interrupt Enable Bit

0	Global interrupt processing disable
1	Global interrupt processing enable

.2 and .0

Page Select Bit

0	0	0	Page 0
0	0	1	Page 1
0	1	0	Page 2 (Not allowed in S3C9664)
0	0	1	Page 3 (Not allowed in S3C9664)

T0CON — Timer 0 Control Register

Page 0, D2H

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

.7 and .6**T0 Counter Input Clock Selection Bits**

0	0	$f_{OSC}/4096$
0	1	$f_{OSC}/256$
1	0	$f_{OSC}/8$
1	1	$f_{OSC}/1$

.5 and .4**T0 Operating Mode Selection Bits**

0	0	Interval timer mode
0	1	Capture mode (capture on falling edge, counter running, OVF)
1	0	Capture mode (capture on rising edge, counter running, OVF)
1	1	PWM mode (OVF interrupt can occur)

.3**T0 Counter Clear Bit**

0	No effect
1	Clear the timer 0 counter (when write)

.2 and .0

Not used for S3C9664

TOINT — Timer 0 Interrupt Control Register

Page 0, D3H

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

.7 and .4

Not used for S3C9664

.3

T0 Overflow interrupt

0	Disable T0OVF interrupt
1	Enable T0OVF interrupt

.2

T0 match/capture interrupt Enable Bit

0	Disable T0INT interrupt
1	Enable T0INT interrupt

.1

T0 Overflow Interrupt Pending Bit

0	No interrupt pending
0	<i>Clear this pending bit (write)</i>
1	Interrupt is pending

.0

T0 Interrupt Pending Bit (Capture or Match Interrupt)

0	No interrupt pending
0	<i>Clear this pending bit (write)</i>
1	Interrupt is pending

T1CON — Timer 1 Control Register

Page 0, E3H

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

.7 and .6**T1 Input Clock Selection Bits**

0	0	$f_{OSC}/10246$
0	1	$f_{OSC}/256$
1	0	$f_{OSC}/64$
1	1	$f_{OSC}/8$

.5 and .4

Not used for S3C9664

.3**T1 Counter Clear Bit**

0	No effect
1	Clear the timer 0 counter (when write)

.2**T1 Counter Enable Bit**

0	Disable counting operation
1	Enable counting operation

.1**T1 Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.0**T1 Interrupt Pending Bit**

0	No interrupt pending
0	Clear this pending bit (write)
1	Interrupt is pending

USBCON — USB Control Register

Page 0, FEH

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

.7- .6 **Reserved****.5** **DP/DM Control Bit**

0	DP/DM can not be individually controlled by MCU
1	DP/DM can be individually controlled by MCU to set USBCON.4 and USBCON.3

.4 **DP Status Bit**

0	DP is Low
1	DP is High

.3 **DM Status Bit**

0	DM is Low
1	DM is High

.2 **USB Reset MCU Bit**

0	USB which is been on reset can not make MCU reset
1	USB which is been on reset can be able to reset MCU

.1 **MCU reset USB Bit**

0	No effect
1	MCU forces USB be reset

.0 **USB Reset Signal Receive Bit**

0	Clear reset signal Bit
1	This bit is set when host send USB rest signal

USBINT — USB Interrupt Enable Register

Page 0, F7H

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

.7–.5

Not used for S3C9664

.4

USB Reset Interrupt Enable Bit

0	Disable USB reset interrupt
1	Enable USB reset interrupt

.3

Endpoint 2 Interrupt Enable Bit

0	Disable Endpoint 2 interrupt
1	Enable Endpoint 2 interrupt (default)

.2

SUSPEND/RESUME Interrupt Enable Bit

0	Disable SUSPEND and RESUME interrupt (default)
1	Enable SUSPEND and RESUME interrupt

.1

ENDPOINT1 Interrupt Enable Bit

0	Disable ENDPOINT 1 interrupt
1	Enable ENDPOINT 1 interrupt (default)

.0

ENDPOINT0 Interrupt Enable Bit

0	Disable ENDPOINT 0 interrupt
1	Enable ENDPOINT 0 interrupt (default)

USBPND — USB Interrupt Pending Register

Page 0, F6H

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

.7–.6	Not used for S3C9664
-------	----------------------

.5 USB Reset Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, when USB reset need to served

.4 Endpoint Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, when endpoint 2 need to served

.3 RESUME Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, if RESUME signaling is received while in SUSPEND mode

.2 SUSPEND Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, when suspend signaling is received

.1 ENDPOINT1 Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, when endpoint1 needs to be serviced

.0 ENDPOINT0 Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, while endpoint 0 needs to serviced. It is set under the following conditions: <ul style="list-style-type: none"> — OUT_PKT_RDY is set — IN_PKT_RDY get cleared — SENT_STALL gets set — DATA_END gets cleared — SETUP_END gets set

XCON — USB Signal Control Register

Page 1, FEH

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

.7 Pull-Up resistor at (D-) enable bit

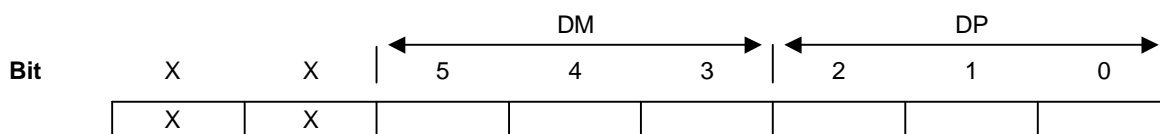
0	Pull-Up resistor at (D-) disable
1	Pull-Up resistor at (D+) enable

.6 GPIO or USB Port Select bit

0	General in/out port enable
1	USB port enable

.6 – .5

This register will be used to control the USB signal quality.
The USB signal (D+/D-) of transceiver can be changed by setting this register.



Edge Control	Bit 5, 2	Bit 4, 1	Bit 3, 0	DLY Value	Unit
Rise edge	0	0	0	DLY 0	About 2.5 nsec
		0	1	DLY 1	
		1	0	DLY 2	
		1	1	DLY 4	
Fall edge	1	0	0	DLY 0	
		0	1	DLY 1	
		1	0	DLY 2	
		1	1	DLY 4	

NOTE: DLY means delay time of rising/falling time.

NUM	HEX Value	Result, DM _{x-point 1} DP _{x-point 2} DM _{x-point 1} DP _{x-point 2} (V _{DD} = 5.12 V)	
1	0x00	Default value, 0.88 V, 1.19 V	
2	0x38	1.58 V, 1.55 V	
3	0x3C	1.50 V, 1.50 V	
4	0x3D	1.65 V, 1.65 V	Good
5	0x3E	1.80 V, 1.80 V	

NOTE: This value is only for OTP products. XCON is a write-only register and can be accessed through page 1.

5 INTERRUPT STRUCTURE

OVERVIEW

The SAM88RCRI interrupt structure has two basic components: a vector, and sources. The number of interrupt sources can be serviced through a interrupt vector which is assigned in ROM address 0000H–0001H.

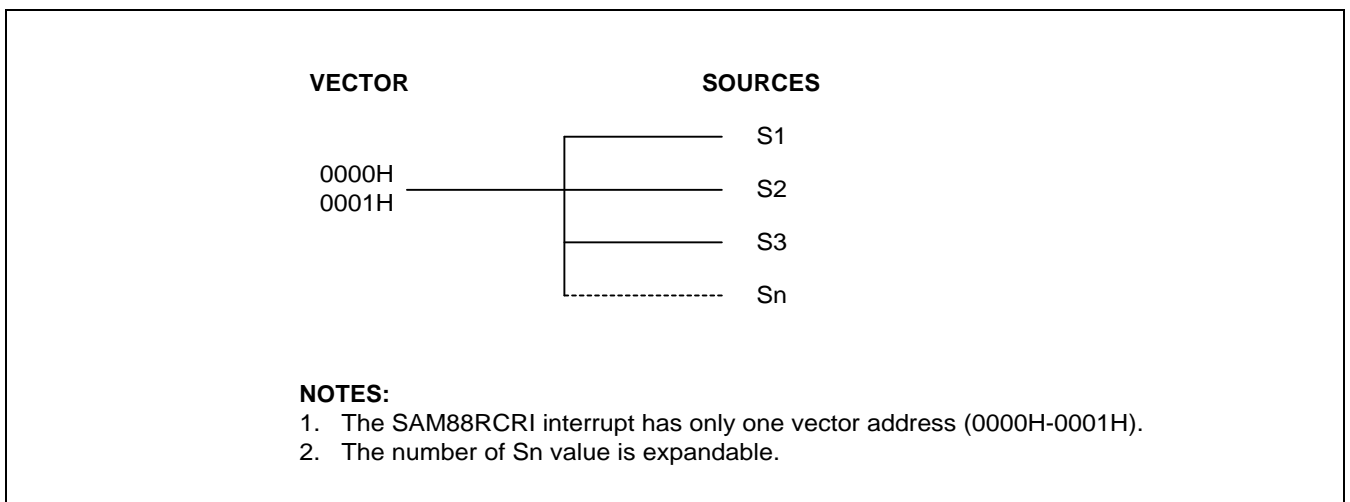


Figure 5-1. S3C9-Series Interrupt Type

INTERRUPT PROCESSING CONTROL POINTS

Interrupt processing can be controlled in two ways: globally, or by specific interrupt level and source. The system-level control points in the interrupt structure are therefore:

- Global interrupt enable and disable (by EI and DI instructions)
- Interrupt source enable and disable settings in the corresponding peripheral control register(s)

ENABLE/DISABLE INTERRUPT INSTRUCTIONS (EI, DI)

The system mode register, SYM (DFH), is used to enable and disable interrupt processing.

SYM.3 is the enable and disable bit for global interrupt processing, which you can set by modifying SYM.3. An Enable Interrupt (EI) instruction must be included in the initialization routine that follows a reset operation in order to enable interrupt processing. Although you can manipulate SYM.3 directly to enable and disable interrupts during normal operation, we recommend that you use the EI and DI instructions for this purpose.

INTERRUPT PENDING FUNCTION TYPES

When the interrupt service routine has executed, the application program's service routine must clear the appropriate pending bit before the return from interrupt subroutine (IRET) occurs.

INTERRUPT PRIORITY

Because there is not a interrupt priority register in SAM87R1, the order of service is determined by a sequence of source which is executed in interrupt service routine.

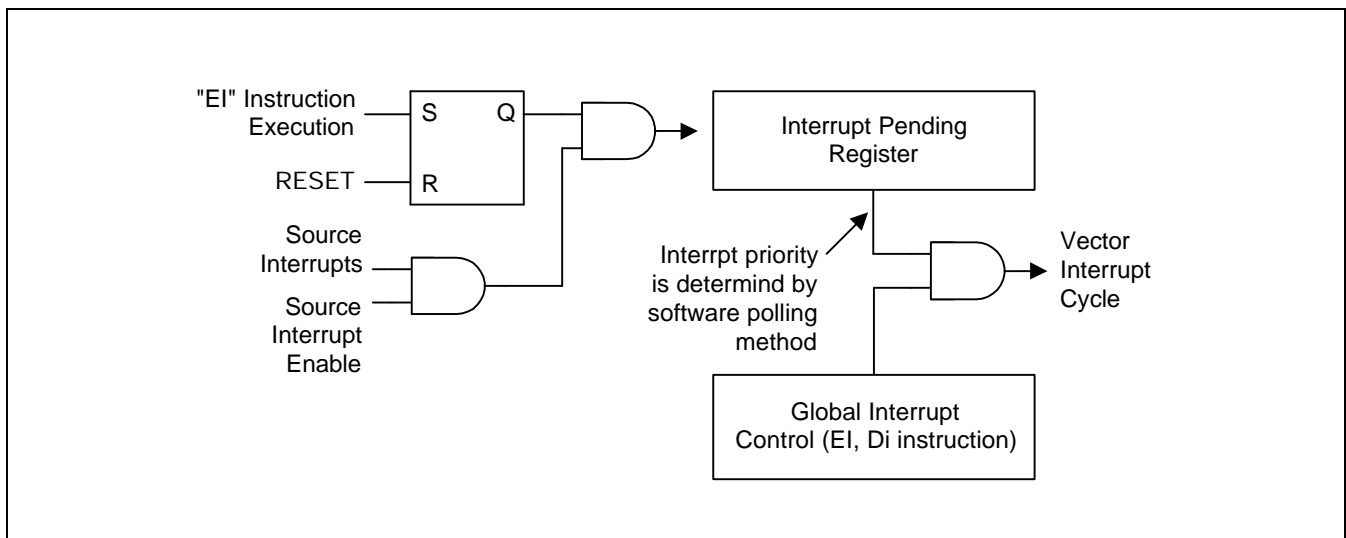


Figure 5-2. Interrupt Function Diagram

INTERRUPT SOURCE SERVICE SEQUENCE

The interrupt request polling and servicing sequence is as follows:

1. A source generates an interrupt request by setting the interrupt request pending bit to "1".
2. The CPU generates an interrupt acknowledge signal.
3. The service routine starts and the source's pending flag is cleared to "0" by software.
4. Interrupt priority must be determined by software polling method.

INTERRUPT SERVICE ROUTINES

Before an interrupt request can be serviced, the following conditions must be met:

- Interrupt processing must be enabled (EI, SYM.3 = "1")
- Interrupt must be enabled at the interrupt's source (peripheral control register)

If all of the above conditions are met, the interrupt request is acknowledged at the end of the instruction cycle. The CPU then initiates an interrupt machine cycle that completes the following processing sequence:

1. Reset (clear to "0") the global interrupt enable bit in the SYM register (DI, SYM.3 = "0") to disable all subsequent interrupts.
2. Save the program counter and status flags to stack.
3. Branch to the interrupt vector to fetch the service routine's address.
4. Pass control to the interrupt service routine.

When the interrupt service routine is completed, an Interrupt Return instruction (IRET) occurs. The IRET restores the PC and status flags and sets SYM.3 to "1"(EI), allowing the CPU to process the next interrupt request.

GENERATING INTERRUPT VECTOR ADDRESSES

The interrupt vector area in the ROM contains the address of the interrupt service routine. Vectored interrupt processing follows this sequence:

1. Push the program counter's low-byte value to stack.
2. Push the program counter's high-byte value to stack.
3. Push the FLAGS register values to stack.
4. Fetch the service routine's high-byte address from the vector address 0000H.
5. Fetch the service routine's low-byte address from the vector address 0001H.
6. Branch to the service routine specified by the 16-bit vector address.

S3C9664 INTERRUPT STRUCTURE

The S3C9664 microcontroller has thirteen peripheral interrupt sources:

- Timer 0 match interrupt
- Timer 0 overflow interrupt
- Timer 1 match interrupt
- Suspend interrupt
- Resume interrupt
- Internal RC interrupt
- USB reset interrupt
- Three Endpoint interrupts for Endpoint 0, Endpoint 1 and Endpoint 2
- Eight external interrupts for port 0, P0.0–P0.7
- Eight external interrupts for port 1, P1.1–P1.7
- Two external interrupts for port 2, P2.0–P2.1

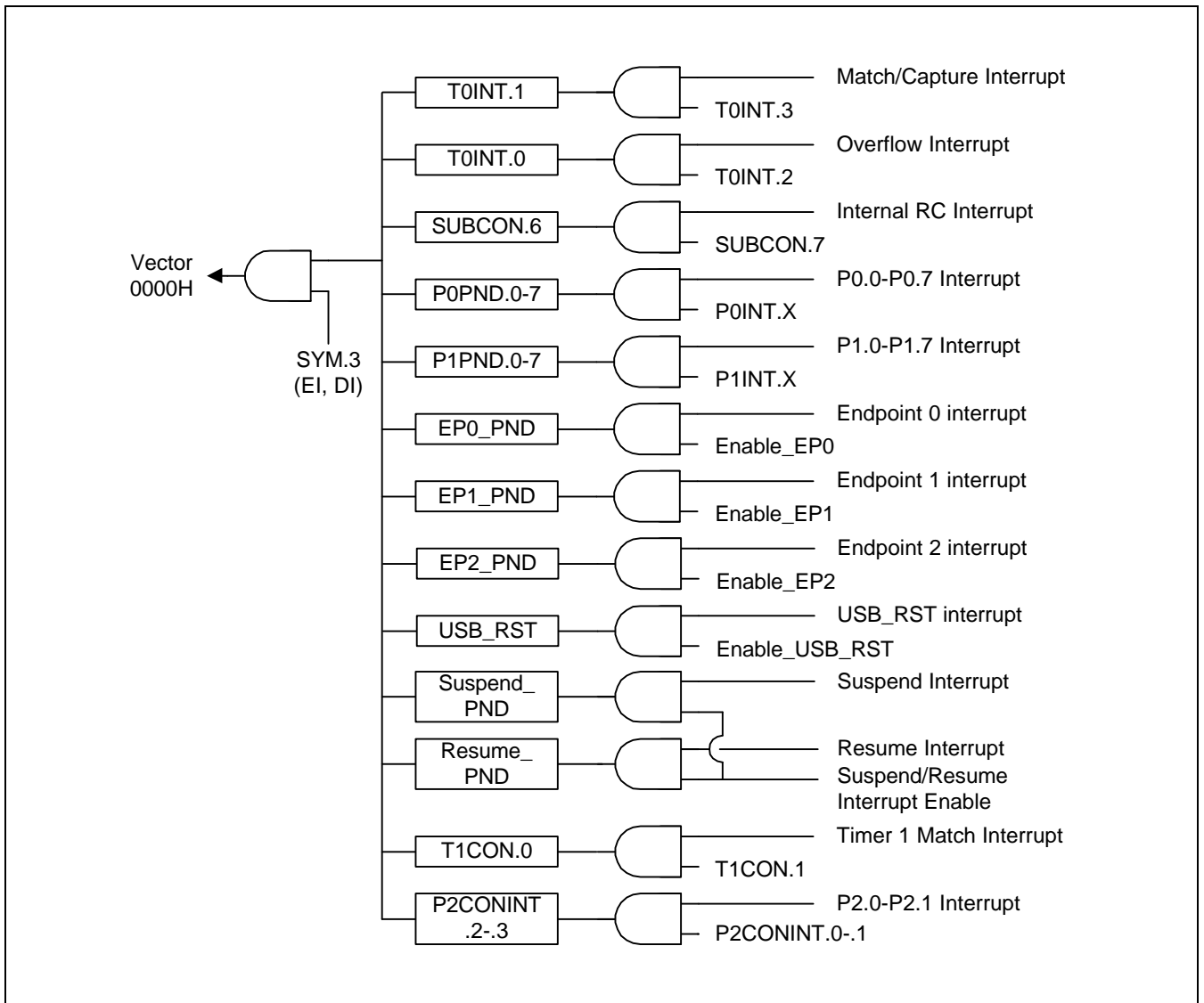


Figure 5-3. S3C9664 Interrupt Structure

7

CLOCK CIRCUIT

OVERVIEW

The S3C9664 has two oscillation circuit options, a crystal/ceramic oscillation and an external clock source. The crystal or ceramic oscillation source provides a maximum 6 MHz clock. The X_{IN} and X_{OUT} pins connect the oscillation source to the on-chip clock circuit. External clock and crystal/ceramic oscillator circuits are shown in Figures 7-1 and 7-2.

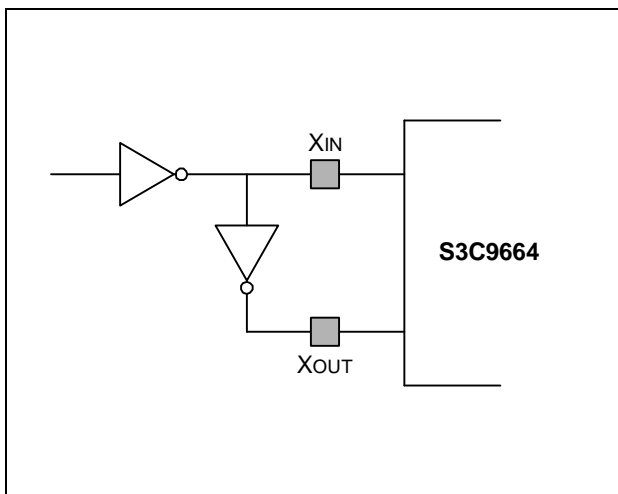


Figure 7-1. External Oscillator

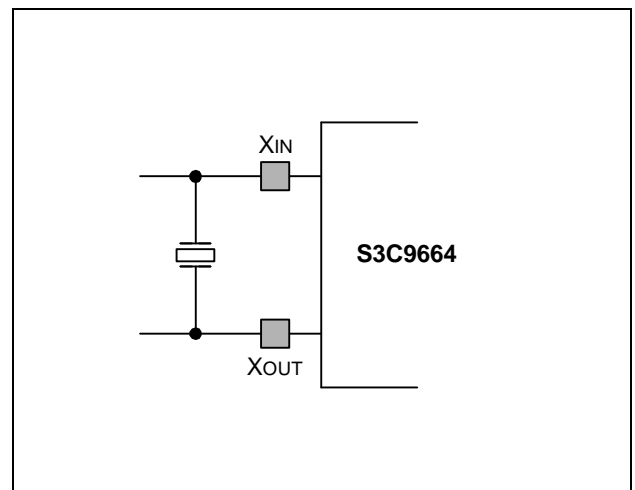


Figure 7-2. Main Oscillator Circuit
(Crystal/Ceramic Oscillator)

MAIN OSCILLATOR LOGIC

To increase processing speed and to reduce clock noise, non-divided logic is implemented for the main oscillator circuit. For this reason, very high resolution waveforms (square signal edges) must be generated in order for the CPU to efficiently process logic operations.

CLOCK STATUS DURING POWER-DOWN MODES

The two power-down modes, Stop mode and Idle mode, affect clock oscillation as follows:

- In Stop mode, the main oscillator "freezes," halting the CPU and peripherals. The contents of the register file and current system register values are retained. RESET operation releases the Stop mode, and starts the oscillator.
- In Idle mode, the internal clock signal is gated off to the CPU, but not to interrupt control and the timer. The current CPU status is preserved, including stack pointer, program counter, and flags. Data in the register file is retained. Idle mode is released by a RESET or by an interrupt (external or internally-generated).

SYSTEM CLOCK CONTROL REGISTER (CLKCON)

The system clock control register, CLKCON, is located in location D4H. It is read/write addressable and has the following functions:

- Oscillator IRQ wake-up function enable/disable (CLKCON.7)
- Oscillator frequency divide-by value: non-divided, 2, 8, or 16 (CLKCON.4 and CLKCON.3)

The CLKCON register controls whether or not an external interrupt can be used to trigger a Stop mode release (This is called the "IRQ wake-up" function). The IRQ wake-up enable bit is CLKCON.7.

After a RESET, the external interrupt oscillator wake-up function is enabled, the main oscillator is activated, and the $f_{OSC}/16$ (the slowest clock speed) is selected as the CPU clock. If necessary, you can then increase the CPU clock speed to f_{OSC} , $f_{OSC}/2$ or $f_{OSC}/8$.

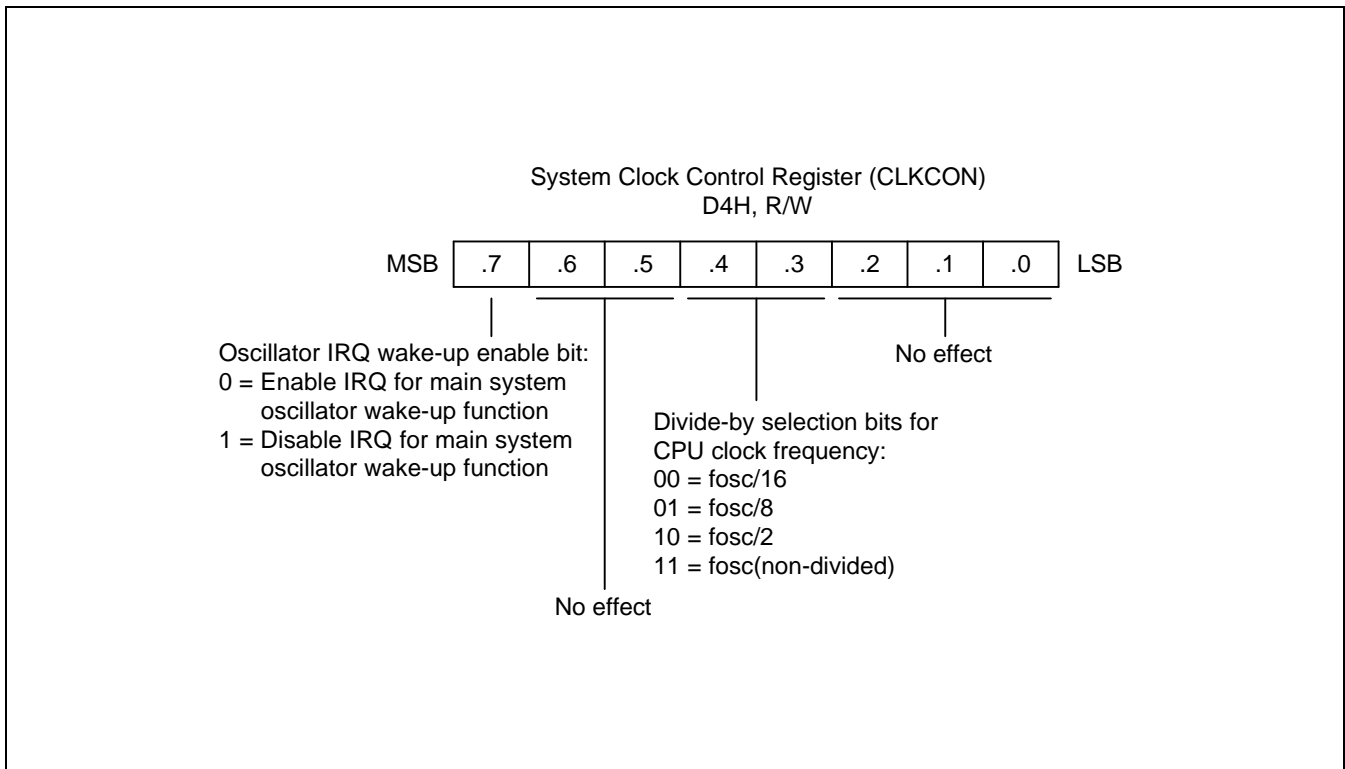


Figure 7-3. System Clock Control Register (CLKCON)

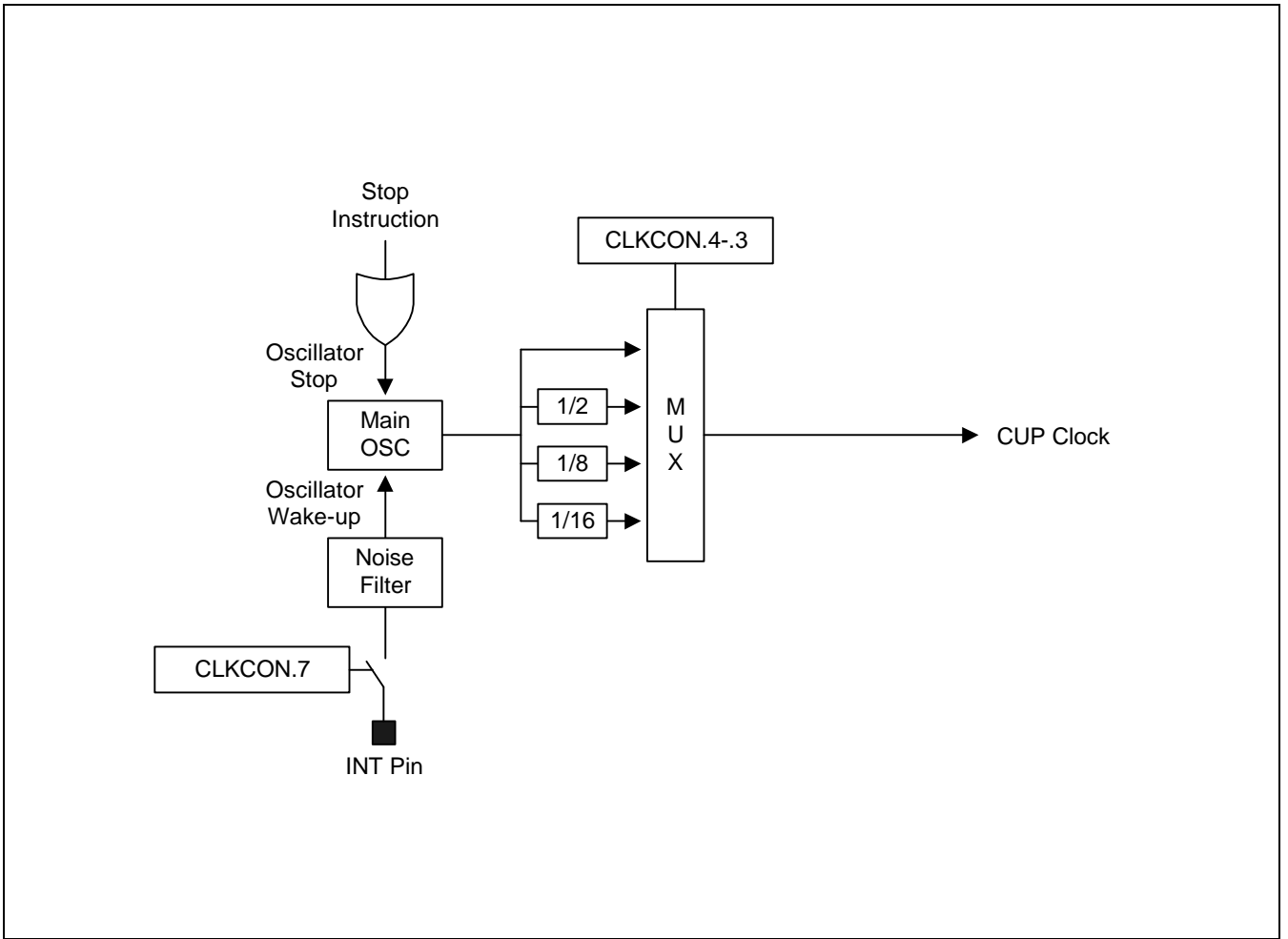


Figure 7-4. System Clock Circuit Diagram

8

RESET and POWER-DOWN

SYSTEM RESET

OVERVIEW

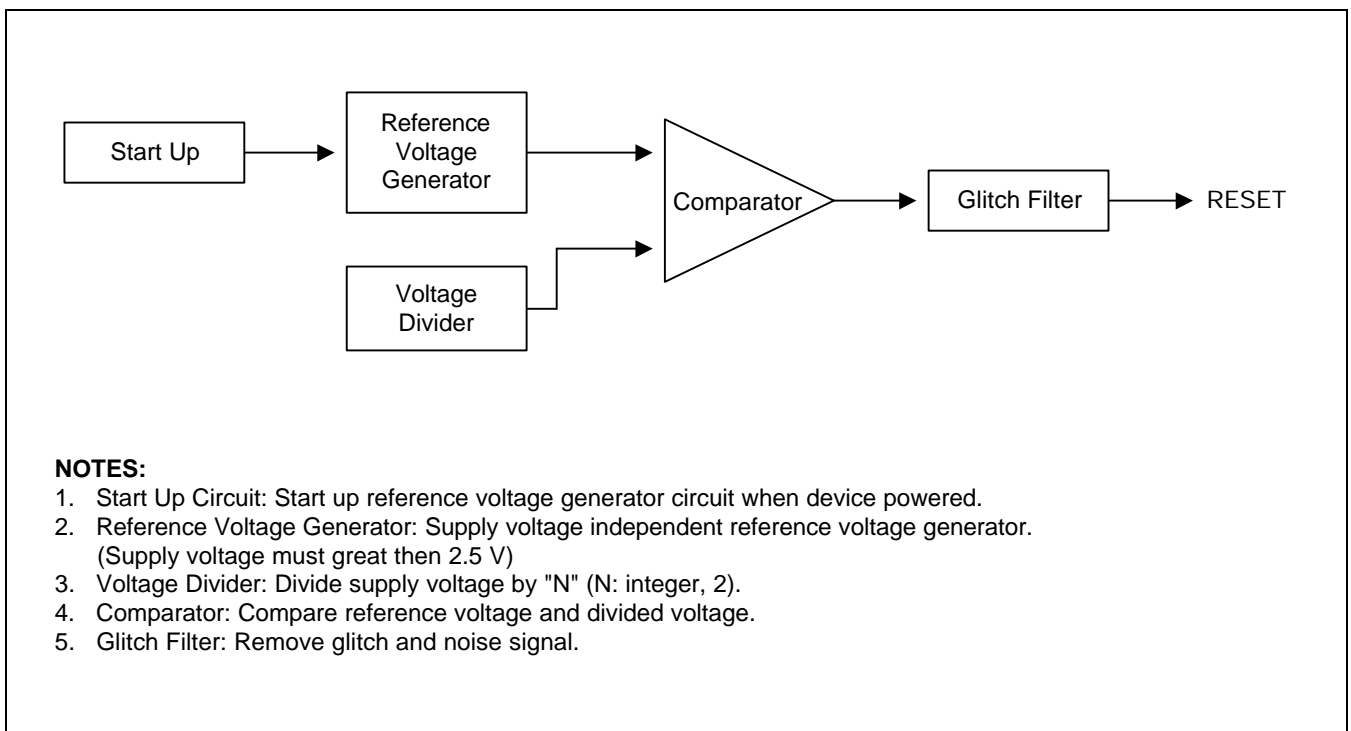


Figure 8-1. LVD Characteristics

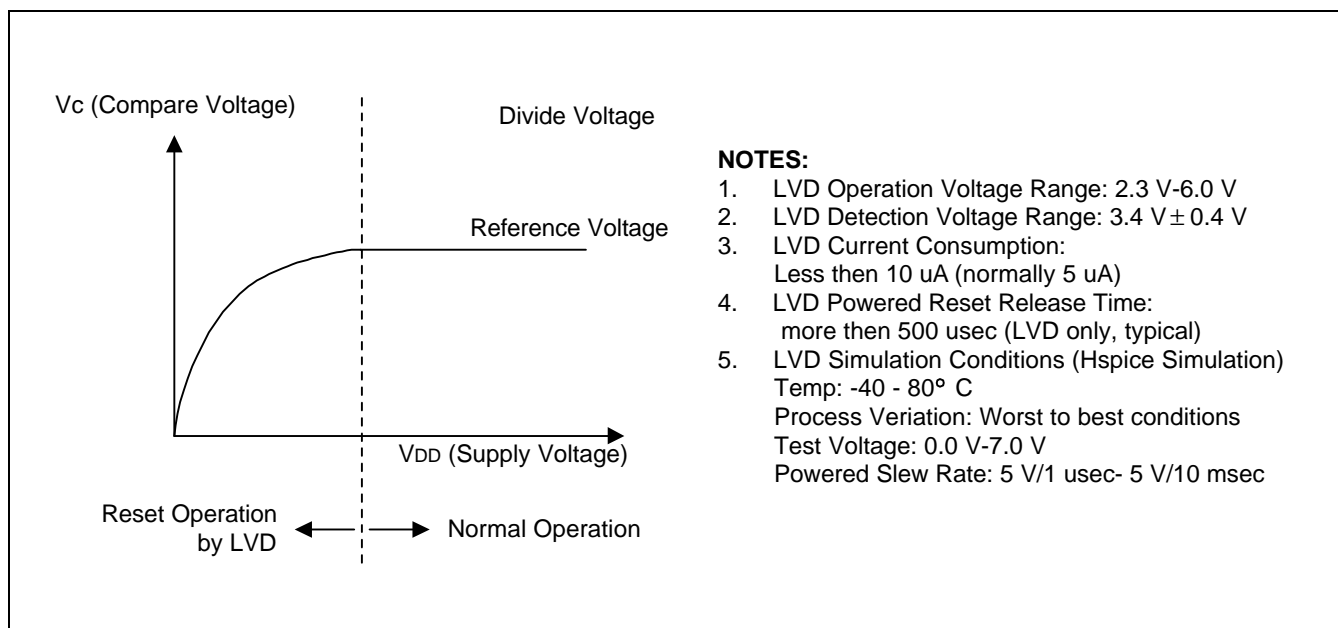


Figure 8-2. LVD Architecture

The following sequence of events occur during a reset operation:

- All interrupts are disabled.
- The watchdog function (basic timer) is enabled.
- Ports 0 and 1 are set to Schmitt trigger input mode and all pull-up resistors are disabled.
- Peripheral control and data registers are disabled and reset to their initial values.
- The program counter is loaded with the ROM reset address, 0100H.
- When the programmed oscillation stabilization time interval has elapsed, the address stored in ROM location 0100H (and 0101H) is fetched and executed.

NOTE

To program the duration of the oscillation stabilization interval, you must make the appropriate settings to the basic timer control register, BTCON, before entering Stop mode. Also, if you do not want to use the basic timer watchdog function (which causes a system reset if a basic timer counter overflow occurs), you can disable it by writing '1010B' to the upper nibble of BTCON.

POWER-DOWN MODES

STOP MODE

Stop mode is invoked by the instruction STOP (opcode 7FH). In Stop mode, the operation of the CPU and all peripherals is halted. That is, the on-chip main oscillator stops and the supply current is reduced to less than 120 μ A. All system functions are halted when the clock "freezes," but data stored in the internal register file is retained. Stop mode can be released in one of two ways: by a RESET signal or by an external interrupt.

Using RESET to Release Stop Mode

Stop mode is released when the RESET signal is released and returns to High level. All system and peripheral control registers are then reset to their default values and the contents of all data registers are retained. RESET operation automatically selects a slow clock (1/16) because CLKCON.3 and CLKCON.4 are cleared to '00B'. After the oscillation stabilization interval has elapsed, the CPU executes the system initialization routine by fetching the 16-bit address stored in ROM locations 0100H and 0101H.

Using an External Interrupt to Release Stop Mode

Only external interrupts with an RC-delay noise filter circuit can be used to release Stop mode (Clock-related external interrupts cannot be used). External interrupts in the KS86C6504/6508 interrupt structure does not meet this criteria.

Note that when Stop mode is released by an external interrupt, the current values in system and peripheral control registers are not changed. When you use an interrupt to release Stop mode, the CLKCON.3 and CLKCON.4 register values remain unchanged, and the currently selected clock value is used. If you use an external interrupt for Stop mode release, you can also program the duration of the oscillation stabilization interval. To do this, you must make the appropriate control and clock settings *before* entering Stop mode.

The external interrupt is serviced when the Stop mode release occurs. Following the IRET from the service routine, the instruction immediately following the one that initiated Stop mode is executed.

NOTE

Do not use the STOP mode when external clock source is being used as the oscillation circuit option.

IDLE MODE

Idle mode is invoked by the instruction IDLE (opcode 6FH). In Idle mode, CPU operations are halted while select peripherals remain active. During Idle mode, the internal clock signal is gated off to the CPU, but not to interrupt logic and timer/counters. Port pins retain the mode (input or output) they had at the time Idle mode was entered.

There are two ways to release Idle mode:

1. Execute RESET. All system and peripheral control registers are reset to their default values and the contents of all data registers are retained. The reset automatically selects a slow clock (1/16) because CLKCON.3 and CLKCON.4 are cleared to '00B'. If interrupts are masked, RESET is the only way to release Idle mode.
2. Activate any enabled interrupt, causing Idle mode to be released. When you use an interrupt to release Idle mode, the CLKCON.3 and CLKCON.4 register values remain unchanged, and the currently selected clock value is used. The interrupt is then serviced. Following the IRET from the service routine, the instruction immediately following the one that initiated Idle mode is executed.

NOTE

Only external interrupts that are not clock-related can be used to release Stop mode. To release Idle mode, however, any type of interrupt (that is, internal or external) can be used.

HARDWARE RESET VALUES

Tables 8-1 through 8-3 list the values for CPU and system registers, peripheral control registers, and peripheral data registers following a reset operation in normal operating mode. The following notation is used in these tables to represent specific reset values:

- A "1" or a "0" shows the RESET bit value as logic one or logic zero, respectively.
- An 'x' means that the bit value is undefined following RESET.
- A dash ('-') means that the bit is either not used or not mapped.

Table 8-1. Register Map and Reset Status (Page 0)

Register Name	Mnemonic	Address	Bit Values After RESET								
			7	6	5	4	3	2	1	0	
General purpose register file & stack area	–	00–7FH	x	x	x	x	x	x	x	x	x
Working register area	–	C0H–CFH	x	x	x	x	x	x	x	x	x
Timer 0 Counter Register	T0CNT	D0H	0	0	0	0	0	0	0	0	0
Timer 0 Data Register	T0DATA	D1H	1	1	1	1	1	1	1	1	1
Timer 0 Control Register	T0CON	D2H	0	0	0	0	0	0	0	0	0
Timer 1 Interrupt Control Register	T0INT	D3H	0	0	0	0	0	0	0	0	0
Clock Control Register	CLKCON	D4H	0	0	0	0	0	0	0	0	0
System Flags Register	FLAGS	D5H	0	0	0	0	–	–	–	–	–
A/D converter data register(High byte)	ADDATAH	D6H	x	x	x	x	x	x	x	x	x
A/D converter data register (Low byte)	ADDATA L	D7H	–	–	–	–	–	–	–	x	x
A/D control register	ADCON	D8H	–	0	0	0	0	0	0	0	0
Stack Pointer Register	SP	D9H	x	x	x	x	x	x	x	x	x
Port 2 Data Register	P2	DAH	0	0	0	0	0	0	0	0	0
Location DBH is not mapped.											
Basic Timer Control Register	BTCON	DCH	0	0	0	0	0	0	0	0	0
Basic Timer Counter	BTCNT	DDH	0	0	0	0	0	0	0	0	0
Location DEH is not mapped.											
System Mode Register	SYM	DFH	–	–	–	–	0	0	0	0	0

NOTE: – : Not mapped, x: Undefined

Table 8-1. Register Map and Reset Status (Page 0) (Continued)

Register Name	Mnemonic	Address	Bit Values After RESET							
			7	6	5	4	3	2	1	0
Port 0 data register	P0	E0H	0	0	0	0	0	0	0	0
Port 1 data register	P1	E1H	0	0	0	0	0	0	0	0
Timer 1 Counter Register	T1CNT	E2H	0	0	0	0	0	0	0	0
Timer 1 Control Register	T1CON	E3H	0	0	0	0	0	0	0	0
Port 0 Pull-up/down register (low byte)	P0PURL	E4H	0	0	0	0	0	0	0	0
Port 0 Pull-up/down register (high byte)	P0PURH	E5H	0	0	0	0	0	0	0	0
Port 0 control register (low byte)	P0CONL	E6H	0	0	0	0	0	0	0	0
Port 0 control register (high byte)	P0CONH	E7H	0	0	0	0	0	0	0	0
Port 1 control register (low byte)	P1CONL	E8H	0	0	0	0	0	0	0	0
Port 1 control register (high byte)	P1CONH	E9H	0	0	0	0	0	0	0	0
Port 0 interrupt enable register	P0INT	EAH	0	0	0	0	0	0	0	0
Port 0 interrupt pending register	P0PND	EBH	0	0	0	0	0	0	0	0
Port 1 interrupt enable register	P1INT	ECH	0	0	0	0	0	0	0	0
Port 1 interrupt pending register	P1PND	EDH	0	0	0	0	0	0	0	0
Timer 1 DATA register	T1DATA	EEH	1	1	1	1	1	1	1	1
Port 2 control/interrupt register	P2CONINT	EFH	0	0	0	0	0	0	0	0
USB function address register	FADDR	F0H	0	0	0	0	0	0	0	0
USB control endpoint status register	EP0CSR	F1H	0	0	0	0	0	0	0	0
USB interrupt endpoint 1 status register	EP1CSR	F2H	0	1	0	0	0	0	0	0
USB control endpoint byte count register	EP0BCNT	F3H	0	0	0	0	0	0	0	0
USB control endpoint FIFO register	EP0FIFO	F4H	x	x	x	x	x	x	x	x
USB interrupt endpoint 1 FIFO register	EP1FIFO	F5H	x	x	x	x	x	x	x	x
USB interrupt pending register	USBPND	F6H	0	0	0	0	0	0	0	0
USB interrupt enable register	USBINT	F7H	0	0	0	0	1	0	1	1
USB power management register	PWRMGR	F8H	0	0	0	0	0	0	0	0
USB interrupt endpoint 2 status register	EP2CSR	F9H	0	1	0	0	0	0	0	0
USB interrupt endpoint 2 FIFO register	EP2FIFO	FAH	x	x	x	x	x	x	x	x
Endpoint mode register	EPMODE	FBH	0	0	0	0	0	0	0	0
USB interrupt endpoint 1 byte count register	EP1BCNT	FCH	0	0	0	0	0	0	0	0
USB control endpoint 2 byte count register	EP2BCNT	FDH	0	0	0	0	0	0	0	0
USB control register	USBCON	FEH	0	0	0	0	0	0	0	0
Sub oscillator control register	SUBCON	FFH	0	0	0	0	0	0	0	0

Table 8-2. Register Map and Reset Status (Page 1)

Register Name	Mnemonic	Address	Bit Values After RESET							
			7	6	5	4	3	2	1	0
USB signal control register	XCON	FEH	0	0	0	0	0	0	0	0

9

I/O PORTS

OVERVIEW

The S3C9664 has three I/O ports (Port 0, Port 1, Port2 only on USB disabled), 18 pins total. You can access these ports directly by writing or reading port data register addresses.

Table 9-1. S3C9664 Port Configuration Overview

Port	Function Description	Programmability
P0.0-P0.7	Bit-programmable I/O port for schmitt trigger input, push-pull output and N-Ch open drain output. Pull-up/pull-down resistors are assignable by software. Port 1 pins can also be used as external interrupt.	Bit
P1.0 – P1.5	Bit-programmable I/O port for schmitt trigger input, schmitt trigger input with pull-up and N-Ch open drain output. Port 1 pins can also be used as AD converter Channel.	Bit
P1.6 – P1.7	Bit-programmable I/O port for schmitt trigger input, schmitt trigger input with pull-up and N-Ch open drain output and push-pull output.	Bit
P2.0/D- – P2.1/D+	Bit-programmable I/O port for schmitt trigger input, schmitt trigger input with pull-up and N-Ch open drain output and push-pull output. Port 2 can be individually configured as external interrupt inputs. Also it can be configured as an USB ports.	Bit

PORT DATA REGISTERS

Table 9-2 gives you an overview of the port data register names, locations, and addressing characteristics. Data registers for ports 0 and 1 have the structure shown in Figure 9-1.

Table 9-2. Port Data Register Summary

Register Name	Mnemonic	Hex	R/W
Port 0 data register	P0	E0H	R/W
Port 1 data register	P1	E1H	R/W
Port 2 data register	P2	FAH	R/W

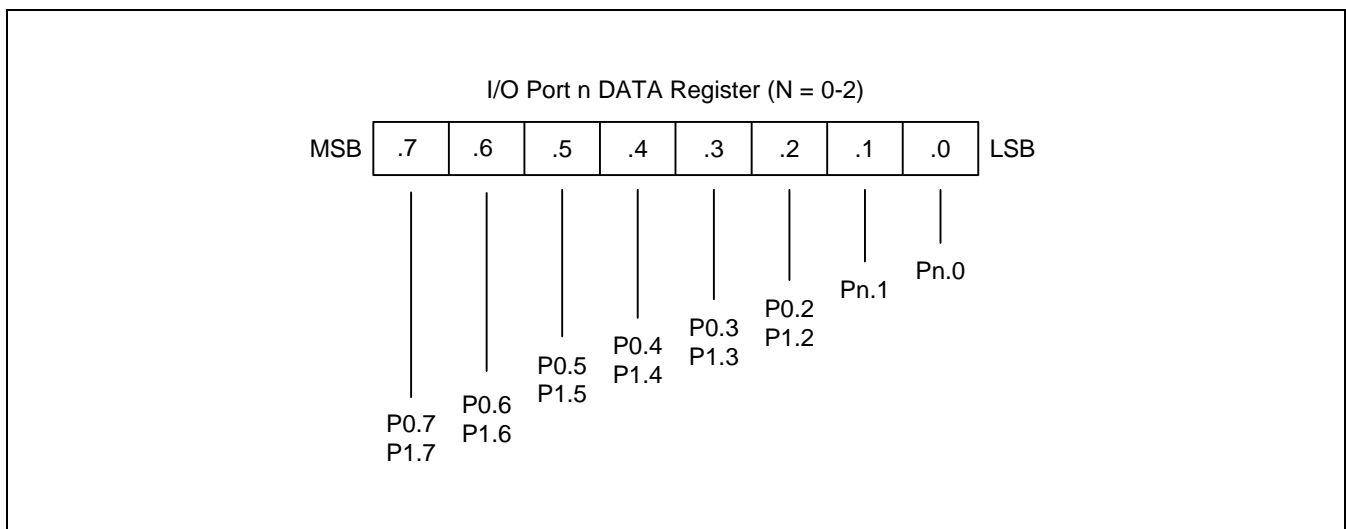


Figure 9-1. Port Data Register Format

PORT 0

Port 0 is bit-programmable, general-purpose, I/O ports. You can configure schmitt trigger input mode with rising edge external interrupt or falling edge external interrupt mode, N-channel open drain output and push pull output mode. Meanwhile, pull-up and pull-down resistor can be configured only on input modes.

In normal operating mode, a reset clears the port 0 control registers (P0CONH, P0CONL, P0PURH, P0PURL) to "00H", configuring P0.0–P0.7 as schmitt trigger input, falling edge external interrupt with pull-up resistor

Port 0 is accessed directly by writing or reading the port 0 data register. P0 (E0H, Page 0).

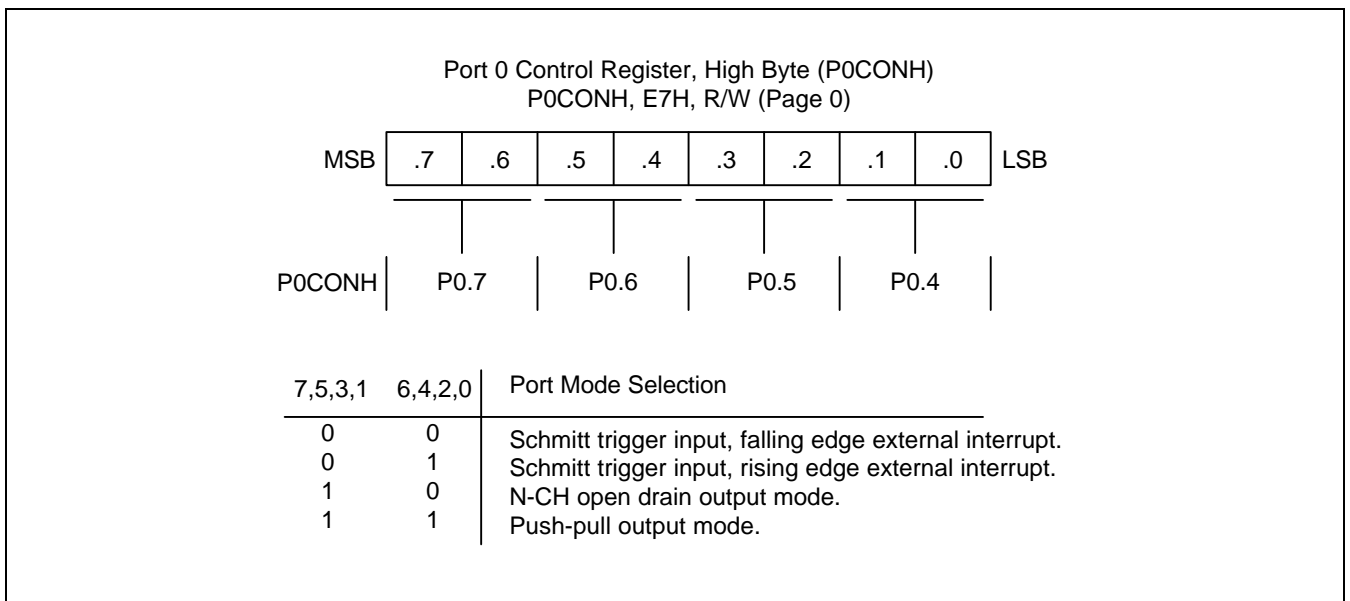


Figure 9-2. Port 0 Control Registers (P0CONH)

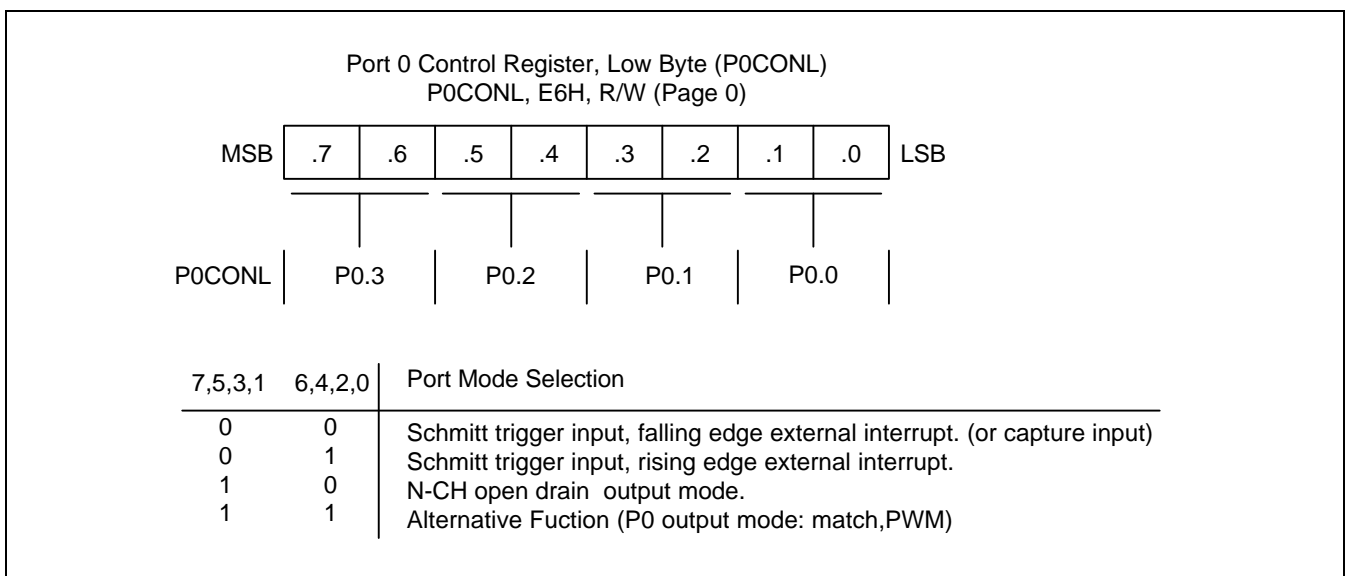


Figure 9-3. Port 0 Control Registers (P0CONL)

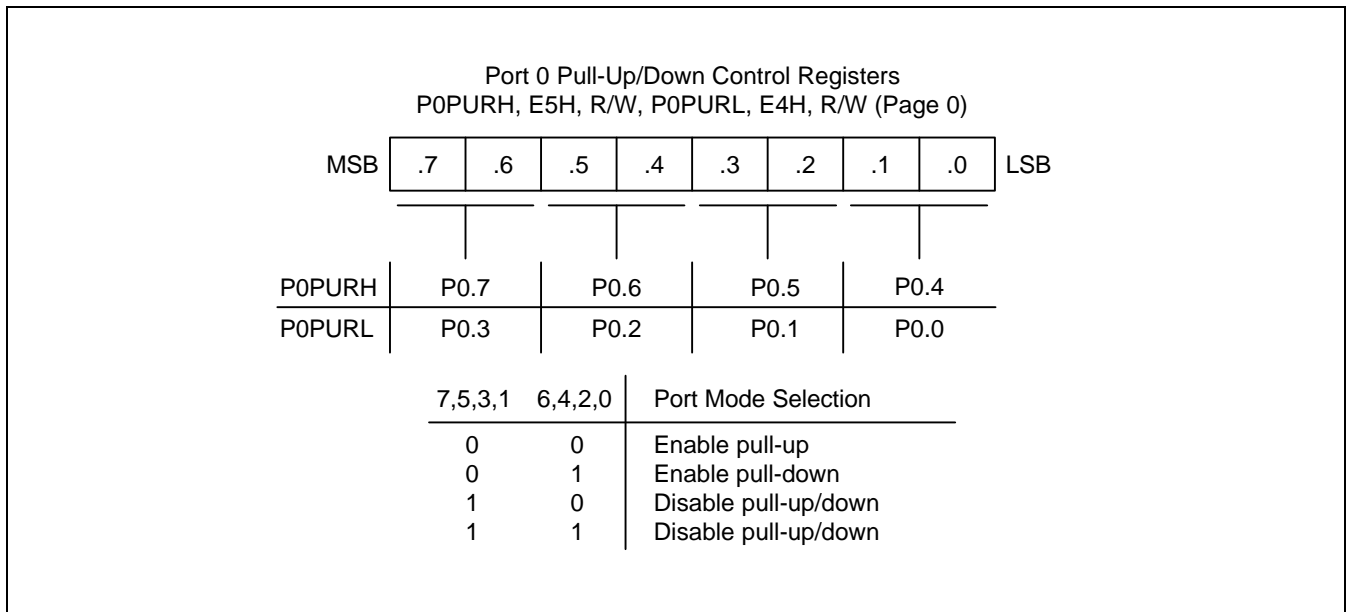


Figure 9-4. Port 0 Pull-Up/Down Control Registers (P0PURH/P0PURL)

PORT 1

Port 1 is bit-programmable, general-purpose, I/O ports. You can configure schmitt trigger input mode, rising edge external interrupt with pull-up or falling edge external interrupt mode, N-channel open drain output (only for P1.7, P1.6) A/D converter input (only for P1.1-P1.5) and push pull output mode .

In normal operating mode, a reset clears the port 0 control registers (P1CONH, P1CONL) to "00H", configuring P1.0–P1.7 as schmitt trigger input, falling edge external interrupt with pull-up resistor.

Port 1 is accessed directly by writing or reading the port 1 data register. P1 (E1H, Page 0).

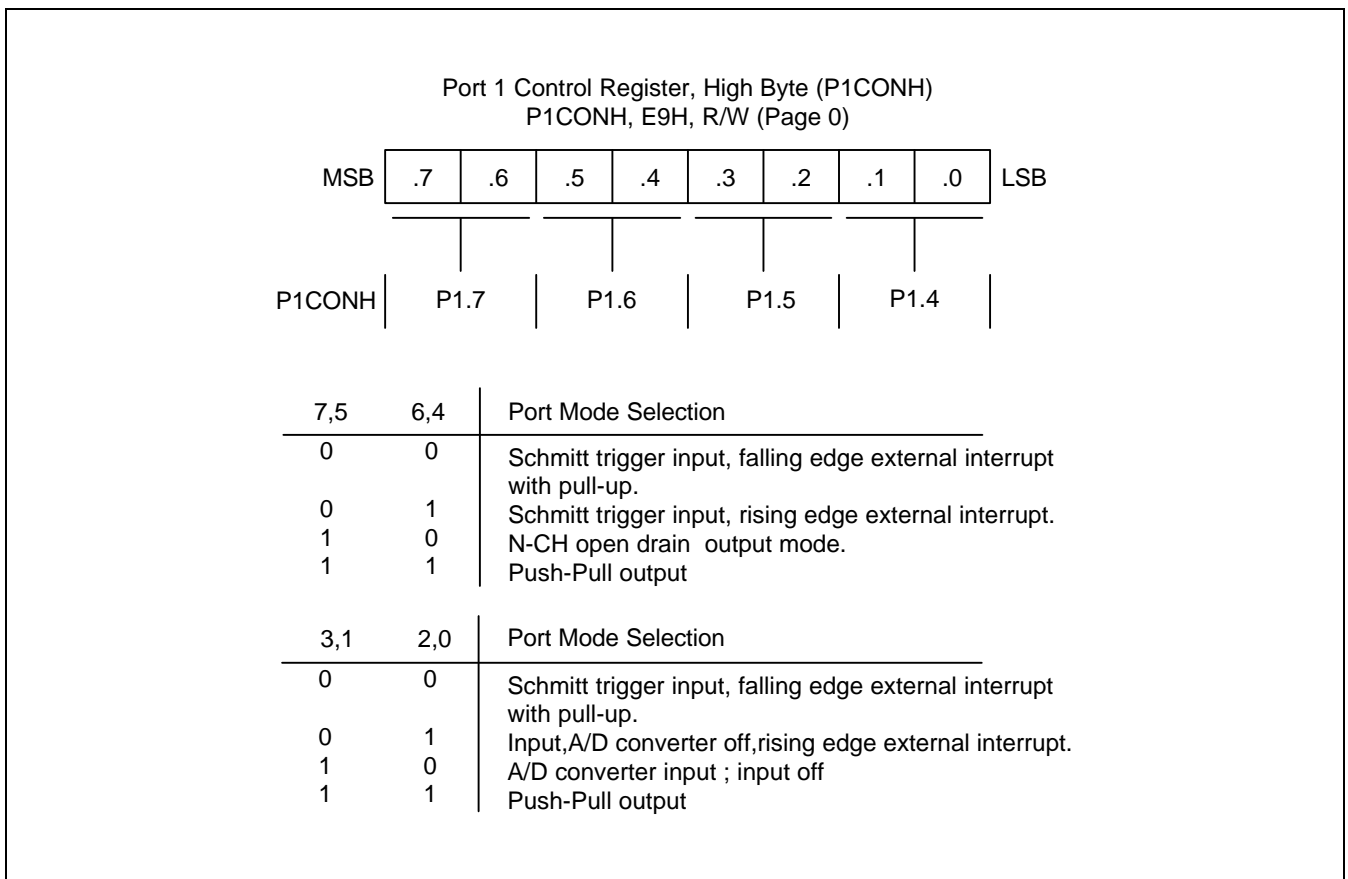


Figure 9-5. Port 1 Control Register High Byte (P1CONH)

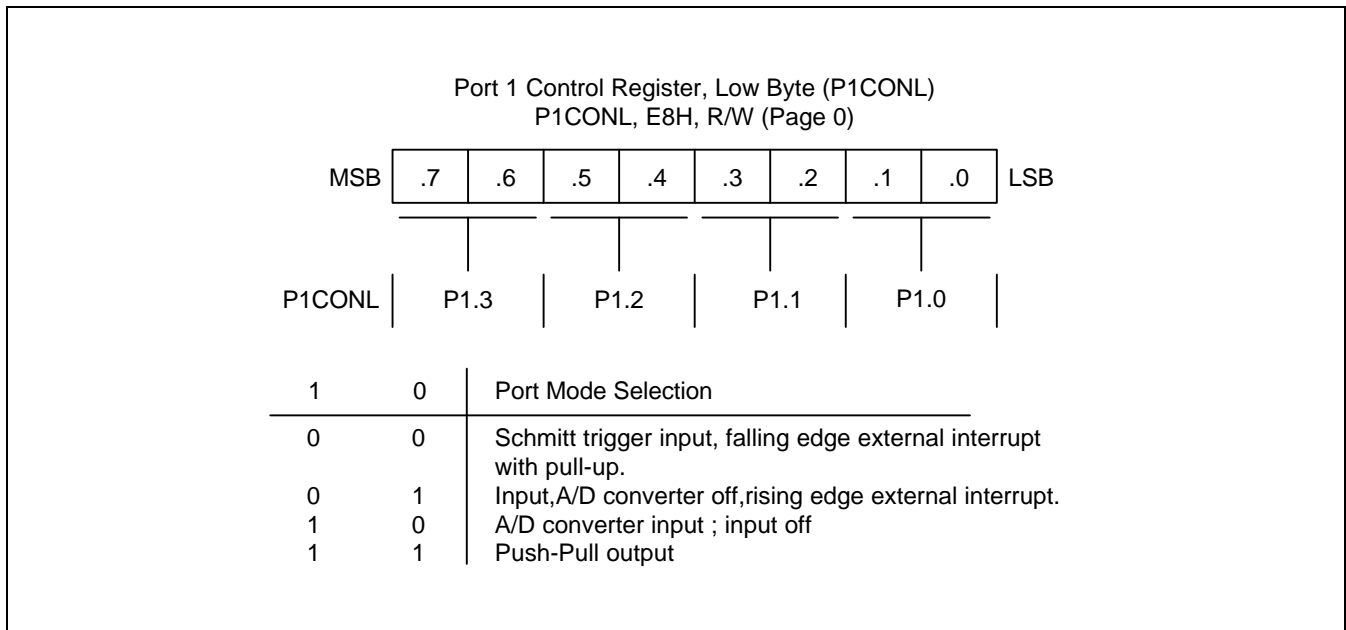


Figure 9-6. Port 1 Control Register Low Byte (P1CONL)

PORT 2

Port 2 can be configured bit-programmable, general-purpose, I/O ports, only when USB ports are disabled (USBSEL.0 = 0). Otherwise (USBSEL.1=1), port 2 is used for D+/D-.

However, in general purpose I/O port mode. You can configure schmitt trigger input mode, rising edge external interrupt and schmitt trigger input falling edge external interrupt mode with pull-up, N-channel open drain output and push pull output mode with pull-up.

In normal operating mode, a reset clears the port 2 control registers (P2CONINT) to "00H", configuring P2.0–P2.1 as schmitt trigger input, rising edge external interrupt with pull-up resistor.

Port 2 is accessed directly by writing or reading the port 2 data register. P2 (EFH, Page 0).

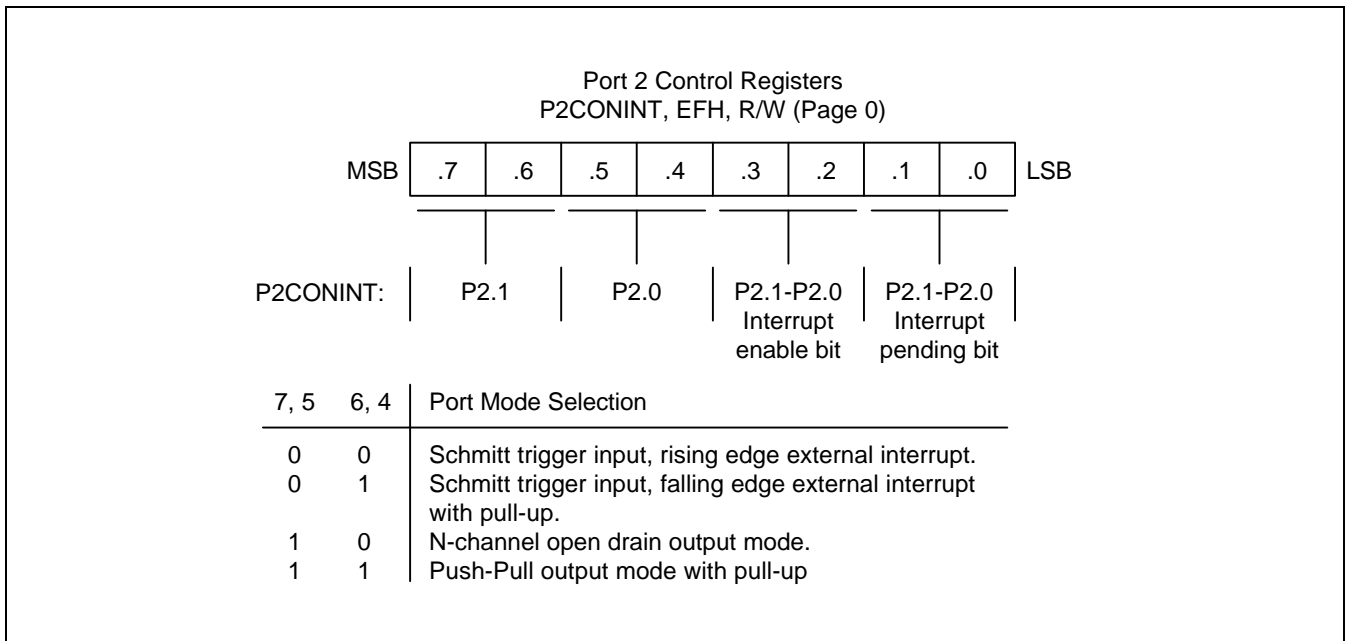


Figure 9-7. Port Control Registers (P2CONINT)

10

BASIC TIMER and TIMER 0

MODULE OVERVIEW

The S3C9664 has two default timers: an 8-bit *basic timer* and one 8-bit general-purpose timer/counter. The 8-bit timer/counter is called *timer 0*.

Basic Timer (BT)

You can use the basic timer (BT) in two different ways:

- As a watchdog timer to provide an automatic reset mechanism in the event of a system malfunction
- To signal the end of the required oscillation stabilization interval after a reset or a Stop mode release.

The functional components of the basic timer block are:

- Clock frequency divider (f_{OSC} divided by 4096, 1024, or 128) with multiplexer
- 8-bit basic timer counter, BTCNT (DDH, read-only)
- Basic timer control register, BTCON (DCH, read/write)

Timer 0

Timer 0 has three operating modes, one of which you select by the appropriate T0CON setting:

- Interval timer mode
- Capture mode with a rising or falling edge trigger at the T0 pin
- PWM mode

Timer 0 has the following functional components:

- Clock frequency divider (f_{OSC} divided by 4096, 256, 8, or 1) with multiplexer
- 8-bit counter (T0CNT), 8-bit comparator, and 8-bit reference data register (T0DATA)
- I/O pin (P1.0/T0) for timer 0 capture input or match output
- Timer 0 overflow interrupt and match/capture interrupt generation
- Timer 0 control register, T0CON
- Timer 0 interrupt control register, T0INT

BASIC TIMER CONTROL REGISTER (BTCON)

The basic timer control register, BTCON, is used to select the input clock frequency, to clear the basic timer counter and frequency dividers, and to enable or disable the watchdog timer function. It is located in set 1, address D3H, and is read/write addressable using Register addressing mode.

A reset clears BTCON to "00H". This enables the watchdog function and selects a basic timer clock frequency of $f_{OSC}/4096$. To disable the watchdog function, you must write the signature code "1010B" to the basic timer register control bits BTCON.7–BTCON.4.

The 8-bit basic timer counter, BTCNT, can be cleared at any time during the normal operation by writing a "1" to BTCON.1. To clear the frequency dividers for both the basic timer input clock and the timer 0 clock, you write a "1" to BTCON.0.

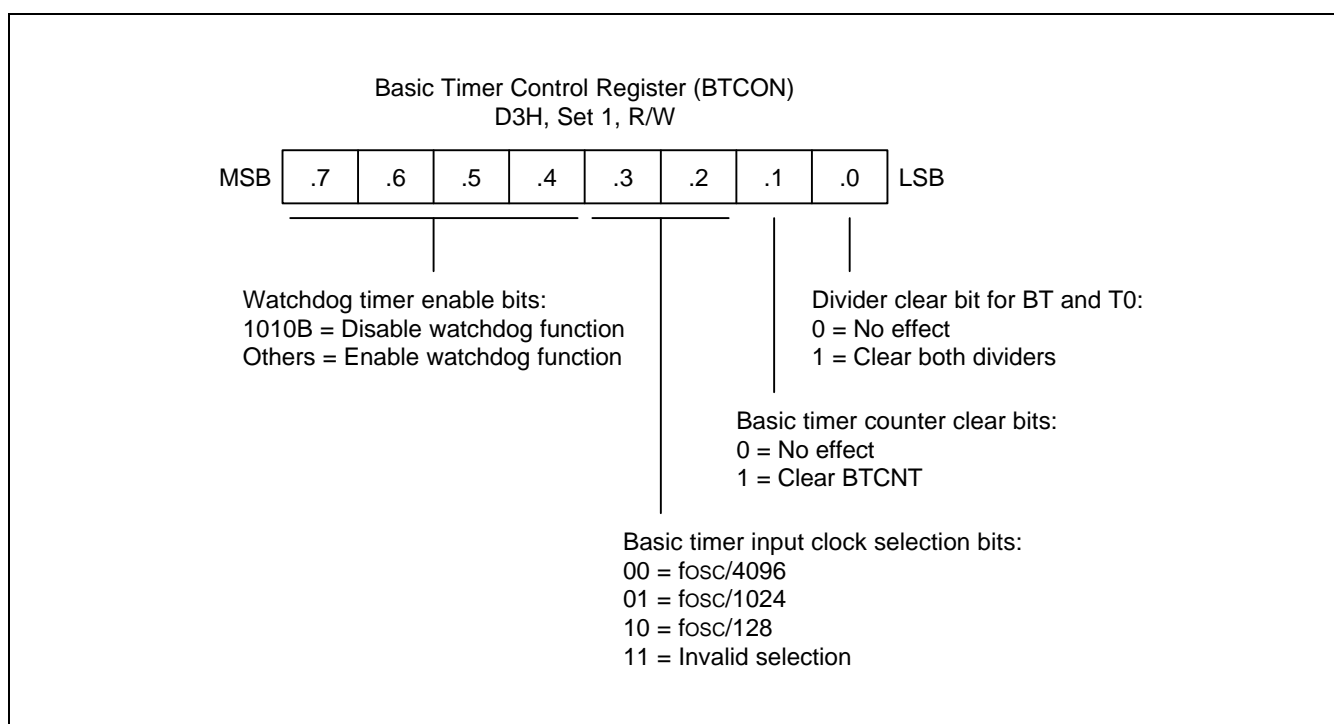


Figure 10-1. Basic Timer Control Register (BTCON)

BASIC TIMER FUNCTION DESCRIPTION

Watchdog Timer Function

You can program the basic timer overflow signal (BTOVF) to generate a reset by setting BTCON.7–BTCON.4 to any value other than "1010B". (The "1010B" value disables the watchdog function.) A reset clears BTCON to "00H", automatically enabling the watchdog timer function. A reset also selects the CPU clock (as determined by the current CLKCON register setting) divided by 4096 as the Basic timer clock.

A reset whenever a basic timer counter overflow occurs. During the normal operation, the application program must prevent the overflow, and the accompanying reset operation, from occurring. To do this, the BTCNT value must be cleared (by writing a "1" to BTCON.1) at regular intervals.

If a system malfunction occurs due to circuit noise or some other error condition, the Basic timer counter clear operation will not be executed and a basic timer overflow will occur, initiating a reset. In other words, during the normal operation, the basic timer overflow loop (a bit 7 overflow of the 8-bit basic timer counter, BTCNT) is always broken by a BTCNT clear instruction. If a malfunction does occur, a reset is triggered automatically.

Oscillation Stabilization Interval Timer Function

You can also use the basic timer to program a specific oscillation stabilization interval after a reset or when Stop mode has been released by an external interrupt.

In Stop mode, whenever a reset or an external interrupt occurs, the oscillator starts. The BTCNT value then starts increasing at the rate of $f_{OSC}/4096$ (for reset), or at the rate of the preset clock source (for an external interrupt). When BTCNT.4 is set, a signal is generated to indicate that the stabilization interval has elapsed and to gate the clock signal off to the CPU so that it can resume normal operation.

In summary, the following events occur when Stop mode is released:

1. During Stop mode, a power-on reset or an external interrupt occurs to trigger the Stop mode release and oscillation starts.
2. If a power-on reset occurred, the basic timer counter will increase at the rate of $f_{OSC}/4096$. If an external interrupt is used to release Stop mode, the BTCNT value increases at the rate of the preset clock source.
3. Clock oscillation stabilization interval begins and continues until bit 4 of the basic timer counter is set.
4. When a BTCNT.4 is set, normal CPU operation resumes.
5. Figure 10-2 and 10-3 shows the oscillation stabilization time on RESET and STOP mode release

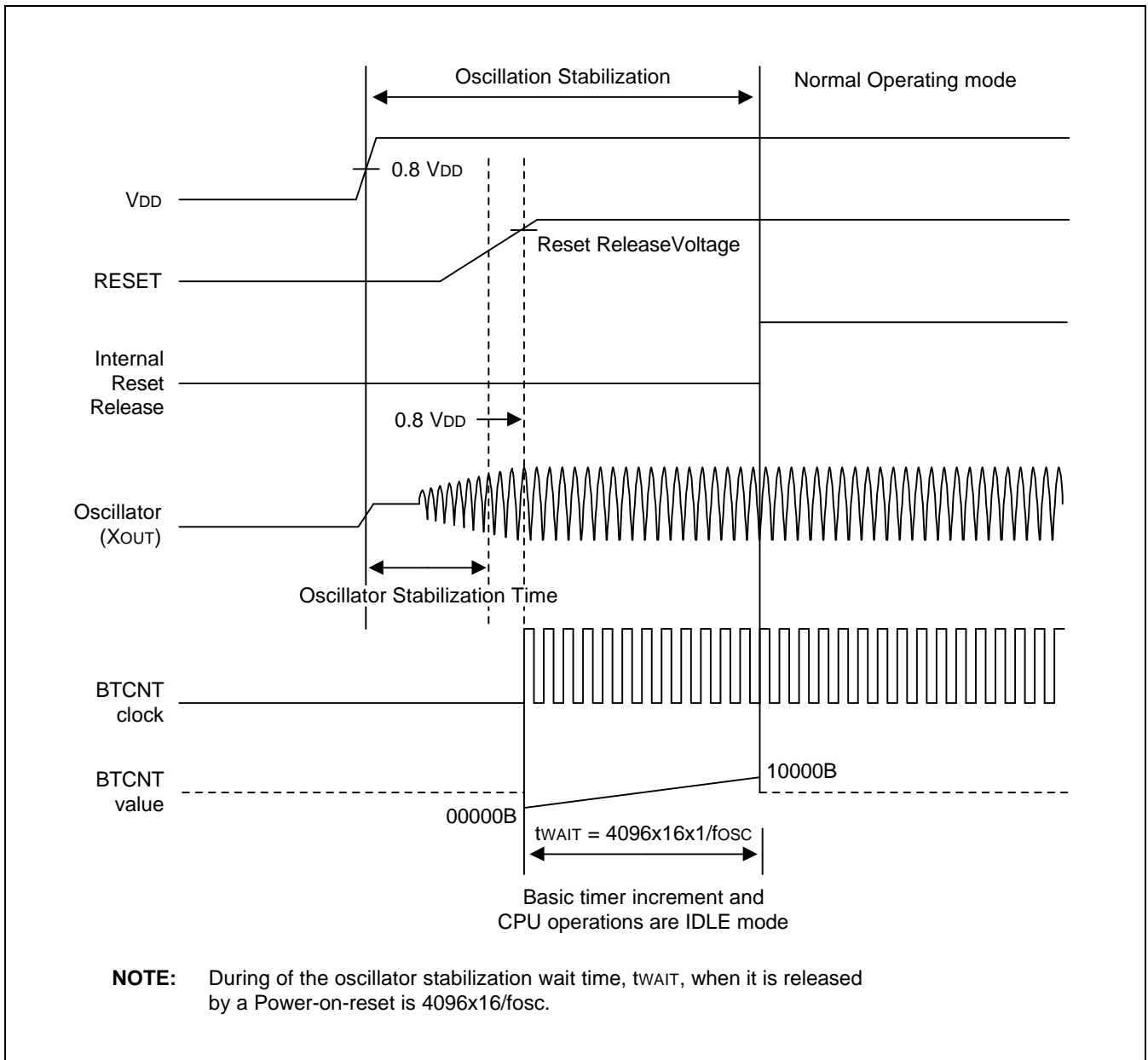


Figure 10-2. Oscillation Stabilization Time on RESET

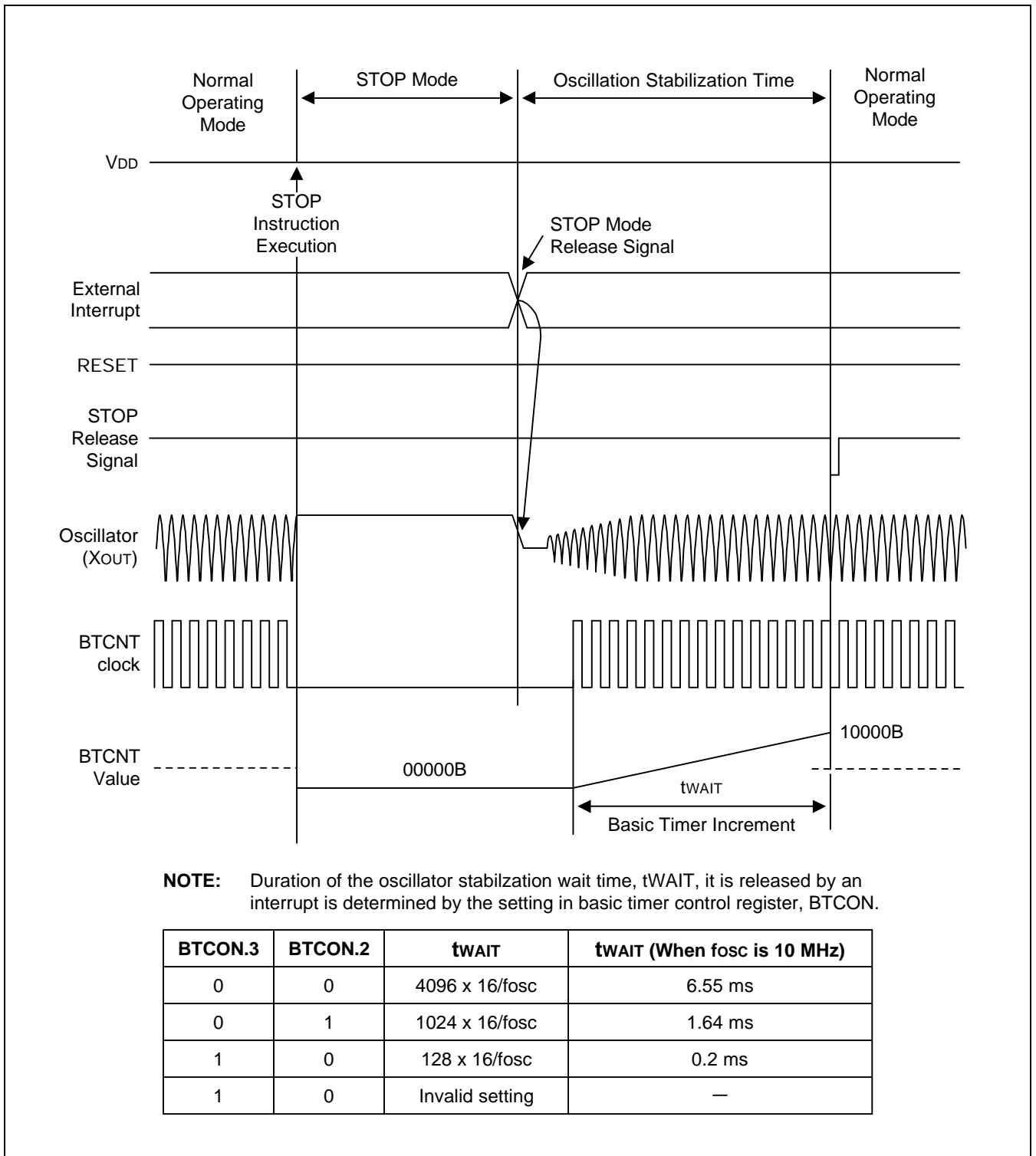


Figure 10-3. Oscillation Stabilization Time on STOP Mode Release

TIMER 0 CONTROL REGISTER (T0CON) AND INTERRUPT CONTROL REGISTER(T0INT)

You use the timer 0 control register (T0CON) and the timer 0 interrupt control register (T0INT), to

- Select the timer 0 operating mode (interval timer, capture mode, or PWM mode)
- Select the timer 0 input clock frequency
- Clear the timer 0 counter, T0CNT
- Enable the timer 0 overflow interrupt and timer 0 match/capture interrupts
- Clear timer 0 match/capture interrupt pending conditions

T0CON is located at address D2H and T0INT at address D3H, both are read/write addressable using Register addressing mode.

A reset clears T0CON and T0INT to "00H". This sets timer 0 to normal interval timer mode, selects an input clock frequency of $f_{OSC}/4096$, and disables the timer 0 overflow interrupt and match/capture interrupts. You can clear the timer 0 counter at any time during the normal operation by writing a "1" to T0CON.3.

When a timer 0 overflow interrupt occurs and is serviced by the CPU, the pending condition is to be cleared by Software. To enable the timer 0 match/capture interrupt, you must write T0INT.2 to "1". To detect an interrupt pending condition, the application program polls T0INT.0. When a "1" is detected, a timer 0 match/capture interrupt is pending. When the interrupt request has been serviced, the pending condition must be cleared by software by writing a "0" to the timer 0 interrupt pending bit, T0INT.0.

For a timer 0 overflow interrupt, like the way of handing timer 0 match/capture interrupt, you have to write T0INT.3 to "1" to be enabled. timer 0 overflow pending condition can be detected on the condition of T0INT.1 being set to "1" when the application program polls T0INT.1 and the pending condition must be cleared by software by writing T0INT.1 a "0" when the interrupt request has been serviced.

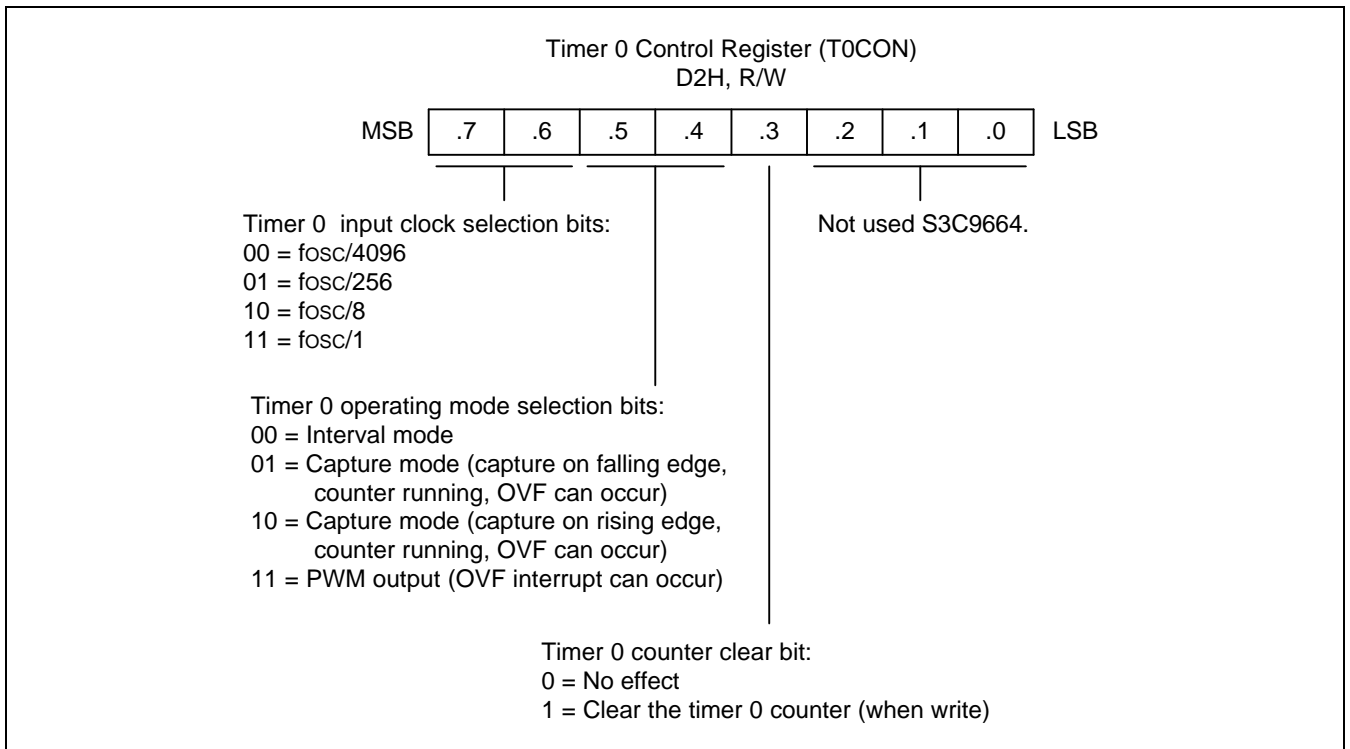


Figure 10-4. Timer 0 Control Register (T0CON)

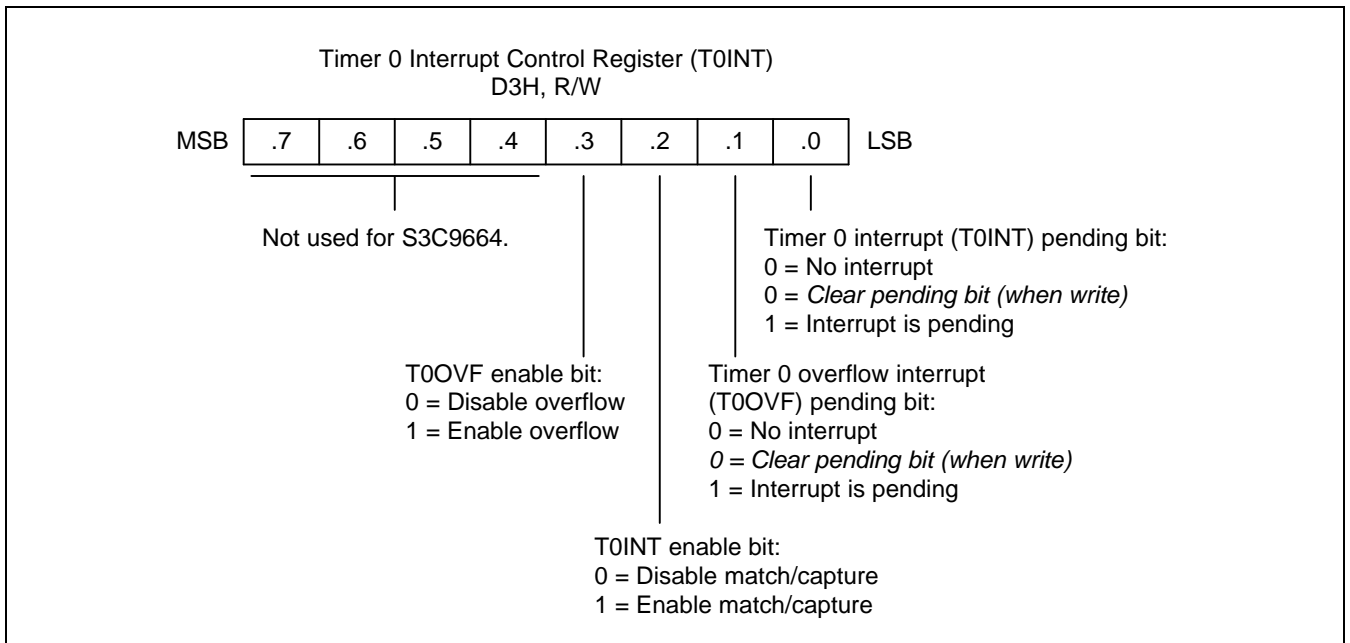


Figure 10-5. Timer 0 Interrupt Control Register (T0INT)

TIMER 0 FUNCTION DESCRIPTION

Timer 0 Interrupts

The timer 0 module can generate two interrupts: the timer 0 overflow interrupt, and the timer 0 match/capture interrupt. A timer 0 overflow interrupt pending condition and a timer 0 match /capture interrupt pending condition must be cleared by software, however, by writing a "0" to the T0INT.1 and T0INT.0 pending bits.

Interval Timer Mode

In interval timer mode, a match signal generates a timer 0 match interrupt and clears the counter. If you write the value "FFH" to the timer 0 reference data register, T0DATA, the counter will increment until an overflow occurs. (This condition is the same as normal counter operation.)

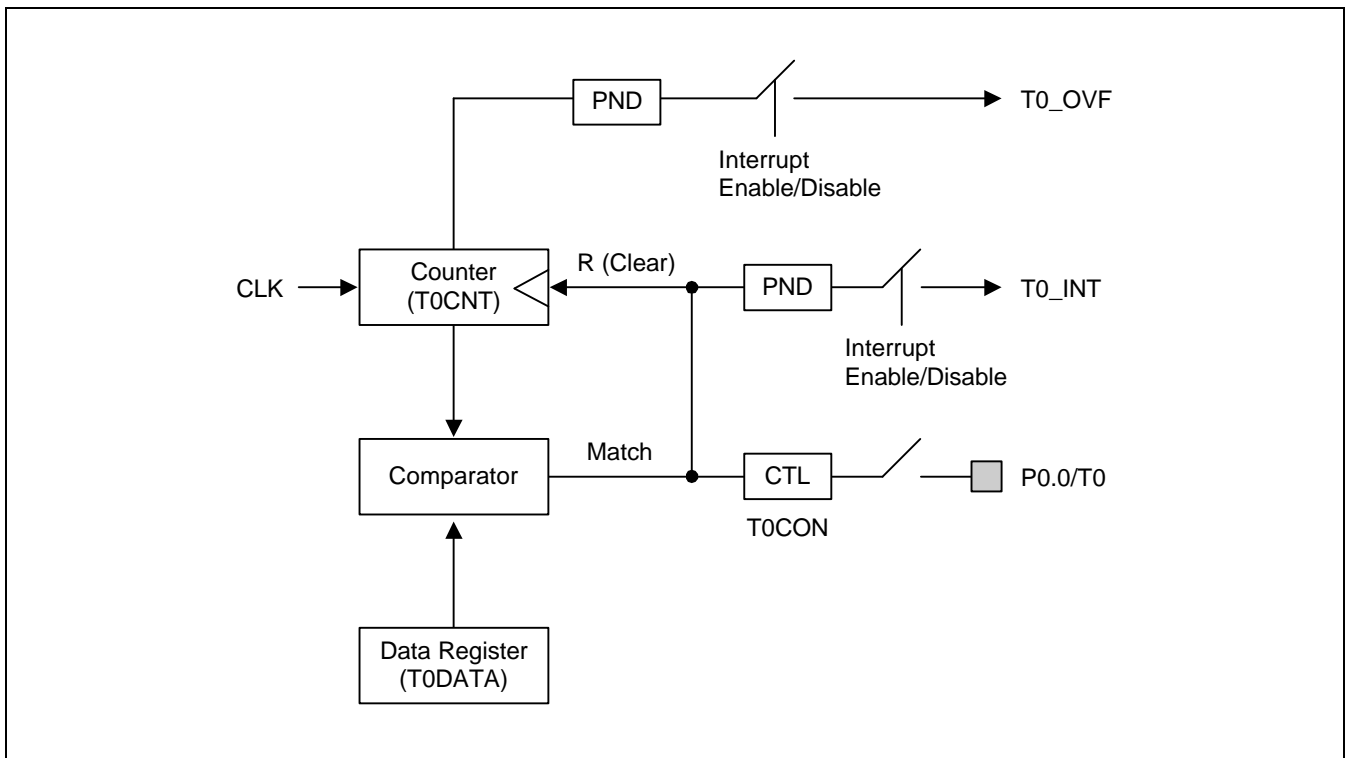


Figure 10-6. Simplified Timer 0 Function Diagram: Interval Timer Mode

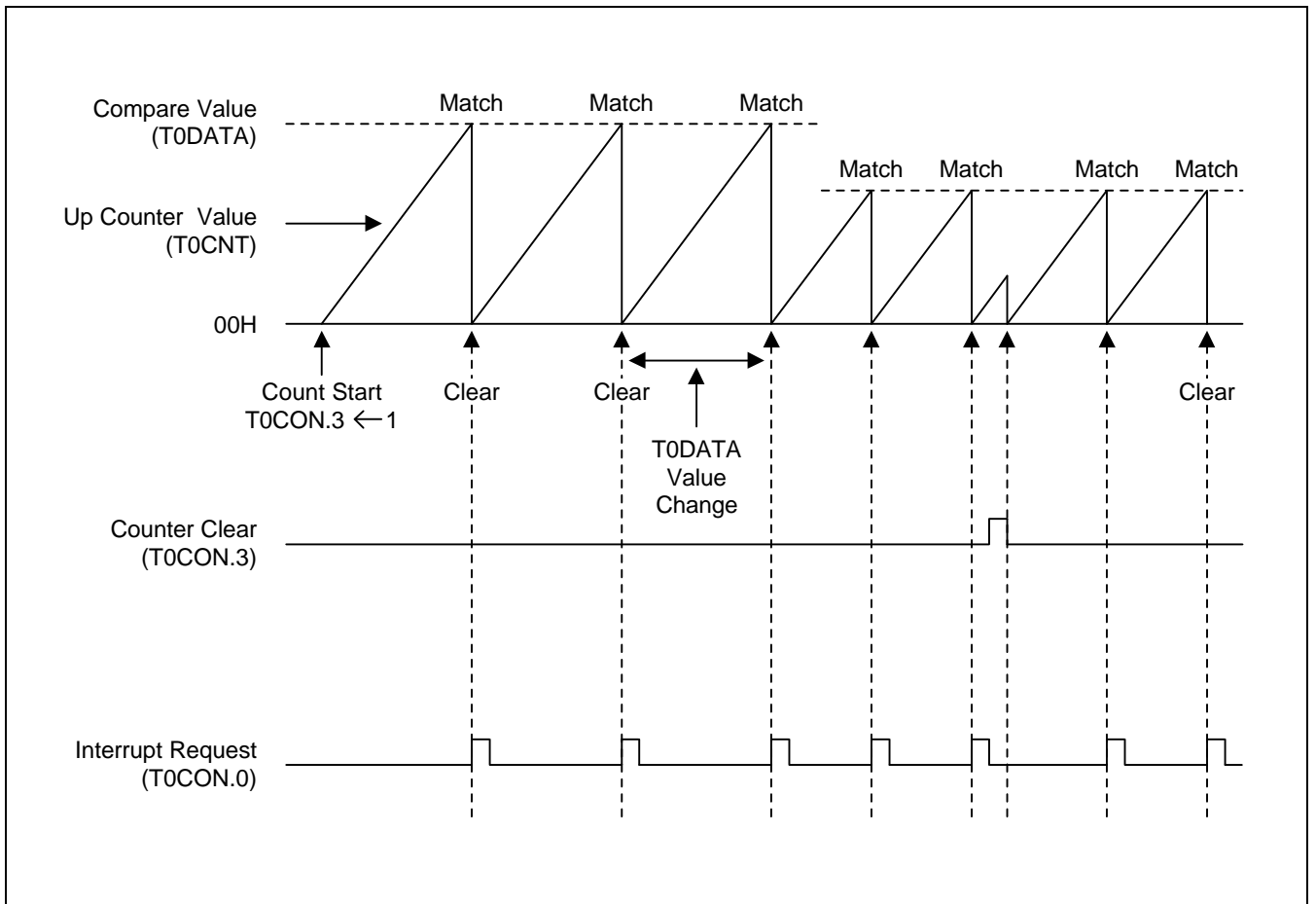


Figure 10-7. Timer 0 Timing Diagram

Pulse Width Modulation Mode

The PWM cycle width (time) is determined the timer 0 input clock. One cycle is equal to $t_{CLK} \times 2^8$ (for the 8-bit counter). The timer 0 data register value determines the pulse modulation width. (The minimum value is Low level and the maximum value is High level.) A match signal generates the timer 0 interrupt (T0_INT), but it does not clear the timer 0 counter value.

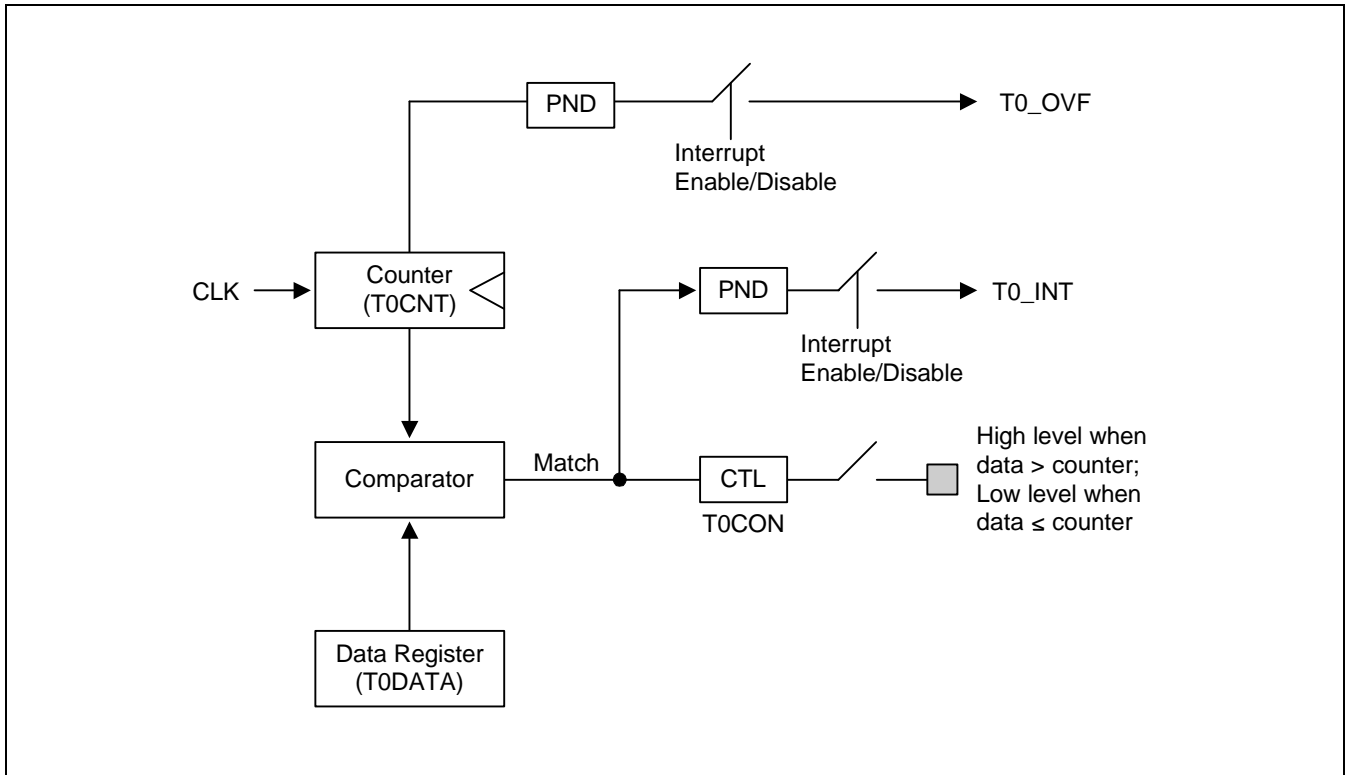


Figure 10-8. Simplified Timer 0 Function Diagram: PWM Mode

PWM FUNCTION DESCRIPTION

The 8-bit counter counts modulus 256, that is, from 0–255, inclusive. The value of the 8-bit counter is compared to the contents of the reference registers, TODATA. When the reference register value equals the counter value, the PWM output goes low. When the counter reaches zero, the PWM output is forced high. The low-to-high ratio (duty) of the PWM output is TODATA/256.

All PWM outputs remain inactive during the first 256 input clock signals. Then, when the counter value changes from FFH back to 00H, the PWM outputs are forced to high level. The pulse width ratio (duty cycle) is defined by the contents of the reference register and is programmed in increments of 1:256. The 8-bit PWM data register TODATA is read and written using 8-bit register addressing mode.

PWM output can be held at low level by continuously loading the reference register with 00H. By continuously loading the reference register with FFH, you can hold the PWM output to high level, except for the last pulse of the clock source, which sends the output low (see Figure 10-8).

Table 10-1. PWM Reference Register Duty Values

Reference Register Value (TODATA)	Duty
0000 0000	0/256 (0 %)
0000 0001	1/256 (0.39 %)
0000 0010	2/256 (0.78%)
•	•
•	•
1000 0000	128/256 (50 %)
1000 0001	129/256 (50.4 %)
•	•
•	•
1111 1110	254/256 (99.2 %)
1111 1111	255/256 (99.6 %)

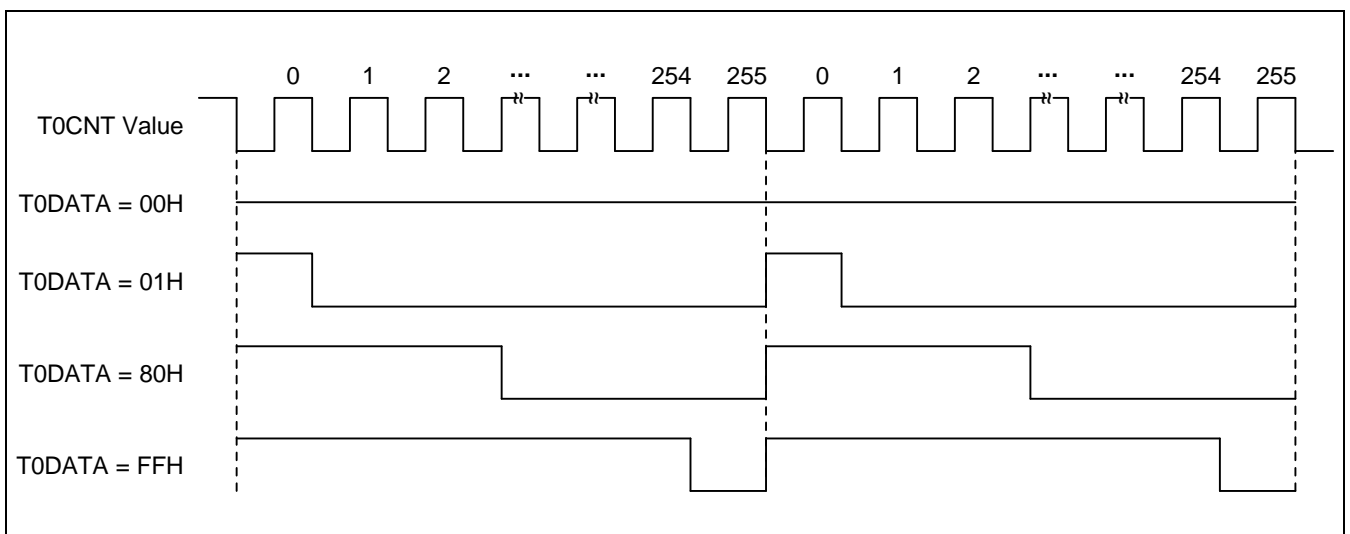


Figure 10-9. PWM Output Waveforms

Capture Mode

In capture mode, the timer 0 counter increases at a rate determined by the timer clock. The trigger signal (a rising or falling edge) for the capture operation occurs at the T0 pin. The interrupt service routine stores the captured 8-bit timer 0 counter value in the timer 0 data register whenever the capture signal is detected at the T0 pin.

By reading the counter value at programmed intervals, the routine can compare the differences between successive read values and calculate elapsed time interval. For example, if 80H is read and then 90H is read, this gives a difference of 10H. Depending on the clock speed, you can convert this hexadecimal value into the equivalent time interval (in this case, it is approximately 10 milliseconds).

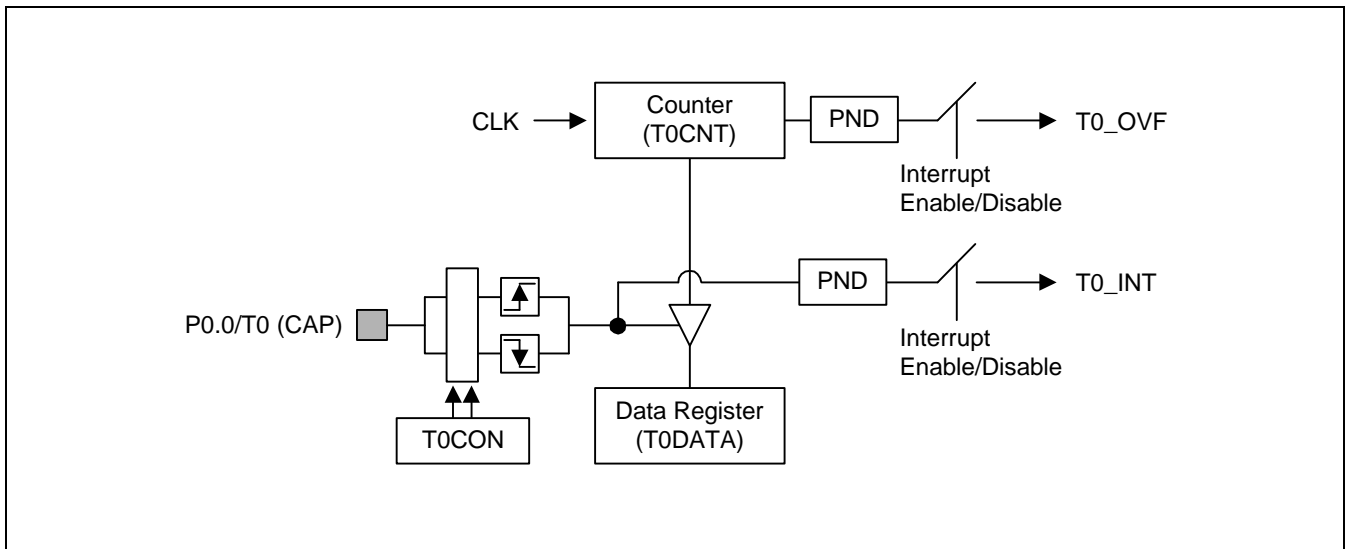


Figure 10-10. Simplified Timer 0 Function Diagram: Capture Mode

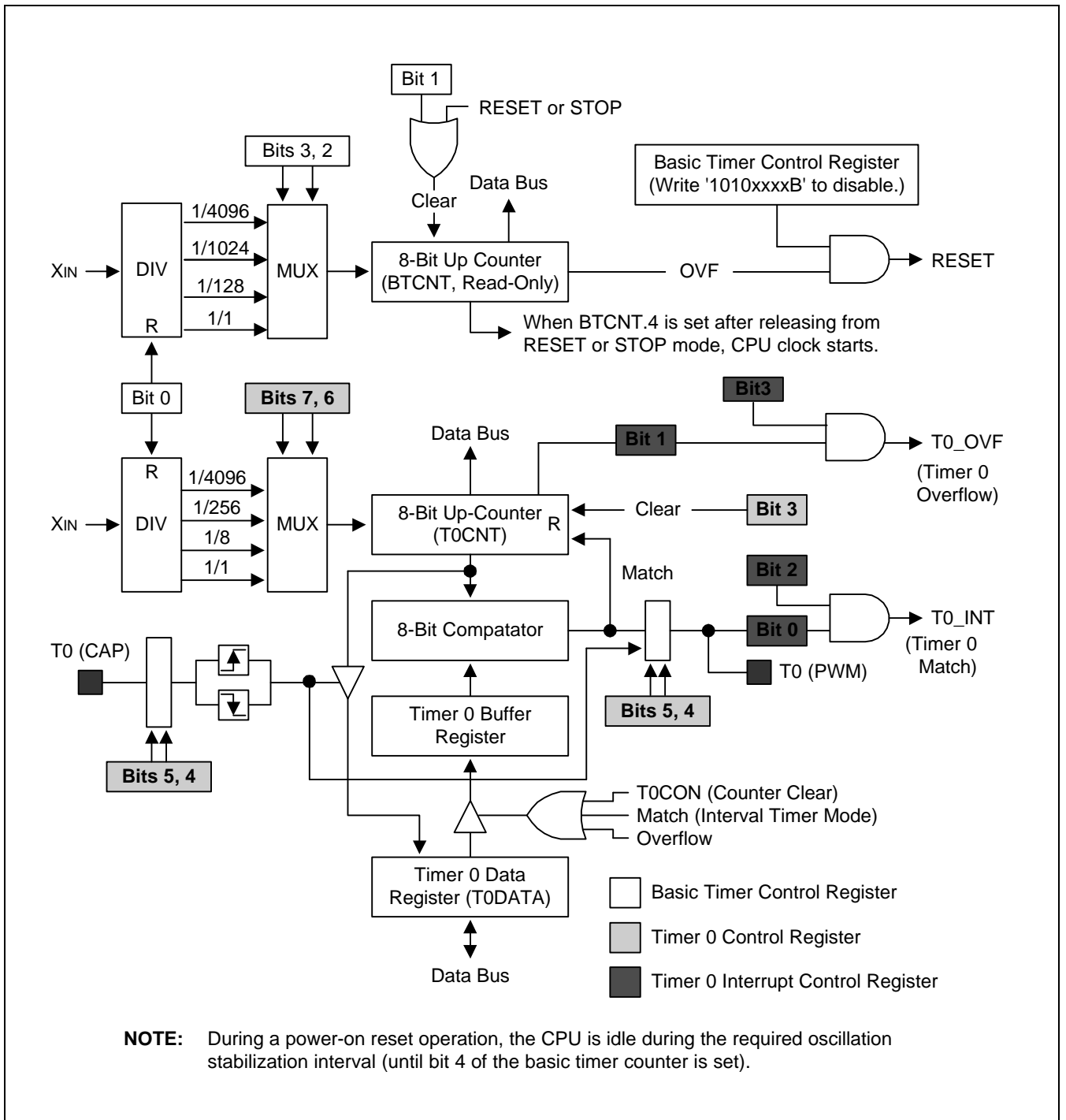


Figure 10-11. Basic Timer and Timer 0 Block Diagram

11

TIMER 1

OVERVIEW

The 8-bit timer 1 is an 8-bit general-purpose timer. Timer 1 has the interval timer mode by using the appropriate T1CON setting.

Timer 1 has the following functional components:

- Clock frequency divider (f_{OSC} divided by 1024, 256, 64 or 8) with multiplexer
- 8-bit counter (T1CNT at address E2H, Page 0), 8-bit comparator, and 8-bit reference data register (T1DATA at address EEH, Page 0)
- Timer 1 match interrupt generation
- Timer 1 control register, T1CON (at address E3H read/write, Page 0)

FUNCTION DESCRIPTION

Interval Timer Function

The timer 1 module can generate an interrupt: the timer 1 match interrupt (T1_INT). The application's service routine can detect a pending condition of T1_INT by the software and execute its sub-routine. When this case is used, the T1_INT pending bit must be cleared by the application sub-routine by writing a "0" to the T1CON.0 pending bit.

In interval timer mode, a match signal is generated when the counter value is identical to the value written to the T1 reference data register, T1DATA. The match signal generates a timer 1 match interrupt (TA_INT) and clears the counter.

If, for example, you write the value 10H to T1DATA and 0FH to T1CON, the counter will increment until it reaches 10H. At this point, the Timer 1 interrupt request is generated, the counter value is reset, and counting resumes.

Timer 1 Control Register (T1CON)

You use the timer 1 control register, T1CON, to

- Enable the timer 1 operating (interval timer)
- Select the timer 1 input clock frequency
- Clear the timer 1 counter, T1CNT
- Enable the timer 1 interrupt
- Clear timer 1 interrupt pending conditions

T1CON is located at address E3H, Page 0, and is read/write addressable using Register addressing mode.

A reset clears T1CON to '00H'. This sets timer 1 to disable interval timer mode, selects an input clock frequency of $f_{OSC}/1024$, and disables timer 1 interrupt. You can clear the timer 1 counter at any time during normal operation by writing a "1" to T1CON.3.

To enable the timer 1 interrupt, you must write T1CON.2 and T1CON.1 to "1". To generate the exact time interval, you should write T1CON.3 and .0, which cleared counter and interrupt pending bit. To detect an interrupt pending condition, the application program polls pending bit, T1CON.0. When a "1" is detected, a timer 0 interrupt is pending. When the T1_INT sub-routine has been serviced, the pending condition must be cleared by software by writing a "0" to the timer 1 interrupt pending bit, T1CON.0.

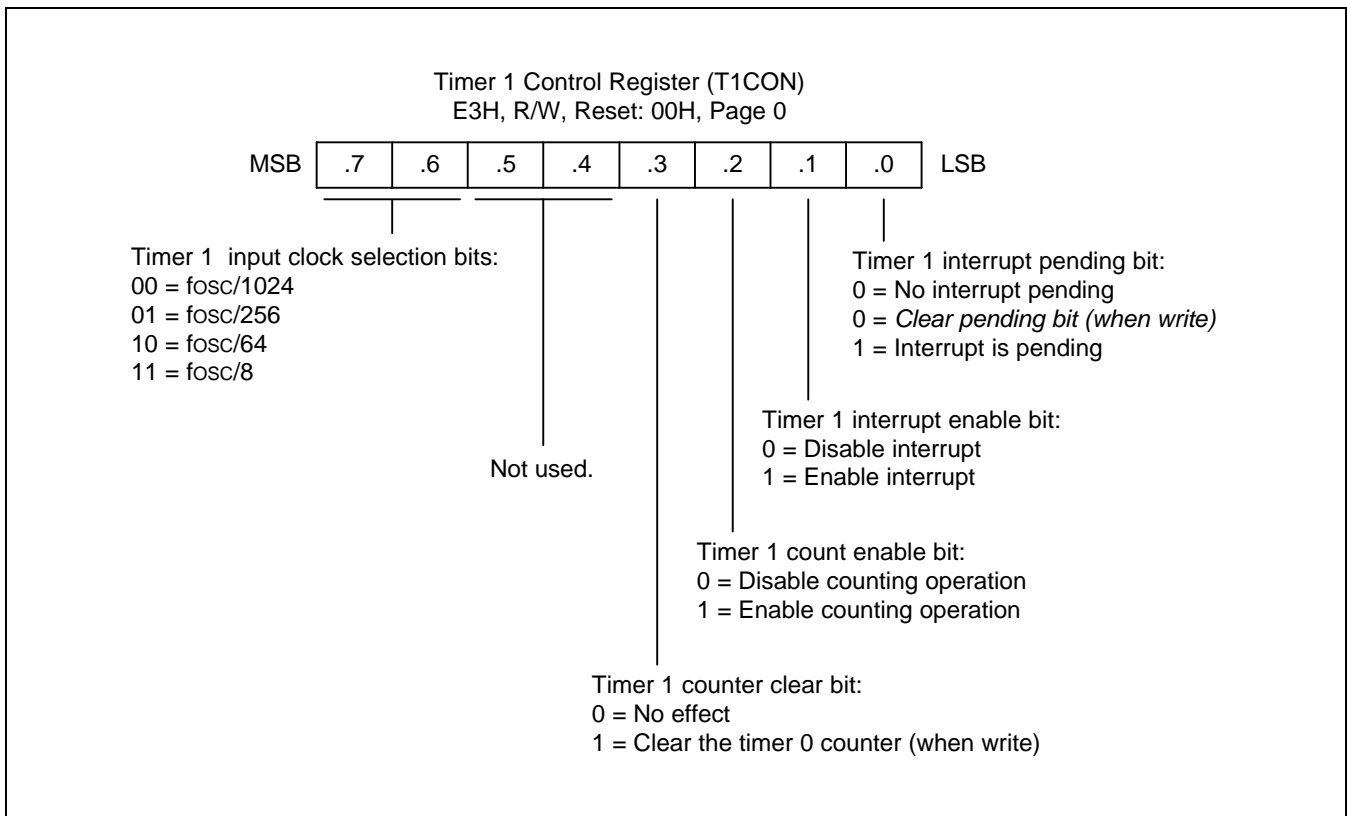


Figure 11-1. Timer 1 Control Register (T1CON)

BLOCK DIAGRAM

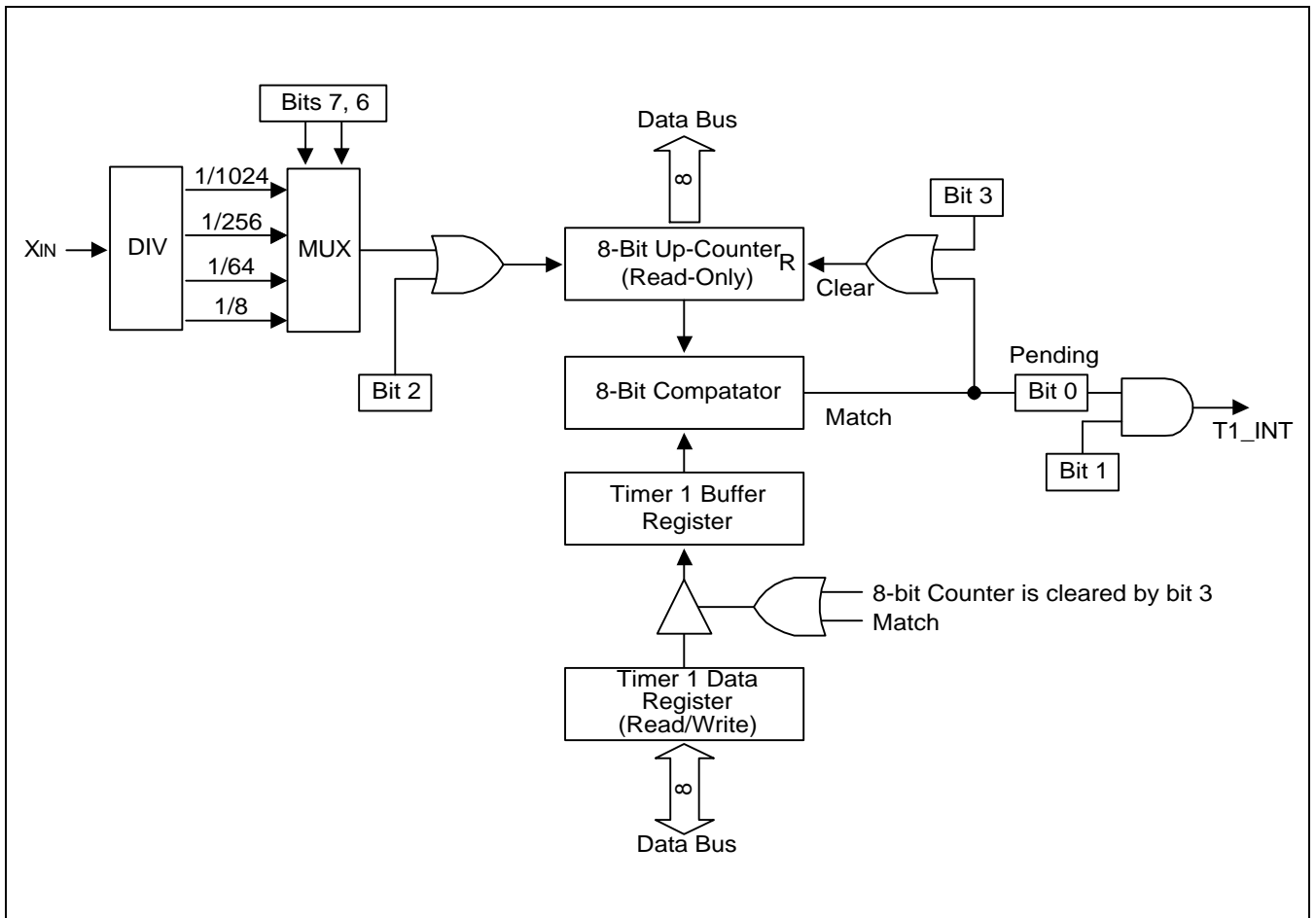


Figure 11-2. Timer 1 Functional Block Diagram

12 A/D CONVERTER

OVERVIEW

The A/D converter (ADC) module uses successive approximation logic to convert analog levels at one of the six input channels to equivalent 10-bit digital values. The analog input level must lie between the V_{DD} and V_{SS} values. The A/D converter has the following components:

- Six multiplexed analog input pins (AD0–AD5)
- Analog comparator with successive approximation logic
- 10-bit A/D conversion data output registers (ADDATAH, ADDATAL)
- ADC control register (ADCON)

An analog-to-digital conversion procedure is initiated when the CPU writes a value to the ADCON register at address (D8H, Page 0) to select one of the six available input pins. You select the desired input channel by setting the appropriate bits in the ADCON register.

The S3C9664/P9664 microcontroller performs 10-bit conversions for only one input channel at a time. You can dynamically select different analog input channels during program execution by manipulating selection bits in the ADCON register.

During a normal conversion, ADC logic initially sets the successive approximation register to 200H (the approximate half-way point of an 10-bit register). This register is then updated automatically during each conversion step. The successive approximation block performs 10-bit conversions for one input channel at a time. You can dynamically select different channels by manipulating the channel selection bit value (ADCON.6–4) in the ADCON register. To start the A/D conversion, you should set the enable bit, ADCON.0. When a conversion is completed, ADCON.3, the end-of-conversion (EOC) bit is automatically set to 1 and the result is dumped into the ADDATA register where it can be read. The A/D converter then enters an idle state. Remember to read the contents of ADDATA before another conversion starts. Otherwise, the previous result will be overwritten by the next conversion result.

NOTE

Because the ADC does not use sample-and-hold circuitry, it is important that any fluctuations in the analog level at the AD0–AD5 input pins during a conversion procedure be kept to an absolute minimum. Any change in the input level, perhaps due to circuit noise, will invalidate the result.

INTERNAL REFERENCE VOLTAGE LEVELS

In the ADC function block, the analog input voltage level is compared to the reference voltage. The analog input level must remain within the range AV_{SS} to V_{DD} .

Different reference voltage levels are generated internally along the resistor tree during the analog conversion process for each conversion step. The reference voltage level for the first bit conversion is always $1/2 V_{DD}$.

USING A/D PINS FOR STANDARD DIGITAL INPUT

The ADC module's input pins are alternatively used as digital input in port 1. The AD0–AD5 share pin names are P1.0–P1.5

A/D CONVERTER CONTROL REGISTER (ADCON)

The A/D converter control register, ADCON, is located at address D8H, Page 0. ADCON has four functions:

- Bits 6-4 select an analog input pin (AD0–AD5).
- Bit 3 indicates the status of the A/D conversion.
- Bit 2-1 select clock source.
- Bit 0 starts the A/D conversion.

Only one analog input channel can be selected at a time. You can dynamically select any one of the six analog input pins (AD0–AD5) by manipulating the 3-bit value for ADCON.6–ADCON.4

It's recommended that the clock source determining the conversion speed would be less than 2.5 MHz to get sound results.

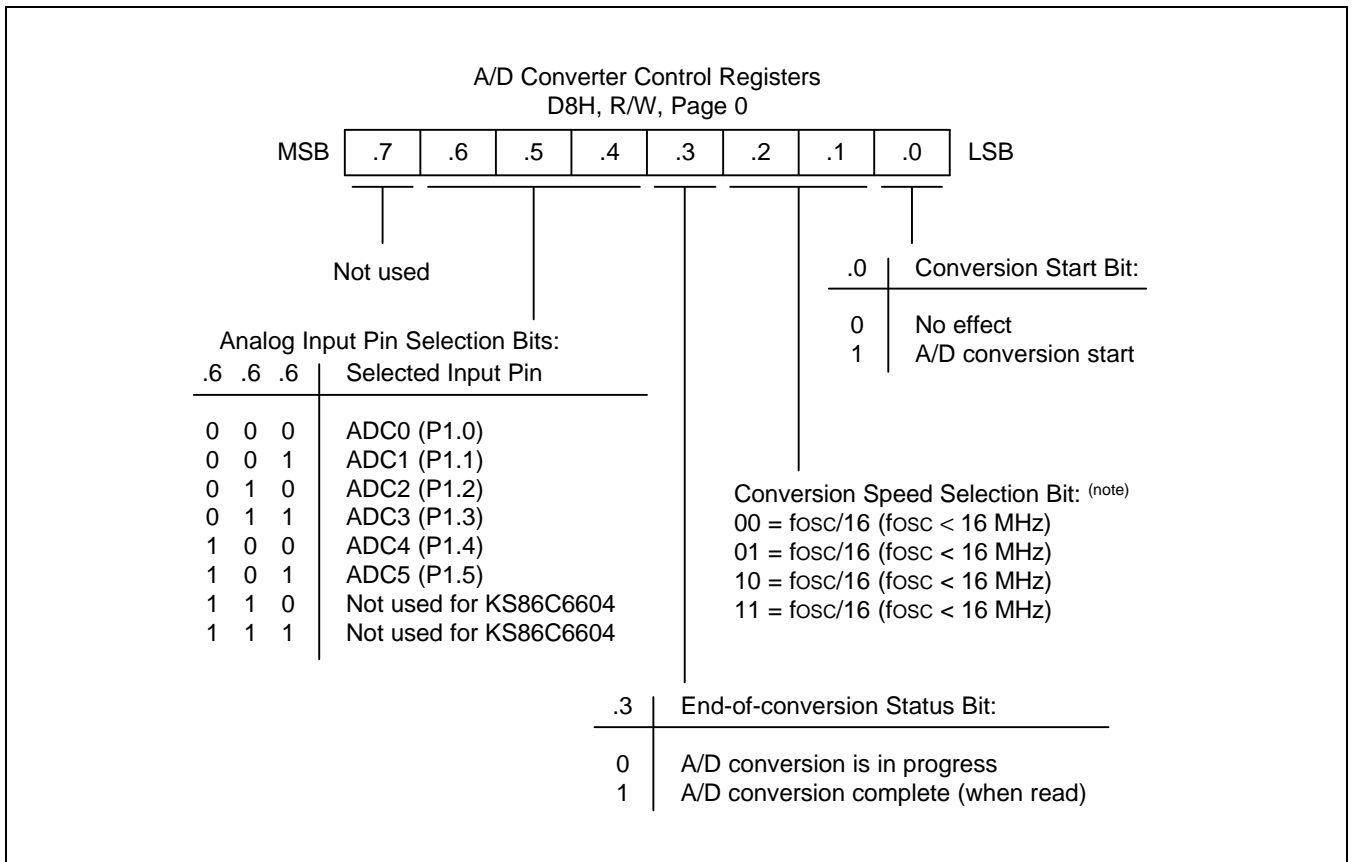


Figure 12-1. A/D Converter Control Register (ADCON)

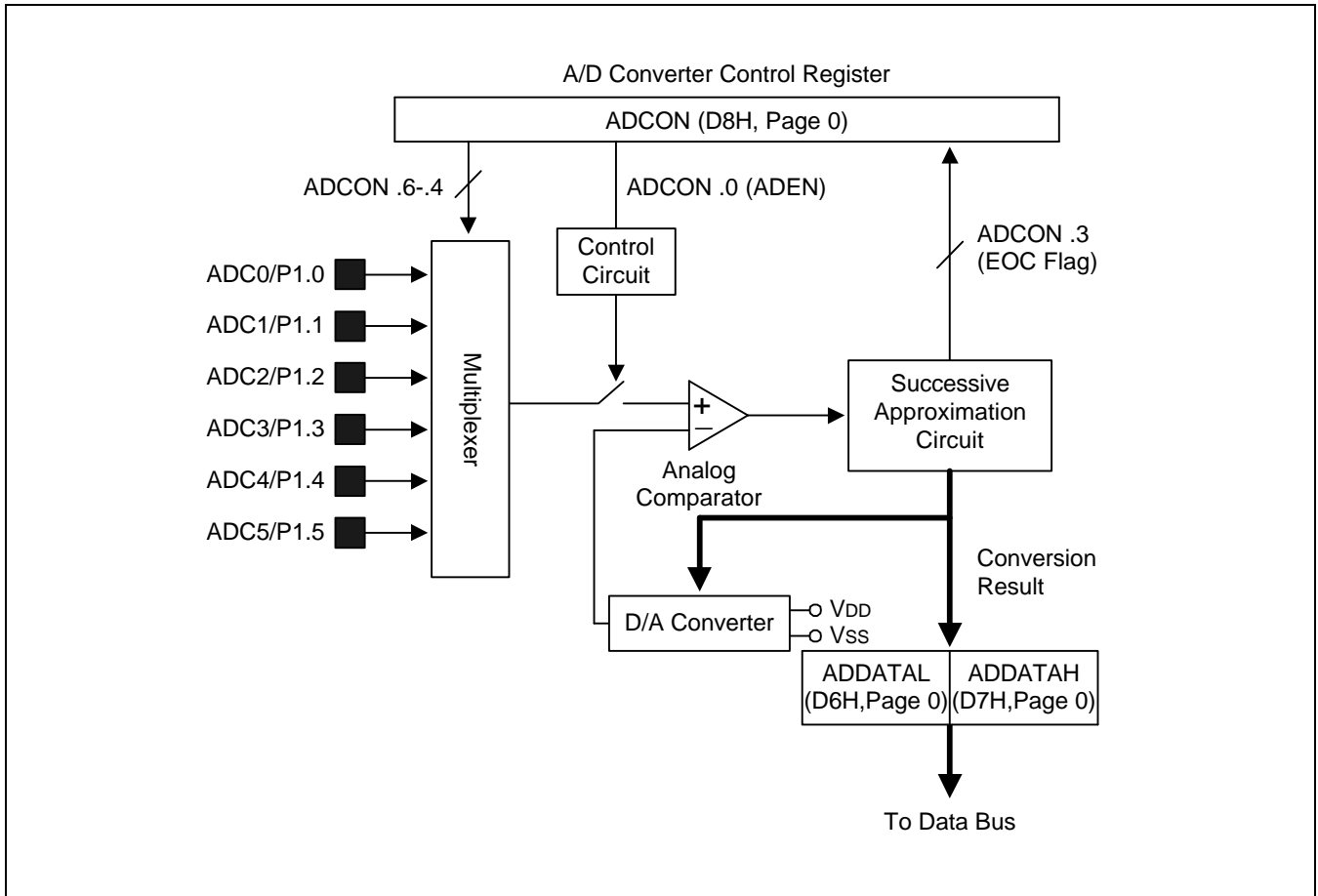


Figure 12-2. A/D Converter Circuit Diagram

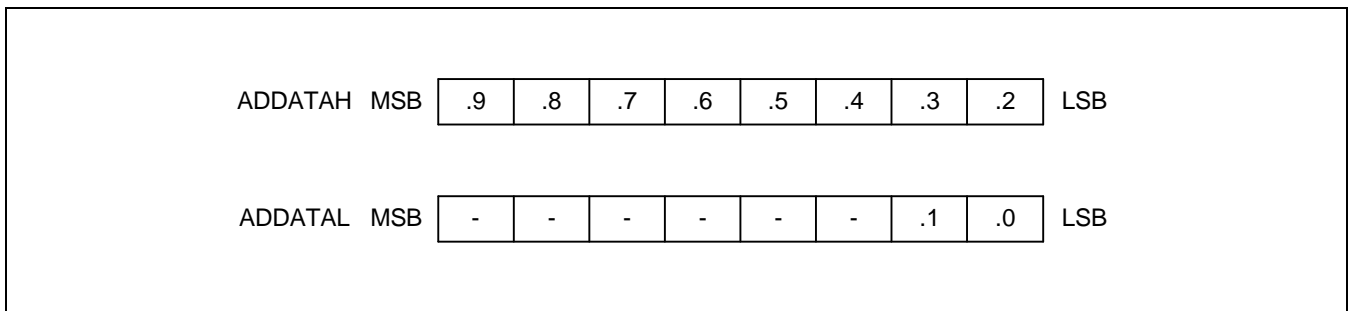


Figure 12-3. A/D Converter Data Register (ADDATA)

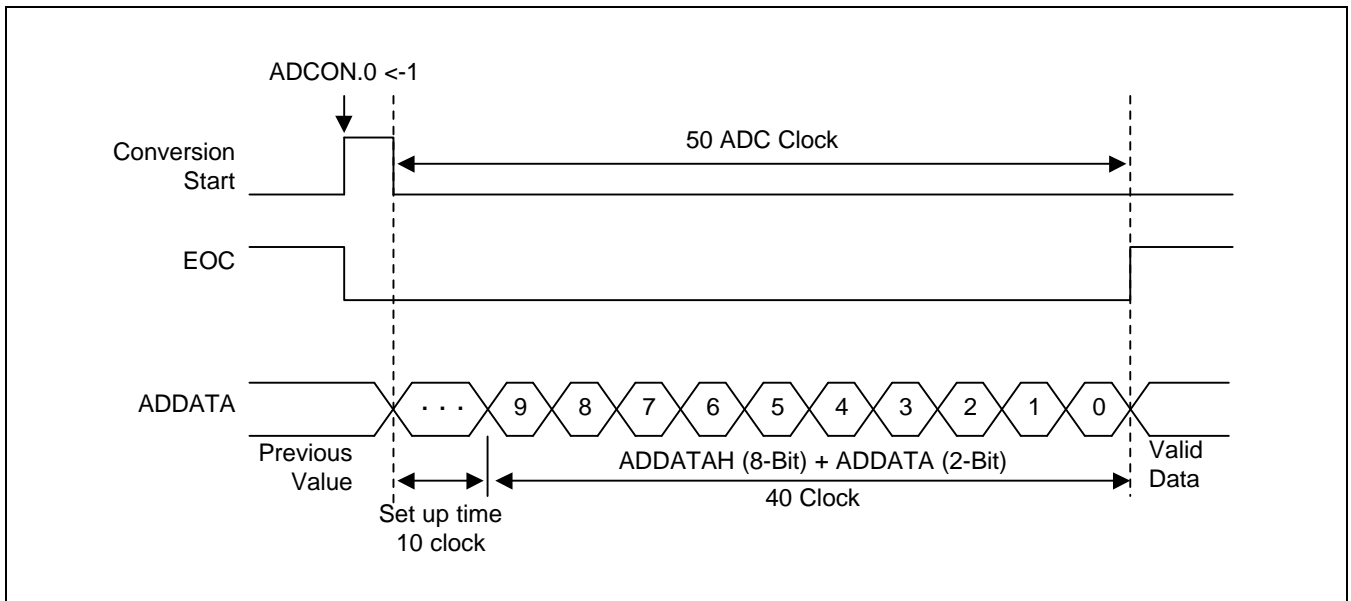


Figure 12-4. A/D Converter Timing Diagram

CONVERSION TIMING (S3C9664)

The A/D conversion process requires 4 steps (4 clock edges) to convert each bit and 10 clocks to step-up A/D conversion. Therefore, total of 50 clocks are required to complete an 10-bit conversion: With an 10 MHz CPU clock frequency, one clock cycle is 400 ns ($4/f_{OSC}$). If each bit conversion requires 4 clocks, the conversion rate is calculated as follows:

$$4 \text{ clocks/bit} \times 10\text{-bits} + \text{step-up time (10 clock)} = 50 \text{ clocks}$$

$$50 \text{ clock} \times 400 \text{ ns} = 20 \mu\text{s at } 10 \text{ MHz, } 1 \text{ clock time} = 4/f_{OSC} \text{ (assuming } ADCON.2\text{--}1 = 10)$$

INTERNAL A/D CONVERSION PROCEDURE

1. Analog input must remain between the voltage range of V_{SS} and AV_{DD} .
2. Configure the analog input pins to input mode by making the appropriate settings in P1CONH, P1CONL
3. Before the conversion operation starts, you must first select one of the eight input pins (AD0–AD5) by writing the appropriate value to the ADCON register.
4. When conversion has been completed, (50 CPU clocks have elapsed), the EOC flag is set to "1", so that a check can be made to verify that the conversion was successful.
5. The converted digital value is loaded to the output register, ADDATAH (High 8-bit) and ADDATAL (Low 2-bit), then the ADC module enters an idle state.

The digital conversion result can now be read from the ADDATAH and ADDATA L registers.

13 UNIVERSAL SERIAL BUS

OVERVIEW

Universal Serial Bus (USB) is a communication architecture that supports data transfer between a host computer and a wide range of PC peripherals. USB is actually a cable bus in which the peripherals share its bandwidth through a host scheduled token based protocol.

The USB module in S3C9664 is designed to serve at a low speed transfer rate (1.5 Mbps) USB device as described in the Universal Serial Bus Specification Revision 1.1. S3C9664 can be briefly described as a microcontroller with SAM 87RCRI core with an on-chip USB peripheral as can be seen in figure 13-1.

The S3C9664 comes equipped with Serial Interface Engine (SIE), which handles the communication protocol of the USB. The S3C9664 supports the following control logic: packet decoding/generation, CRC generation/checking, NRZI encoding/decoding, Sync detection, EOP (end of packet) detection and bit stuffing.

S3C9664 supports two types of data transfer; control and interrupt. Three endpoints are used in this device; Endpoint 0, Endpoint 1, Endpoint2. Please refer to the USB specification revision 1.1 for detail description of USB.

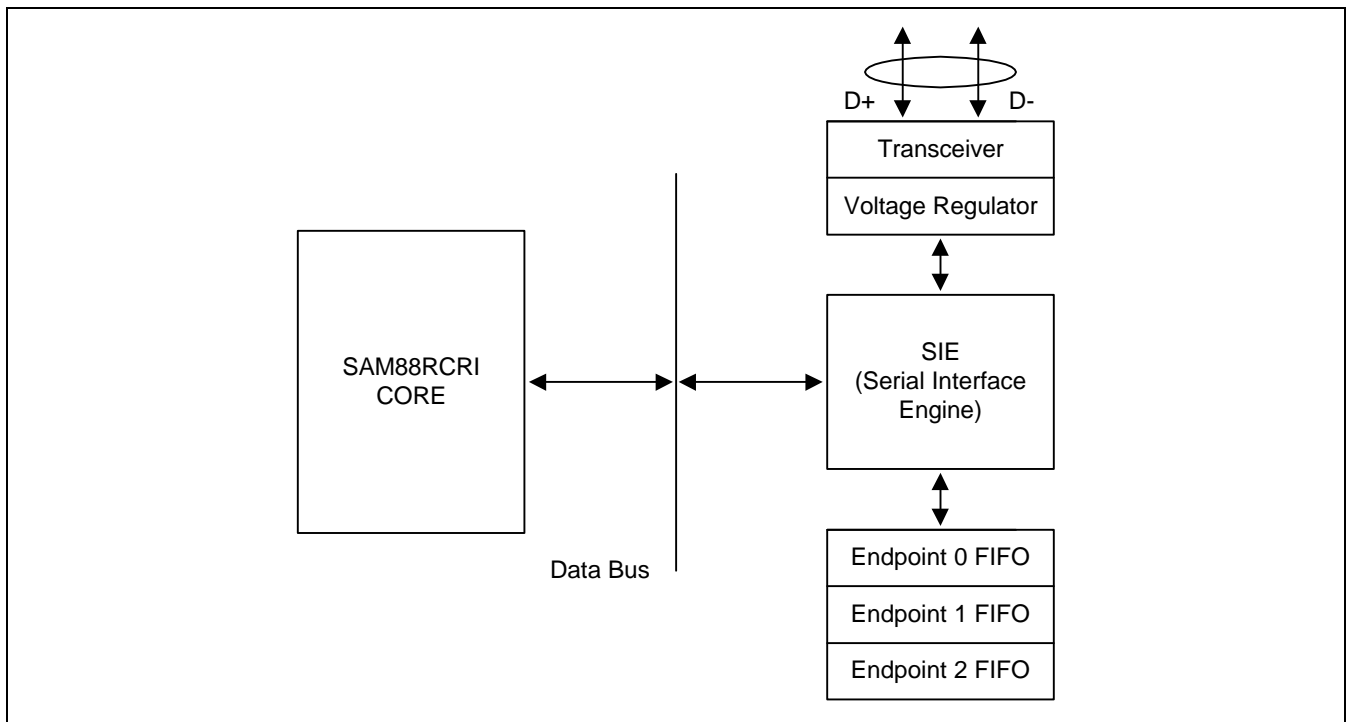


Figure 13-1. USB Peripheral Interface

Serial Bus Interface Engine (SIE)

The Serial Interface Engine interfaces to the USB serial data and handles, deserialization/serialization of data, NRZI encoding/decoding, clock extraction, CRC generation and checking, bit stuffing and other specifications pertaining to the USB protocol such as handling inter packet time out and PID decoding.

Control Logic

The USB control logic manages data movements between the CPU and the transceiver by manipulating the transceiver and the endpoint register. This includes both transmit and receive operations on the USB. The logic contains byte count buffers for transmit operations that load the active transmit endpoint's byte count and use this to determine the number of bytes to transfer. The same buffer is used for receive transactions to count the number of bytes received and transfer that number to the receiver endpoint's byte count register at the end of the transaction.

The control logic in S3C9664, when transmit, manages parallel to serial conversion, packet generation, CRC generation, NRZI encoding and bit stuffing.

When receive, the control logic in S3C9664 handles Sync detection, packet decoding, EOP (End Of Packet) detection, remove bit stuffing, NRZI decoding, CRC checking and serial to parallel conversion

Bus Protocol

All bus transactions involve the transmission of packets. S3C9664 supports low-speed packets. Each transaction starts when the host controller sends a Token Packet to the USB device. The Token packets are generated by the USB host and decoded by the USB device. A Token Packet includes the type description, direction of the transaction, USB device address and the endpoint number.

Data and Handshake packets are both decoded and generated by the USB device. In any transaction, the data is transferred from the host to a device or from a device to the host. The transaction source then sends a Data Packet or indicates that it has no data to transfer. The destination then responds with a Handshake Packet indicating whether the transfer was successful.

Data Transfer Types

USB data transfer occurs between the host software and a specific endpoint on the USB device. An endpoint supports a specific type of data transfer. The S3C9664 supports two types transfer endpoints: control and interrupt.

Control transfer configures and assigns an address to the device when detected. Control transfer also supports status transaction, returning status information from device to host.

Interrupt transfer refers to a small, spontaneous data transfer from USB device to host.

Endpoints

Communication flows between the host software and the endpoints on the USB device. Each endpoint on a device has an identifier number. In addition to the endpoint number, each endpoint supports a specific transfer type. S3C9664 supports three endpoints: Endpoint 0 supports control transfer, and Endpoint 1, Endpoint 2 supports interrupt transfer.

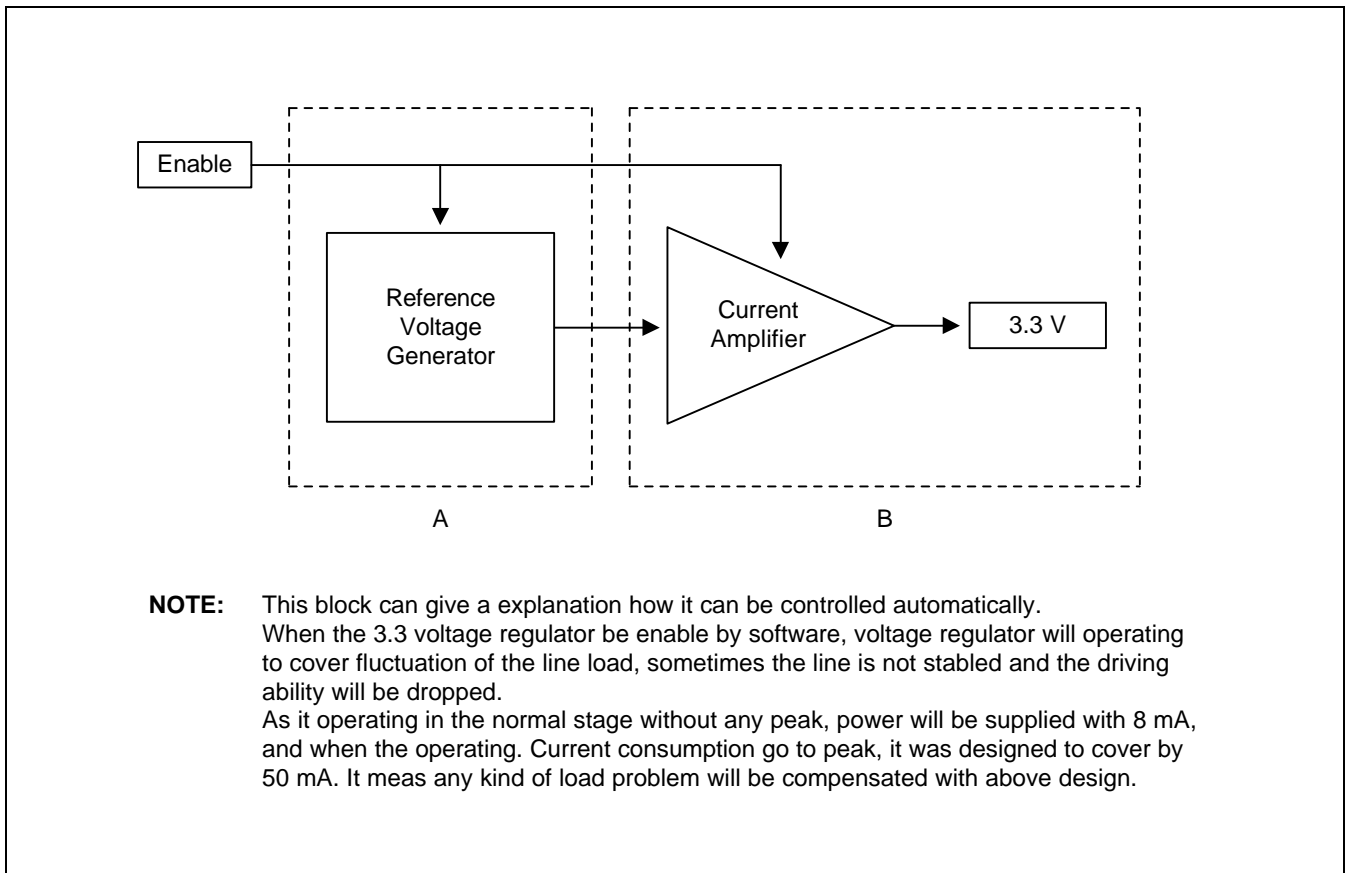


Figure 13-2. Block Diagram of Voltage Regulator

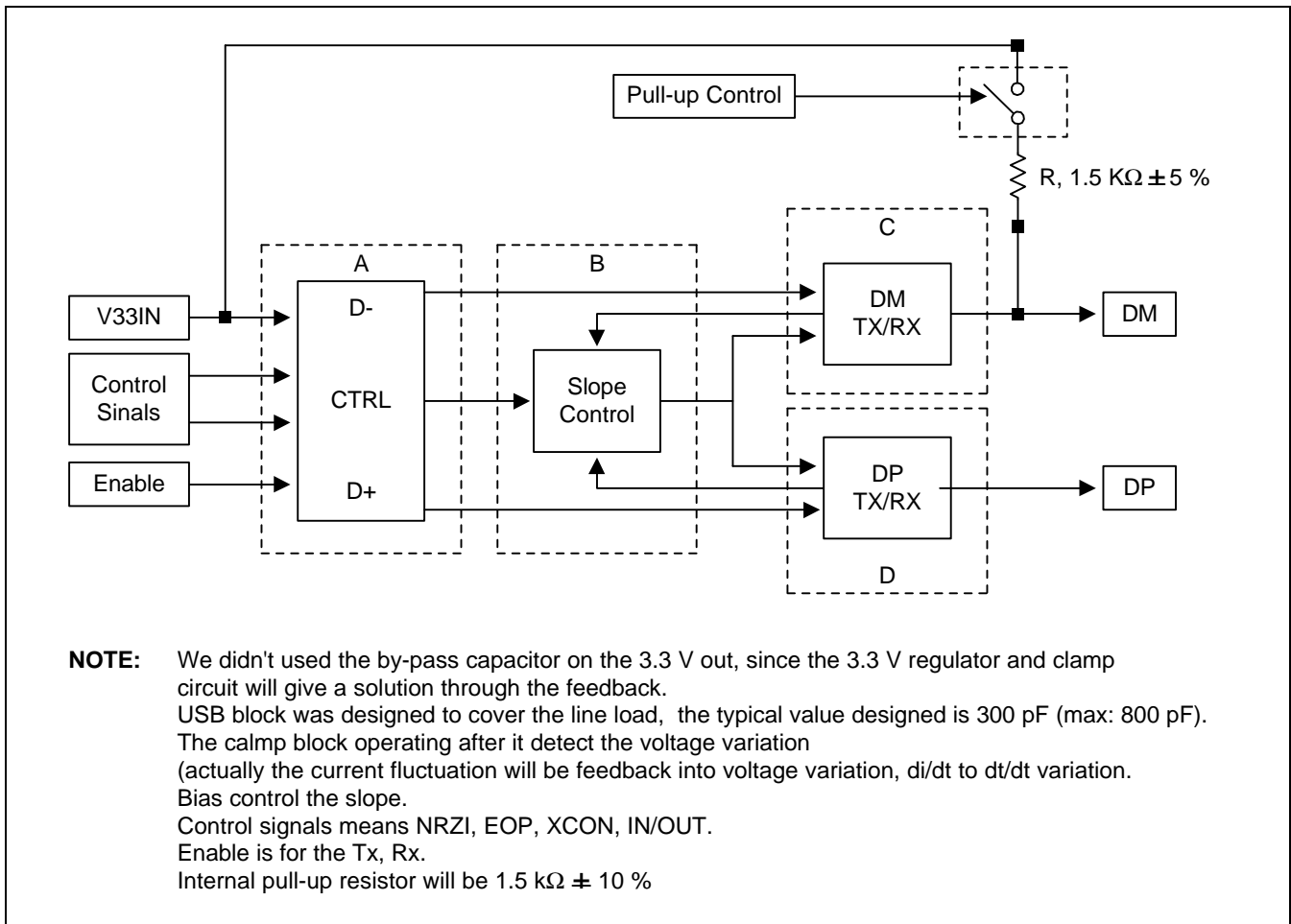


Figure 13-3. Block Diagram of USB Signal Transceiver

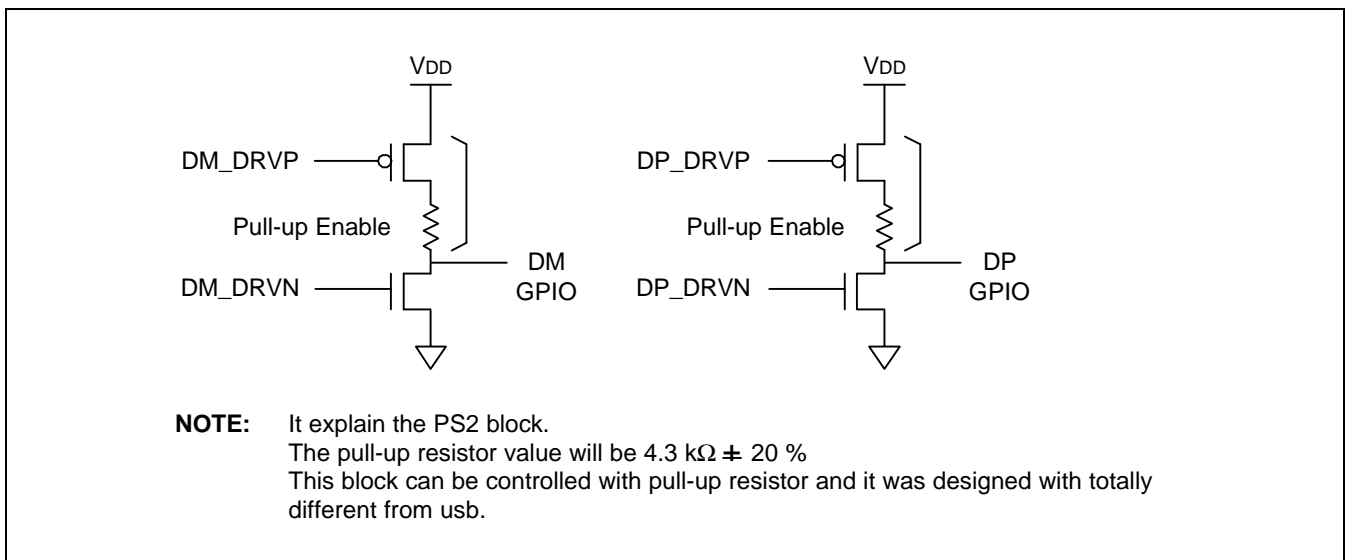


Figure 13-4. Block Diagram of GPIO Signal Transmitter

USB FUNCTION ADDRESS REGISTER (FADDR)

This register holds the USB address assigned by the host computer. FADDR is located at address F0H, Page 0 and is read/write addressable.

Bit7 Not used (Read value will be always "0").

Bit6–0 **FADDR**: MCU updates this register when it decodes a SET_ADDRESS command. MCU must write this register with clears OUT_PKT_RDY (bit0) and sets DATA_END (bit3) in the EP0CSR register. That is, MCU should write #48h to EP0CSR register after write this register. The function controller use this register's value to decode USB Token packet address. Reset value of this register is "0". The new address will work after successful status stage.

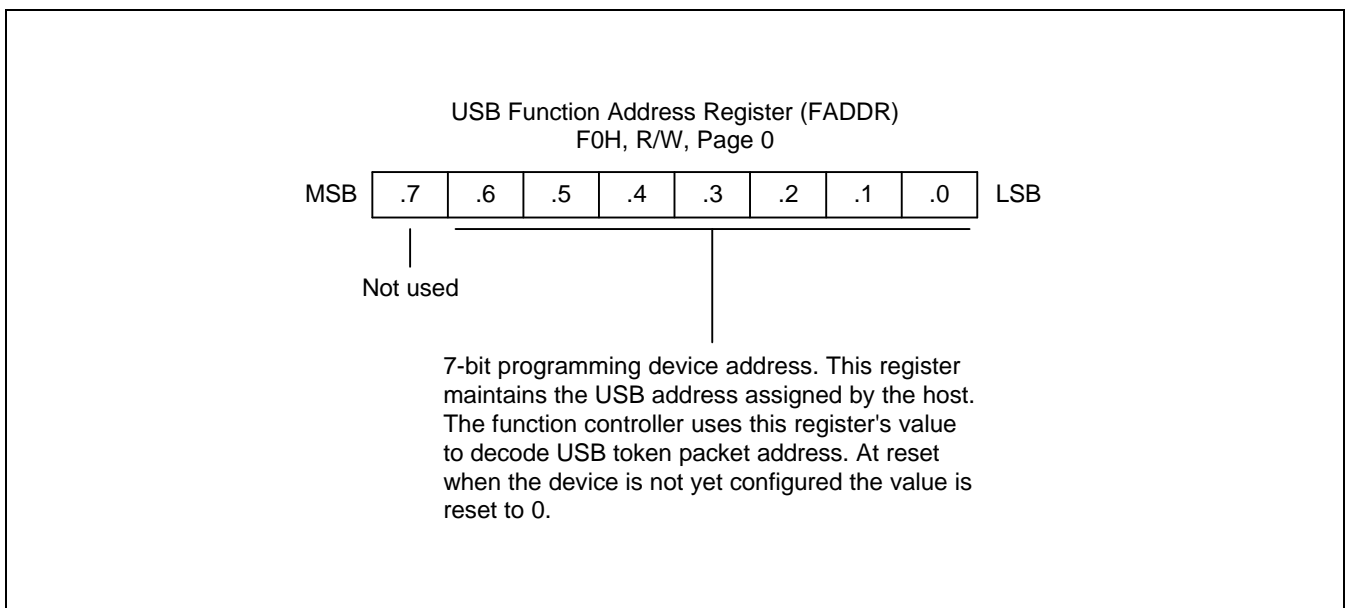


Figure 13-5. USB Function Address Register (FADDR)

CONTROL ENDPOINT STATUS REGISTER (EPOCSR)

EPOCSR register controls Endpoint 0 (Control Endpoint), and also holds status bits for Endpoint 0. EPOCSR is located at F1H and is read/write addressable.

- Bit7 **CLEAR_SETUP_END** : MCU writes "1" to this bit to clear SETUP_END bit (bit4). This bit is automatically cleared after clearing SETUP_END bit by SIE. So read value will be always "0".
- Bit6 **CLEAR_OUT_PKT_RDY**: MCU writes "1" to this bit to clear OUT_PKT_RDY bit (bit0). This bit is automatically cleared after clearing OUT_PKT_RDY bit by USB block. So read value will be always "0".
- Bit5 **SEND_STALL**: MCU writes "1" to this bit to send STALL packet to Host, it must clear OUT_PKT_RDY (bit 0) at the same time. If MCU receive invalid command then should write #60h to this register. The SIE issues a STALL handshake to the current control transfer(Means next transaction). This bit will be cleared after sending STALL handshake.
- Bit4 **SETUP_END** : SIE sets this bit, when a control transfer ends without setting DATA_END bit (bit3). MCU clears this bit, by writing a "1" to CLEAR_SETUP_END bit (bit7). When SIE sets this bit, an interrupt is generated to MCU. When such condition occurs, SIE flushes the FIFO. MCU can not access to FIFO until this bit cleared. This flag is a read only bit so MCU can not write to this bit directly.
- Bit3 **DATA_END**: MCU sets this bit:
- After loading the last packet of data into the FIFO, and at the same time IN_PKT_RDY bit should be set.
 - While it clears OUT_PKT_RDY bit after unloading the last packet of data.
 - For a zero length data phase, this bit should be set when it clears OUT_PKT_RDY bit.
- Bit2 **SENT_STALL**: SIE sets this bit after send stall handshake to host. There are two cases which issue stall packet to host. If MCU set SEND_STALL bit, then SIE will send stall to the next transaction and set this bit. The other case is send stall by SIE automatically since protocol violation. An interrupt is generated when this bit gets set. This bit is a read/write bit so MCU should clears this bit to end the STALL condition.
- Bit1 **IN_PKT_RDY**: MCU sets this bit, after loading data into Endpoint 0 FIFO. SIE clears this bit, once the packet has been successfully sent to the host. An interrupt is generated when SIE clears this bit so that MCU can load the next packet. For a zero length data phase, MCU sets IN_PKT_RDY bit without load data to FIFO.
- Bit0 **OUT_PKT_RDY**: SIE sets this bit, if the device receive valid data from host. An interrupt is generated, when SIE sets this bit. MCU should download data and clears this bit by writing "1" to CLEAR_OUT_PKT_RDY bit at the end of execution.

NOTES:

1. When SETUP_END bit is set, OUT_PKT_RDY bit may also be set. This happens when the current transfer has terminated by new setup transaction. In such case, MCU should first clear SETUP_END bit, and then start servicing the new control transfer.

INTRRUPT ENDPOINTS STATUS REGISTER (EP1CSR, EP2CSR)

EP1CSR register controls Endpoint 1, and also holds status bits for Endpoint 1. EP1CSR is located at F2H and is read/write addressable. EP2CSR register controls Endpoint2, and the contents is perfectly same to EP1CSR. EP2CSR is located at F9H and is read/write addressable.

EP1CSR and EP2CSR have two modes. These are IN and OUT mode which are decided by ENDPOINT_MODE register. The below is IN mode configuration.

- Bit7 **CLR_DATA_TOGGLE** : MCU write "1" to this bit for initializing data toggle sequence. Data toggle sequence can be monitored through WRT_CNT register.
- Bit6 MAXP[3].
- Bit5 MAXP[2].
- Bit4 MAXP[1].
- Bit3 MAXP[0].
- KS86P6604 is a low speed USB controller so the maximum packet size is 8 bytes,
 - This part is a limitation of MAXMUM packet size so the device can not send more data than this value.
- Bit2 **UC_FIFO_FLUSH** : MCU sets this bit for initializing the FIFO. MCU can not clear IN_PKT_RDY so if MCU want to clear IN_PKT_RDY after set then MCU should issue UC_FIFO_FLUSH for clearing IN_PKT_RDY.
- Bit1 **FORCE_STALL** : MCU sets this bit for sending stall packet. This flag will not be cleared by SIE. So MCU should clear this flag for stopping stall condition. Device will send stall until this flag is cleared.
- Bit0 **IN_PKT_RDY** : MCU sets this bit after loading data to FIFO. SIE will clear this flag after sending data to host. An interrupt is generated when this flag is cleared. If MCU issue UC_FIFO_FLUSH during this flag set then this flag is cleared and generate interrupt to MCU. So MCU will get interrupt directly after setting UC_FIFO_FLUSH flag if this flag was set.

INTRRUPT ENDPOINTS STATUS REGISTER (EP1CSR, EP2CSR)

The below is OUT mode configuration.

Bit7 **Reserved.**

Bit6 **CLEAR_OUT_PKT_RDY** : MCU writes "1" to this bit to clear OUT_PKT_RDY bit (bit0). This bit is automatically cleared after clearing OUT_PKT_RDY bit by SIE. So read value will be always "0".

Bit5 **Reserved.**

Bit4 **Reserved.**

Bit3 **SENT_STALL** :. This flag is set by SIE after sending stall packet. And this flag is just for monitoring the action of SIE so it does not mean any other things. This flag can be cleared by MCU.

Bit2 **UC_FIFO_FLUSH** : MCU sets this bit for initializing the FIFO. MCU can not clear IN_PKT_RDY so if MCU want to clear IN_PKT_RDY after set then MCU should issue UC_FIFO_FLUSH for clearing IN_PKT_RDY.

Bit1 **FORCE_STALL** : MCU sets this bit for sending stall packet. This flag will not be cleared by SIE. So MCU should clear this flag for stopping stall condition. Device will send stall until this flag is cleared.

Bit0 **OUT_PKT_RDY** : SIE sets this bit, if the device receive valid data from host. An interrupt is generated, when SIE sets this bit. MCU should download data and clears this bit by writing "1" to CLEAR_OUT_PKT_RDY bit at the end of execution.

ENDPOINT 0 WRITE COUNT REGISTER (EP0BCNT)

EP0BCNT register contains data count value, some monitoring and flow control flag. EP0BCNT is located at F3H, Page 0 and read addressable.

- Bit7 **DATA_TOGGLE** : This bit is a read only flag. This flag is just for monitoring the data toggle sequence.
- Bit6 **TOKEN** :.This flag is for monitoring. If this value is set then it means the last received token packet is SETUP token and if the value is "0" then the last received token packet is OUT or IN packet.
- Bit5 **OVER_8** :.If device receive over 8 bytes SETUP or OUT transaction then the device does not answer to these transaction and set this flag as a error indicator.
- Bit4 **ENABLE** :. MCU set this bit for disabling endpoint 0. Device does not answer to any traffic if addressed to endpoint 0 until this bit is cleared.
- Bit3 **EP0WRT_CNT[3]**.
- Bit2 **EP0WRT_CNT[2]**.
- Bit1 **EP0WRT_CNT[1]**.
- Bit0 **EP0WRT_CNT[0]** : SIE store data count after receive valid data from host. The maximum value is 8. And if MCU downloading the FIFO then this value also decreased according to remain data count.

ENDPOINT 1 WRITE COUNT REGISTER (EP1BCNT)

EP1BCNT register contains data count value, some monitoring and flow control flag. EP1BCNT is located at FCH, Page 0 and read/write addressable.

- Bit7 **DATA_TOGGLE** : This bit is a read only flag. This flag is just for monitoring the data toggle sequence.
- Bit6 **Reserved.**
- Bit5 **OVER_8** :.If device receive over 8 bytes SETUP or OUT transaction then the device does not answer to these transaction and set this flag as a error indicator.
- Bit4 **ENABLE** :. MCU set this bit for disabling endpoint 1. Device does not answer to any traffic if addressed to endpoint 1 until this bit is cleared.
- Bit3 **EP1WRT_CNT[3].**
- Bit2 **EP1WRT_CNT[2].**
- Bit1 **EP1WRT_CNT[1].**
- Bit0 **EP1WRT_CNT[0]** : SIE store data count after receive valid data from host. The maximum value is 8. And if MCU downloading the FIFO then this value also decreased according to remain data count.

ENDPOINT 2 WRITE COUNT REGISTER (EP2BCNT)

EP2BCNT register contains data count value, some monitoring and flow control flag. EP2BCNT is located at FDH, Page 0 and read/write addressable.

- Bit7 **DATA_TOGGLE** : This bit is a read only flag. This flag is just for monitoring the data toggle sequence.
- Bit6 **Reserved.**
- Bit5 **OVER_8** :.If device receive over 8 bytes SETUP or OUT transaction then the device does not answer to these transaction and set this flag as a error indicator.
- Bit4 **ENABLE** :. MCU set this bit for disabling endpoint 2. Device does not answer to any traffic if addressed to endpoint 2 until this bit is cleared.
- Bit3 **EP2WRT_CNT[3].**
- Bit2 **EP2WRT_CNT[2].**
- Bit1 **EP2WRT_CNT[1].**
- Bit0 **EP2WRT_CNT[0]** : SIE store data count after receive valid data from host. The maximum value is 8. And if MCU downloading the FIFO then this value also decreased according to remain data count.

ENDPOINT MODE REGISTER (EPMODE)

EPMODE register contains the field which defines USB reset signal length and the field which defines the direction of endpoints. EPMODE is located at FBH, Page 0 and read/write addressable.

Bit7 **RESET_LENGTH[1]**.

Bit6 **RESET_LENGTH[0]** : This field defines the length of USB reset signal. The reset value is "00". MCU can control USB reset length through this field. The definition is as below.

- "00" : $22.4 \mu\text{s} + \alpha$
- "01" : $12.0 \mu\text{s} + \alpha$
- "10" : $6.0 \mu\text{s} + \alpha$.
- "11" : $4.0 \mu\text{s} + \alpha$ ($\alpha \cong 0.66 \mu\text{s}$)

Bit5 **Reserved.**

Bit4 **Reserved.**

Bit3 **Reserved.**

Bit2 **Reserved.**

Bit1 **ENDPOINT_MODE[1]** : MCU can defines direction of interrupt transfer. If this value is "1" then endpoint 2 act as a OUT interrupt endpoint and if this value is "0" then endpoint 2 act as a IN interrupt endpoint. The reset value is "0".

Bit0 **ENDPOINT_MODE[0]** : MCU can defines direction of interrupt transfer. If this value is "1" then endpoint 1 act as a OUT interrupt endpoint and if this value is "0" then endpoint 1 act as a IN interrupt endpoint. The reset value is "0".

USB POWER MANAGEMENT REGISTER (PWRMGR)

PWRMGR register interacts with the Host's power management system to execute system power events such as SUSPEND or RESUME. And this register also contains monitoring field for detail control of MCU. This register is located at address F8H, Page 0 and is read/write addressable.

- Bit7 **Reserved.**
- Bit6 **Reserved.**
- Bit5 **Reserved.**
- Bit4 **VPIN** : MCU can read the VPIN value through this bit. This value is the output of transceiver and the input of USB module. And this value is different from the port VPIN value. Port VP value is the input of controller, not USB module.
- Bit3 **VMIN** : MCU can read the VMIN value through this bit. This value is the output of transceiver and the input of USB module. And this value is different from the port VMIN. Port VM value is the input of controller, not USB module.
- Bit2 **CLEAR_SUSP_CNT** : MCU write "1" value to this bit for clearing suspend counter which count 3 ms. And during this value stay "1" the suspend counter does not proceed. That means the USB controller can not go into suspend state during this value stays "1".
- Bit1 **SEND_RESUME**: While in SUSPEND state, if the MCU wants to initiate RESUME, it writes "1" to this register for 10ms (maximum of 15ms), and clears this register. In SUSPEND mode if this bit reads "1" then SIE generates RESUME signaling to upstream.
- Bit0 **SUSPEND_STATE**: Suspend state is set when the MCU sets suspend interrupt. This bit is cleared automatically when:
- MCU writes "0" to SEND_RESUME bit to end the RESUME signaling (after SEND_RESUME is set for 10ms).
 - MCU receives RESUME signaling from the Host while in SUSPEND mode.

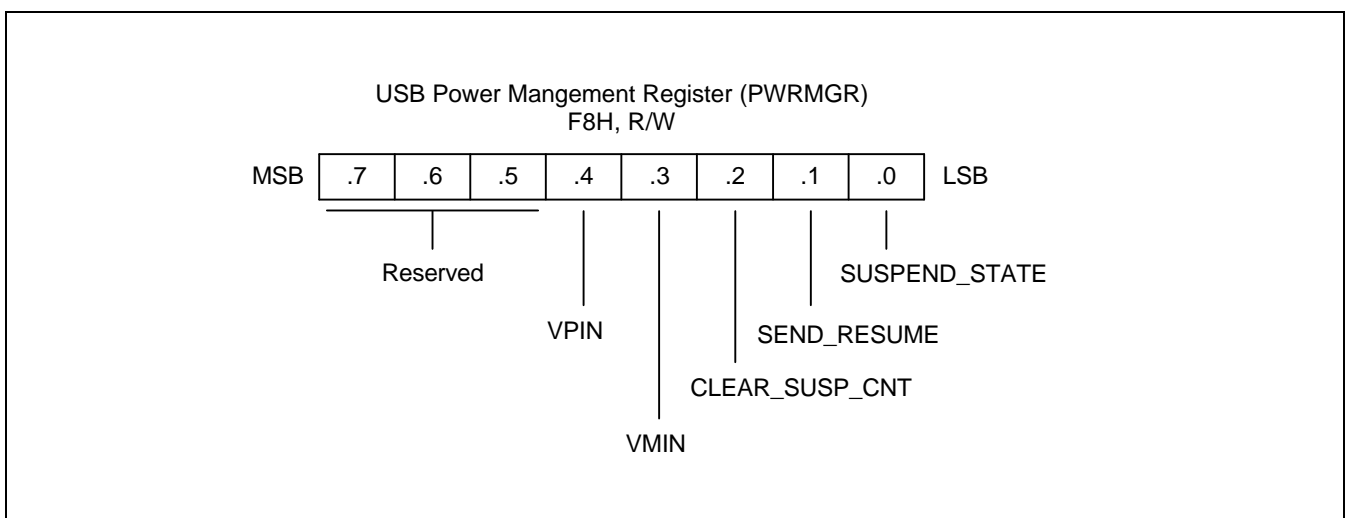


Figure 13-6. USB Power Management Register (PWRMGR)

CONTROL ENDPOINT FIFO REGISTER (EP0FIFO)

This register is bi-directional, 8 byte depth FIFO used to transfer Control Endpoint data. EP0FIFO is located at address F4H and is read/write addressable.

Initially, the direction of the FIFO, is from the Host to the MCU. After a setup token is received for a control transfer, that is, after MCU unload the setup token bytes, and clears OUT_PKT_RDY, the direction of FIFO is changed automatically from MCU to the Host.

INTERRUPT ENDPOINT 1 FIFO REGISTER (EP1FIFO)

EP1FIFO is an bi-direction 8-byte depth FIFO used to transfer data from the MCU to the Host or from the Host to the MCU. MCU writes data to this register, and when finished set IN_PKT_RDY. Meanwhile, when USB receives valid data through this register, it sets OUT_PKT_RDY, after MCU unload Data bytes, and clears OUT_PKT_RDY, This register is located at address F5H.

INTERRUPT ENDPOINT 2 FIFO REGISTER (EP2FIFO)

EP2FIFO is an bi-direction 8-byte depth FIFO used to transfer data from the MCU to the Host or from the Host to the MCU. MCU writes data to this register, and when finished set IN_PKT_RDY. Meanwhile, when USB receives valid data through this register, it sets OUT_PKT_RDY, after MCU unload Data bytes, and clears OUT_PKT_RDY, This register is located at address FAH.

USB INTERRUPT PENDING REGISTER (USBPND)

USBPND register has the interrupt bits for endpoints and power management. *This register is cleared once read by MCU.* While any one of the bits is set, an interrupt is generated. USBPND is located at address F6H.

Bit7–6 Not used

Bit5 **USB_RST_PND:** This bit is set, when USB reset signal is received.

Bit4 **ENDPT2_PND:** This bit is set, when Endpoint 2 needs to be received.

Bit3 **RESUME_PND:** While in suspend mode, if resume signaling is received this bit gets set.

Bit2 **SUSPEND_PND:** This bit is set, when suspend signaling is received.

Bit1 **ENDPT1_PND:** This bit is set, when Endpoint 1 needs to be serviced.

Bit0 **ENDPT0_PND:** This bit is set, when Endpoint 0 needs to be serviced. It is set under any one of the following conditions:

- OUT_PKT_RDY is set.
- IN_PKT_RDY gets cleared.
- SENT_STALL gets set.
- DATA_END gets cleared.
- SETUP_END gets set.

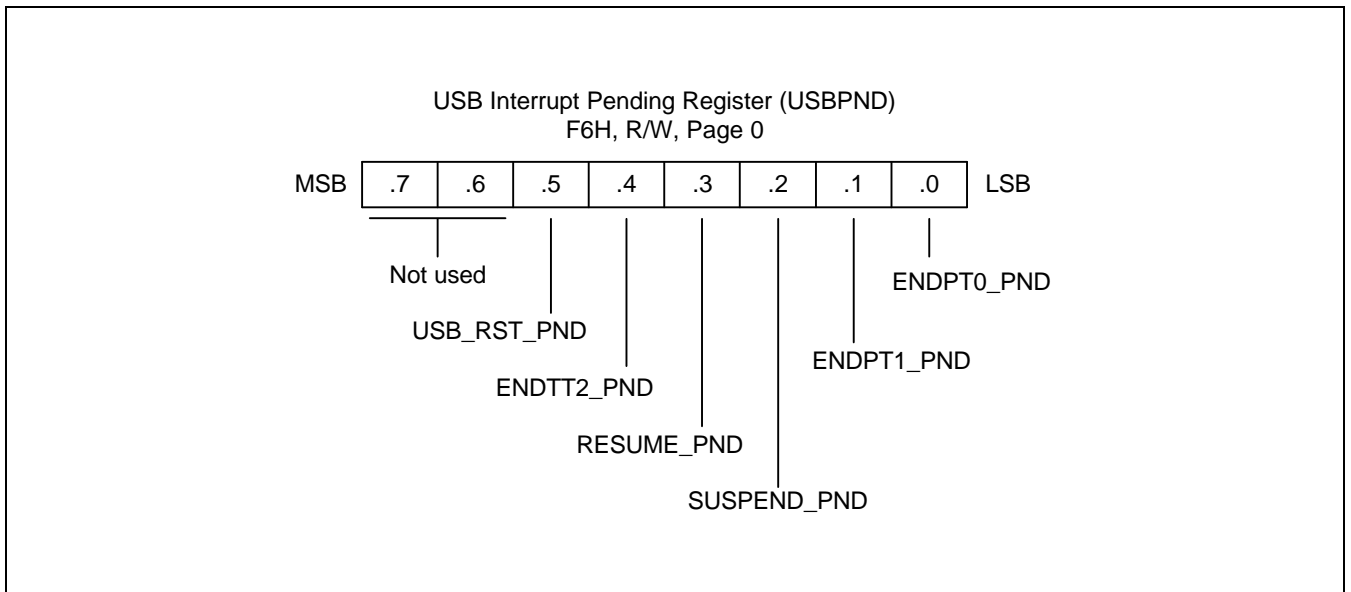


Figure 13-7. USB Interrupt Pending Register (USBPND)

USB INTERRUPT ENABLE REGISTER (USBINT)

USBINT is located at address F7H, Page 0 and is read/write addressable. This register serves as an interrupt mask register. If the corresponding bit = 1 then the respective interrupt is enabled.

By default, all interrupts except suspend interrupt is enabled. Interrupt enables bits for suspend and resume is combined into a single bit (bit 2).

Bit7–5 Not used

Bit4 ENABLE_USB_RST_INT:

- 1 Enable USB RESET INTERRUPT (default)
- 0 Disable USB RESET INTERRUPT

Bit3 ENABLE_ENDPT2_INT:

- 1 Enable ENDPOINT 2 INTERRUPT (default)
- 0 Disable ENDPOINT 2 INTERRUPT

Bit2 ENABLE_SUSPEND_RESUME_INT:

- 1 Enable SUSPEND and RESUME INTERRUPT
- 0 Disable SUSPEND and RESUME INTERRUPT (default)

Bit1 ENABLE_ENDPT1_INT:

- 1 Enable ENDPOINT 1 INTERRUPT (default)
- 0 Disable ENDPOINT 1 INTERRUPT

Bit0 ENABLE_ENDPT0_INT:

- 1 Enable ENDPOINT 0 INTERRUPT (default)
- 0 Disable ENDPOINT 0 INTERRUPT

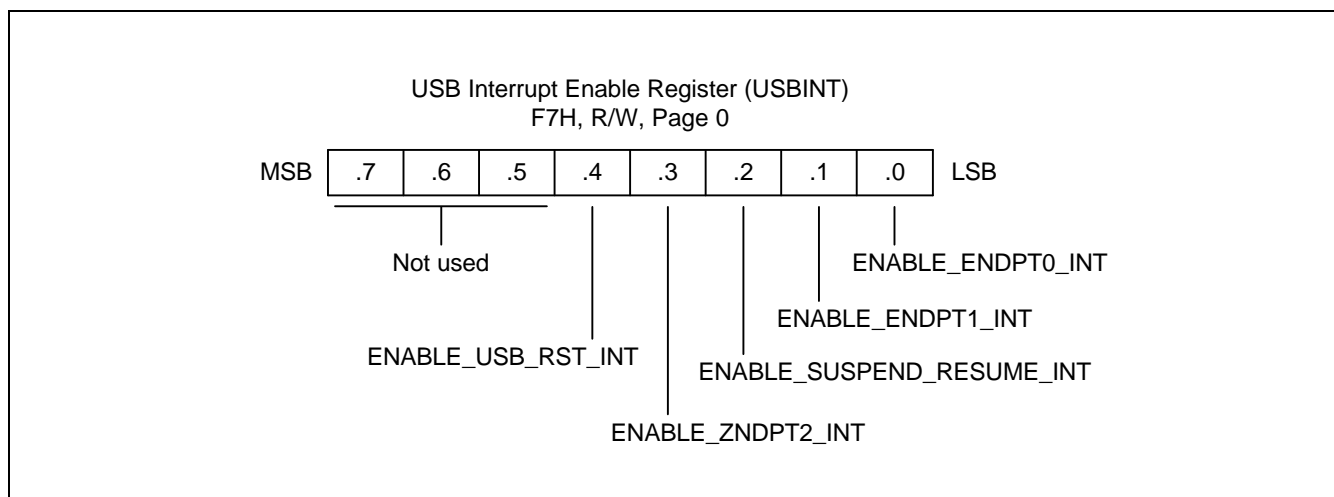


Figure 13-8. USB Interrupt Enable Register (USBINT)

USB CONTROL REGISTER (USBCON)

USBCON is for the control of USB data line and the control of reset .This register is located at address FEH, Page 0 and is read/write addressable.

- Bit5 **DP/DM Control:** When this bit is set , DP/DM lines can be controlled by MCU as bellows
- Bit4 **DP:** On the condition of bit5 set, if this bit is 1 , DP line is to be high and the other case this bit is 0 DP line is low .
- Bit3 **DM:** On the condition of bit5 set, if this bit is 1 , DM line is to be high and the other case this bit is 0 DM line is low .
- Bit2 **USB_RESET_EN:** When this bit is set , it is USB is made reset , which trigger MCU reset automatically
- Bit1 **MCU_RESET:** When this bit is set , MCU makes USB reset
- Bit0 **USB_RSTN:** USB reset status bit
 0 : USB is not reset
 1 : USB is reset

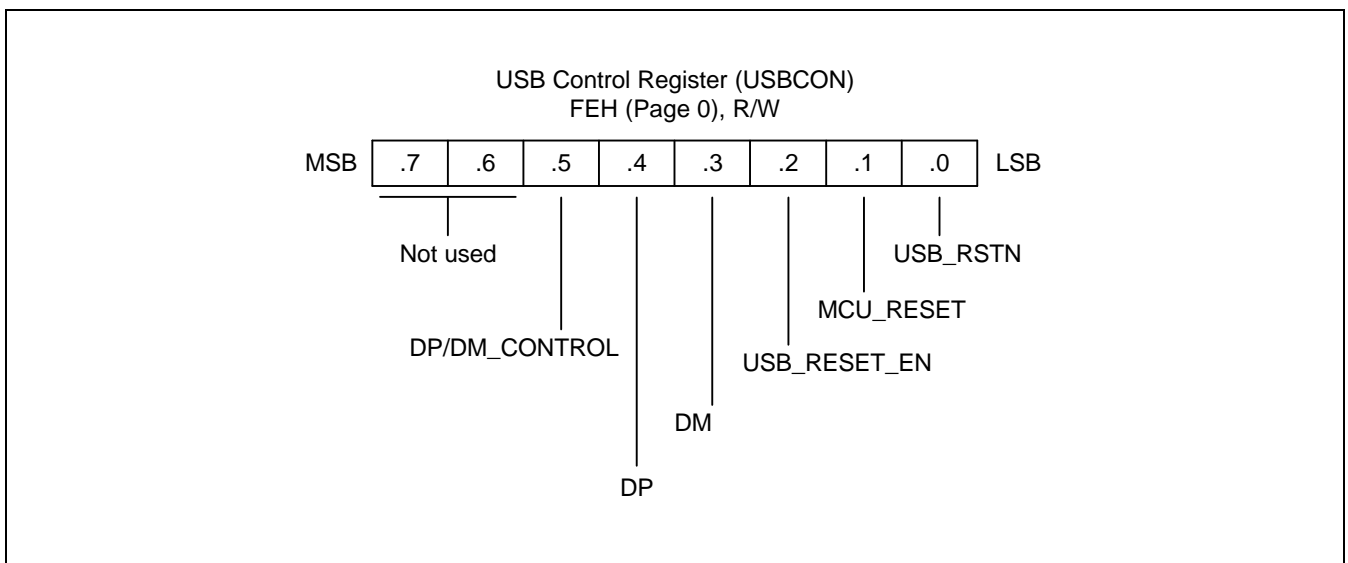


Figure 13-9. USB Control Register (USBCON)

XCON REGISTER (XCON)

XCON register is an 8-bit register that is used to adjust signal quality and Port2 (D+/D-) configuration. This register is located at address FEH (Page1) and is read/write addressable.

Bit7 Pull-up resistor at(D-) enable bit:

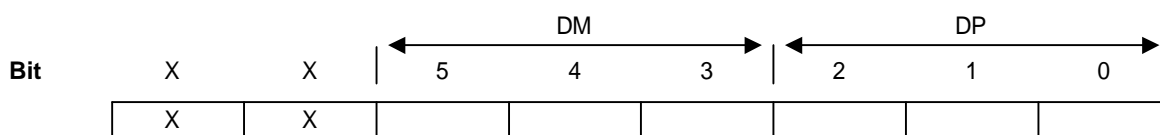
0 : Pull up resistor at (D-) disable

1 : Pull up resistor at (D-) enable

Bit7 GPIO or USB Port Select bit:

0 : General in/out port enable

1 : USB port enable



Edge Control	Bit 5, 2	Bit 4, 1	Bit 3, 0	DLY Value	Unit
Rise edge	0	0	0	DLY 0	About 2.5 nsec
		0	1	DLY 1	
		1	0	DLY 2	
		1	1	DLY 4	
Fall edge	1	0	0	DLY 0	
		0	1	DLY 1	
		1	0	DLY 2	
		1	1	DLY 4	

NOTE: DLY means delay time of rising/falling time.

NUM	HEX Value	Result, DM \searrow DP \nearrow DM \searrow DP \nearrow (V _{DD} = 5.12 V) x - point 1 x - point 2	
1	0x00	Default value, 0.88 V, 1.19 V	
2	0x38	1.58 V, 1.55 V	
3	0x3C	1.50 V, 1.50 V	
4	0x3D	1.65 V, 1.65 V	Good
5	0x3E	1.80 V, 1.80 V	

NOTE: This value is only for OTP products.

14 SUB RC OSCILLATOR

OVERVIEW

The S3C9664 have a programmable SUB RC oscillator. During IDLE or STOP, programmable SUB RC oscillator generated interrupt using SUB RC oscillator control register (SUBCON).

SUB RC OSCILLATOR CONTROL REGISTER (SUBCON)

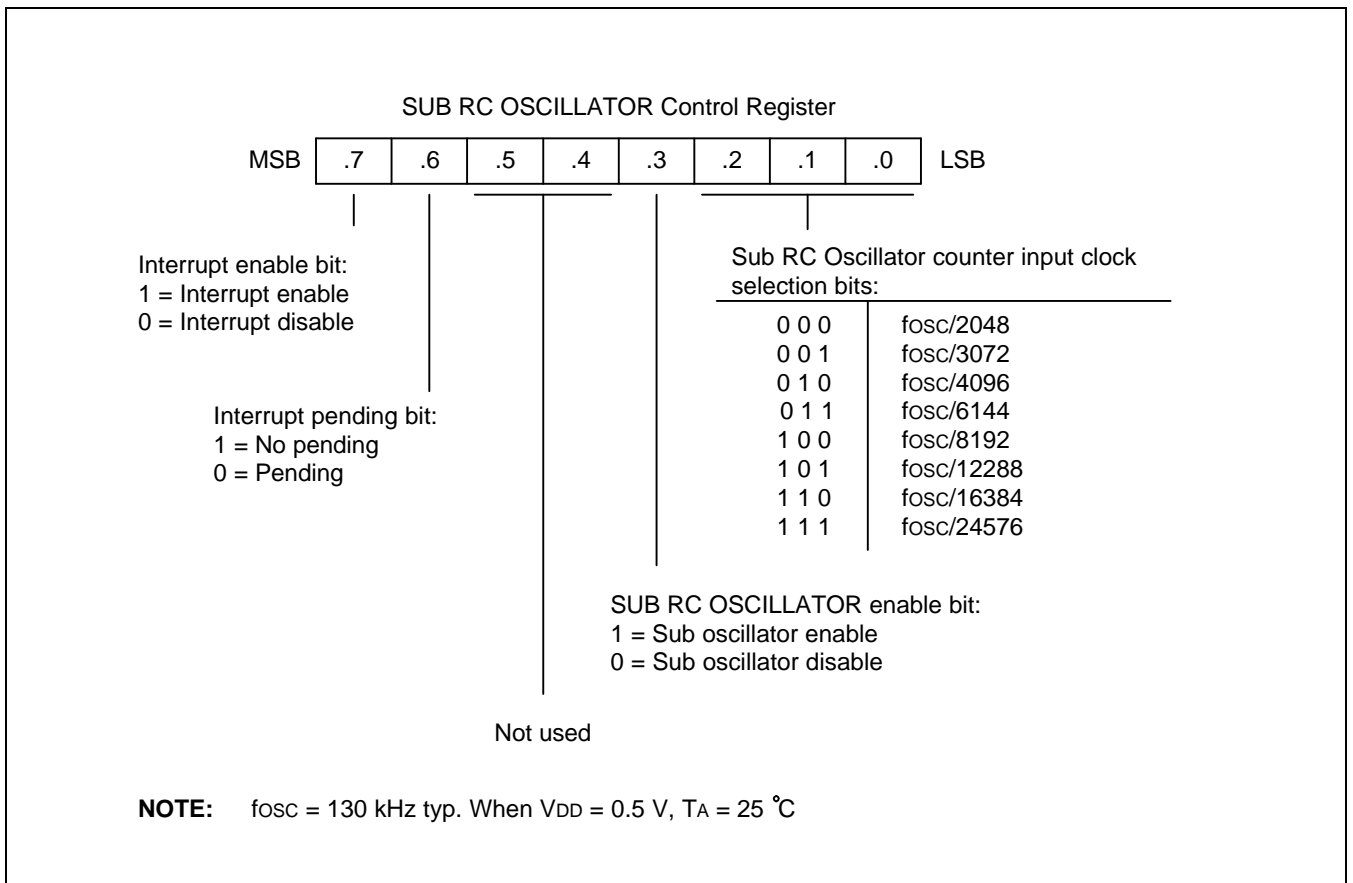


Figure 14-1. SUB RC OSCILLATOR Control Register

15

LVR (LOW VOLTAGE RESET)

OVERVIEW

The S3C9664 have a LVR (Low Voltage Reset) for power on reset and voltage reset.

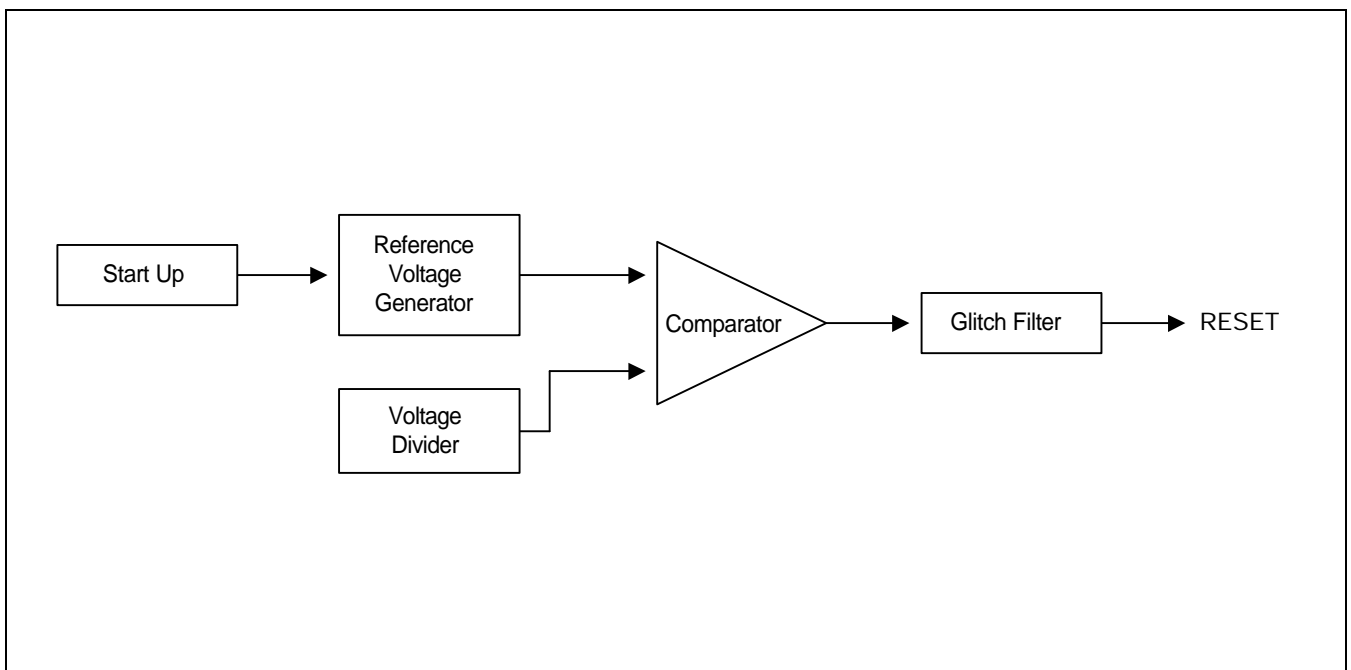


Figure 15-1. LVR Architecture

- Low Voltage Reset generated RESET signal.
- Start Up Circuit: Start up reference voltage generator circuit when device powered.
- Reference Voltage Generator: Supply Voltage independent reference voltage generator.
- Voltage Divider: Divide supply voltage by "N"
- Comparator: Compare reference voltage and divided voltage.
- Glitch Filter: Remove glitch and noise signal.

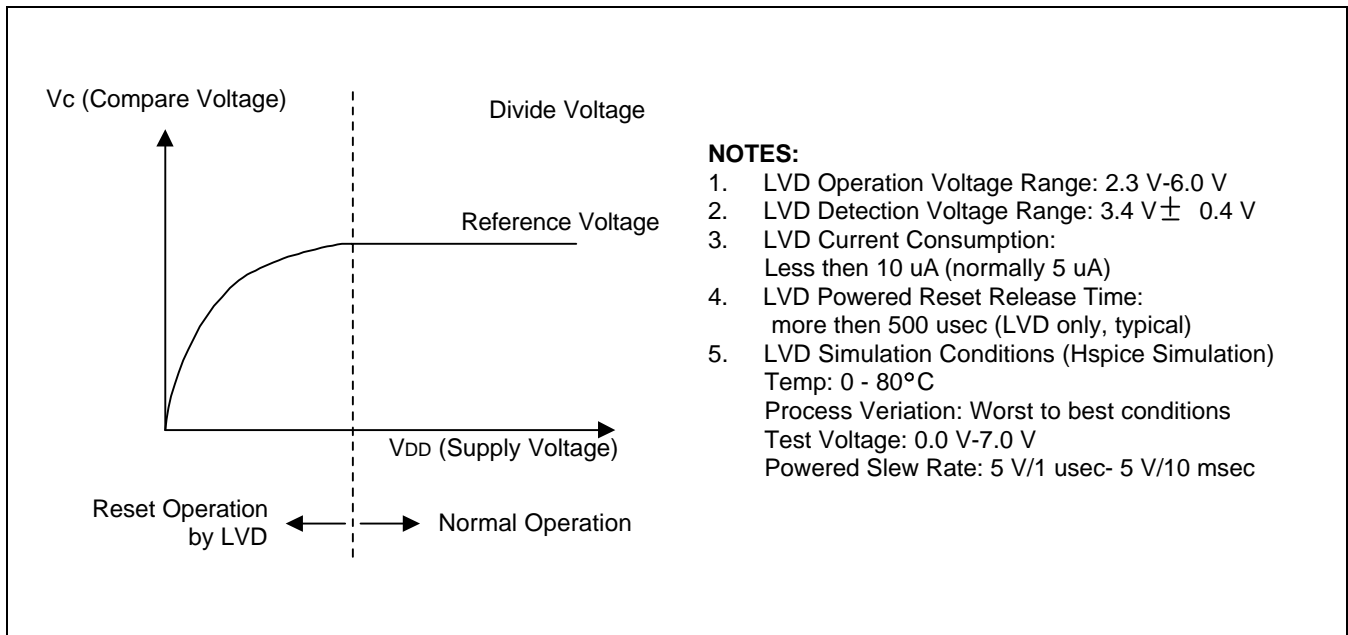


Figure 15-2. LVR Characteristics

LVR AND POWER ON RESET OPERATIONS

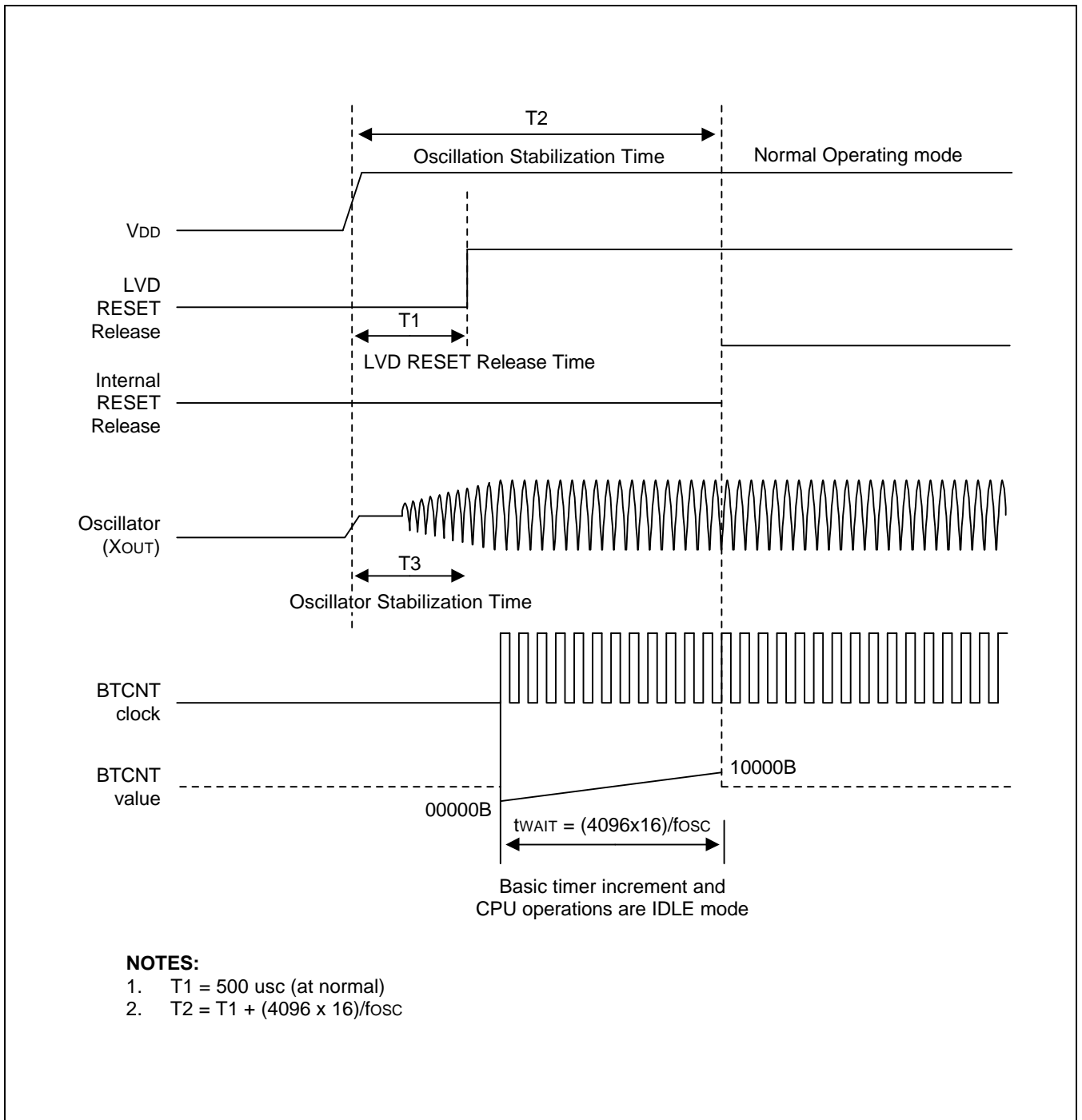


Figure 15-3. LVR and Power On RESET Operation

16 ELECTRICAL DATA

OVERVIEW

In this section, the following S3C9664 electrical characteristics are presented in tables and graphs:

- Absolute maximum ratings
- D.C. electrical characteristics
- I/O capacitance
- A.C. electrical characteristics
- Oscillator characteristics
- Operating voltage range
- Oscillation stabilization time
- Clock timing measurement points at X_{IN}
- Data retention supply voltage in Stop mode
- Stop mode release timing when initiated by a RESET
- Stop mode release timing when initiated by an external interrupt
- Characteristic curves
- AD Converter Electrical Characteristics

Table 16-1. Absolute Maximum Ratings

 $(T_A = 25^\circ\text{C})$

Parameter	Symbol	Conditions	Rating	Unit
Supply voltage	V_{DD}	–	– 0.3 to + 6.5	V
Input voltage	V_I	All ports	– 0.3 to $V_{DD} + 0.3$	V
Output voltage	V_O	All output ports	– 0.3 to $V_{DD} + 0.3$	V
Output current high	I_{OH}	One I/O pin active	– 18	mA
		All I/O pins active	– 60	
Output current low	I_{OL}	One I/O pin active	+ 30	mA
		Total pin current for ports 0, 1, 2	+ 100	
Operating temperature	T_A	–	0 to + 85	°C
Storage temperature	T_{STG}	–	– 60 to + 150	

Table 16-2. D.C. Electrical Characteristics

(T_A = -40°C to +85°C, V_{DD} = 4.0 V to 5.25 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input highvoltage	V _{IH1}	All input pins except V _{IH2} , D+, D-	0.8 V _{DD}	-	V _{DD}	V
	V _{IH2}	X _{IN}	V _{DD} - 0.5		V _{DD}	
Input low voltage	V _{IL1}	All input pins except V _{IL2} , D+, D-	-	-	0.2 V _{DD}	
	V _{IL2}	X _{IN}	-	-	0.4	
Output high voltage	V _{OH}	V _{DD} = 4.0 V - 5.25 V I _{OH} = -200 mA All output ports except D+, D-	V _{DD} - 1.0	-	-	
Output low voltage	V _{OL}	V _{DD} = 4.0 V - 5.25 V I _{OL} = 2 mA All output ports except D+, D-	-	-	0.4	
Input high leakage current	I _{LIH1}	V _{IN} = V _{DD} All inputs except I _{LIH2} except D+, D-, X _{OUT}	-	-	3	μA
	I _{LIH2}	V _{IN} = V _{DD} , X _{IN}	-	-	20	
Input low leakage current	I _{LIL1}	V _{IN} = 0 V All inputs except I _{LIL2} except D+, D-, X _{OUT}	-	-	-3	
	I _{LIL2}	V _{IN} = 0 V, X _{IN}	-	-	-20	
Output high leakage current	I _{LOH}	V _{OUT} = V _{DD} All output pins except D+, D-	-	-	3	
Output low leakage current	I _{LOL}	V _{OUT} = 0 V All output pins except D+, D- X _{OUT}	-	-	-3	
Pull-up resistors	R _{L1}	V _{IN} = 0 V, V _{DD} = 5.0 V, Port 0, Port 1, Port 2	25	50	100	KΩ
	R _{L2}	V _{IN} = 0 V, V _{DD} = 5.0 V, RESET only	100	220	400	
Pull-down resistors	R _{L3}	V _{IN} = 0 V, V _{DD} = 5.0 V, Port 0	25	50	100	KΩ
Supply current	I _{DD1}	Normal operation mode, V _{DD} = 5 V ± 10 %, 6 MHz, CPU clock	-	6.5	15	mA
	I _{DD2}	IDLE mode V _{DD} = 5 V ± 10 %, 6 MHz, CPU clock	-	4	8	
	I _{DD3}	Stop mode, oscillator stop V _{DD} = 5 V ± 10 %,	-	150	300	μA

NOTES:

1. Supply current does not include current drawn through internal pull-up resistors or external output current load.
2. This parameter is guaranteed, but not tested (include D+, D-).
3. Only in 4.0 V to 5.25 V, D+ and D- satisfy the USB spec 1.1.

Table 16-3. Input/Output Capacitance

 $(T_A = 0^\circ\text{C to } +85^\circ\text{C}, V_{DD} = 0\text{ V})$

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input capacitance	C_{IN}	$f = 1\text{ MHz}$; unmeasured pins are connected to V_{SS}	-	-	10	pF
Output capacitance	C_{OUT}					
I/o capacitance	C_{IO}	expect X_{IN} , X_{OUT}				
XI/XO capacitance	C_{XI} , C_{XO}	X_{IN} , X_{OUT}			33	

Table 16-4. A.C. Electrical Characteristics

 $(T_A = -40^\circ\text{C to } +85^\circ\text{C}, V_{DD} = 4.0\text{ V to } 5.25\text{ V})$

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Noise filter	t_{NF1H} , t_{NF1L}	P1 (RC delay)	100	-	200	ns

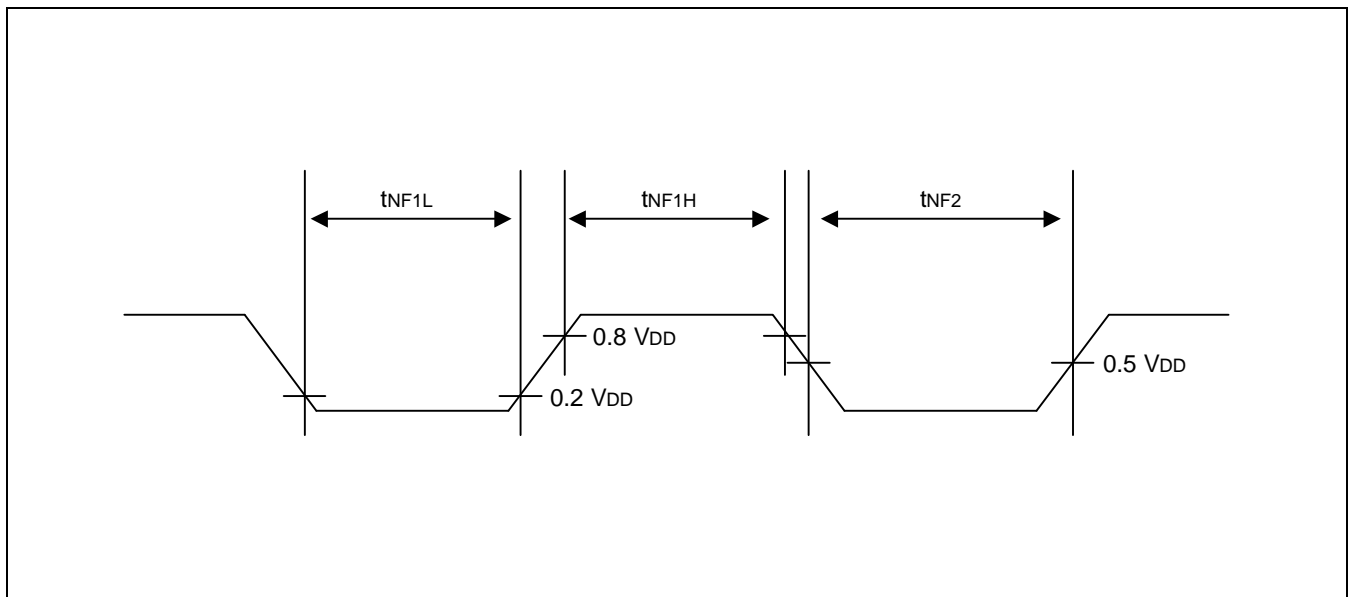


Figure 16-1. Input Timing for External Interrupts

Table 16-5. Oscillator Characteristics

 $(T_A = 0^\circ\text{C} + 85^\circ\text{C})$

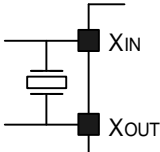
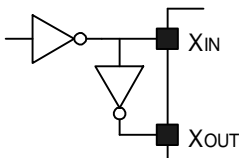
Oscillator	Clock Circuit	Test Condition	Min	Typ	Max	Unit
Main crystal Main ceramic (f_{OSC})		Oscillation frequency $V_{\text{DD}} = 4.0\text{ V} - 5.25\text{ V}$	–	6.0	–	MHz
External clock		Oscillation frequency $V_{\text{DD}} = 4.0\text{ V} - 5.25\text{ V}$	–	6.0	–	

Table 16-6. Oscillation Stabilization Time

 $(T_A = 0^\circ\text{C} + 85^\circ\text{C}, V_{\text{DD}} = 4.0\text{ V to } 5.25\text{ V})$

Oscillator	Test Condition	Min	Typ	Max	Unit
Main crystal	$V_{\text{DD}} = 4.0\text{ V to } 5.25\text{ V}, f_{\text{OSC}} > 6.0\text{ MHz}$ (Oscillation stabilization occurs when V_{DD} is equal to the minimum oscillator voltage range.)	–	–	10	ms
Main ceramic					
Oscillator stabilization wait time	t_{WAIT} stop mode release time by a reset	–	$2^{16}/f_{\text{OSC}}$	–	
	t_{WAIT} stop mode release time by an interrupt	–	–	–	

NOTE: The oscillator stabilization wait time, t_{WAIT} , when it is released by an interrupt, is determined by the setting in the basic timer control register, BTCON.

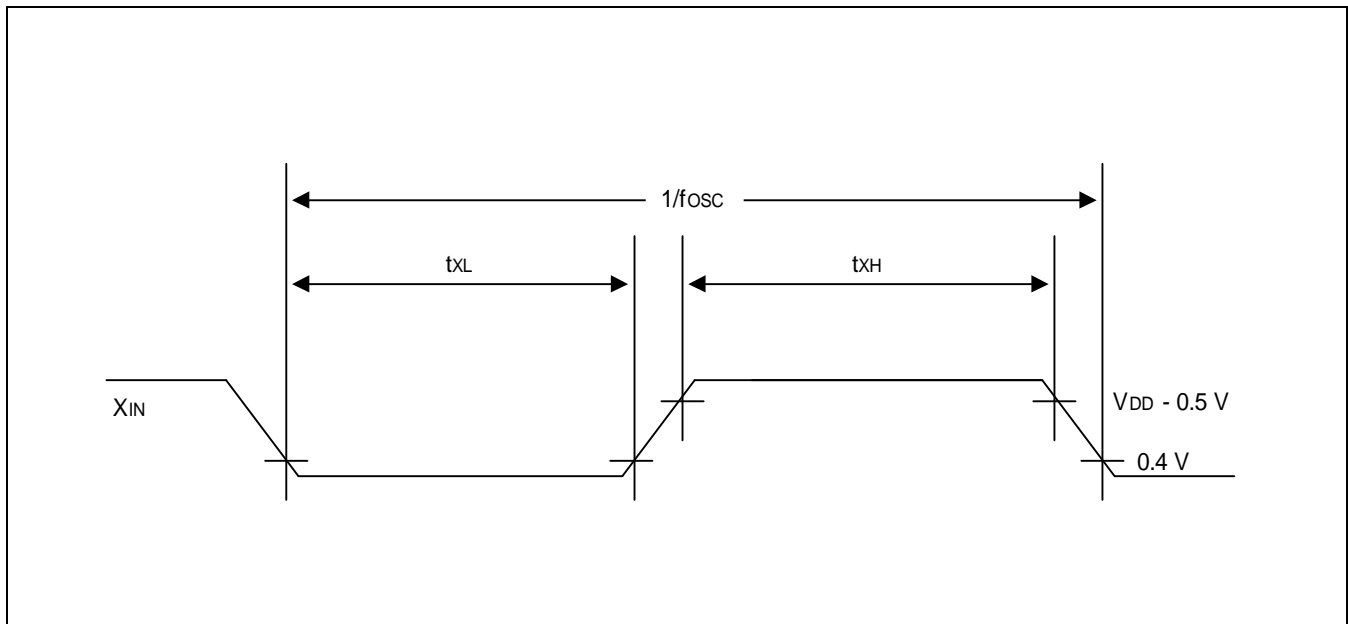
Figure 16-2. Clock Timing Measurement Points at X_{IN}

Table 16-7. Data Retention Supply Voltage in Stop Mode

($T_A = 0^\circ C$ to $+70^\circ C$)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V_{DDDR}	Stop mode	2.0	–	6	V
Data retention supply current	I_{DDDR}	Stop mode; $V_{DDDR} = 2.0 V$	–	–	5	μA

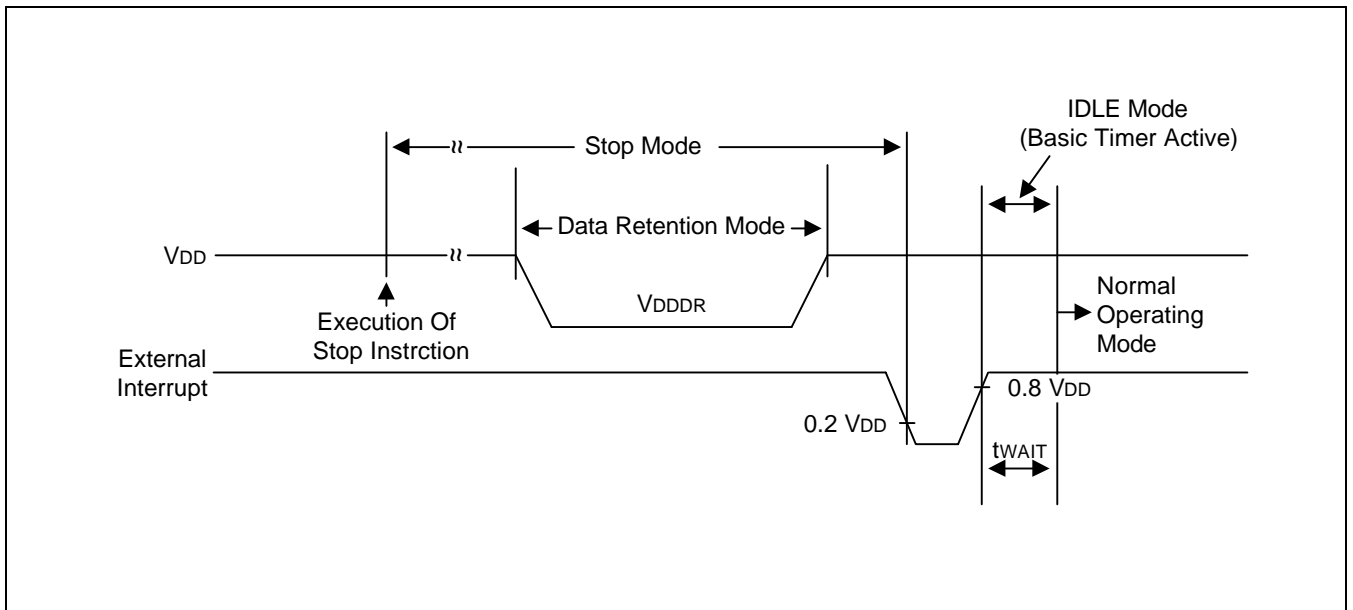


Figure 16-3. Stop Mode Release Timing When Initiated by an External Interrupt

Table 16-8. Low Speed Source Electrical Characteristics (USB)

(T_A = 0°C to +85°C, Voltage Regulator Output V_{33OUT} = 2.8 V to 3.5 V, typ 3,3 V)

Parameter	Symbol	Conditions	Min	Max	Unit
Transition Time:					
Rise Time	Tr	CL = 200 pF	75	–	ns
Fall Time	Tf	CL = 650 pF	–	300	
		CL = 200 pF	75	–	
		CL = 650 pF	–	300	
Rise/Fall Time Matching	Trfm	(Tr/Tf) CL = 50 pF	80	125	%
Output Signal Crossover Voltage	Vcrs	CL = 50 pF	1.3	2.0	V
Output Voltage Regulator Built-in	V _{33OUT}	V _{DD} = 4.0–5.25 V	2.8	3.6	V

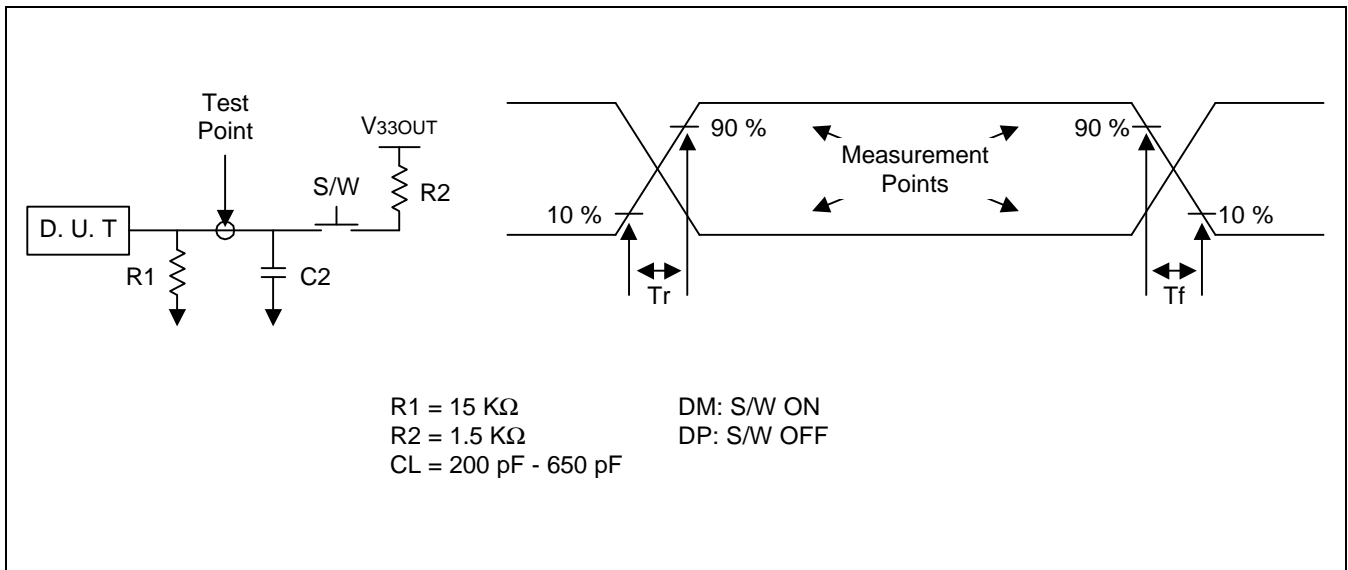


Figure 16-4. USB Data Signal Rise and Fall Time

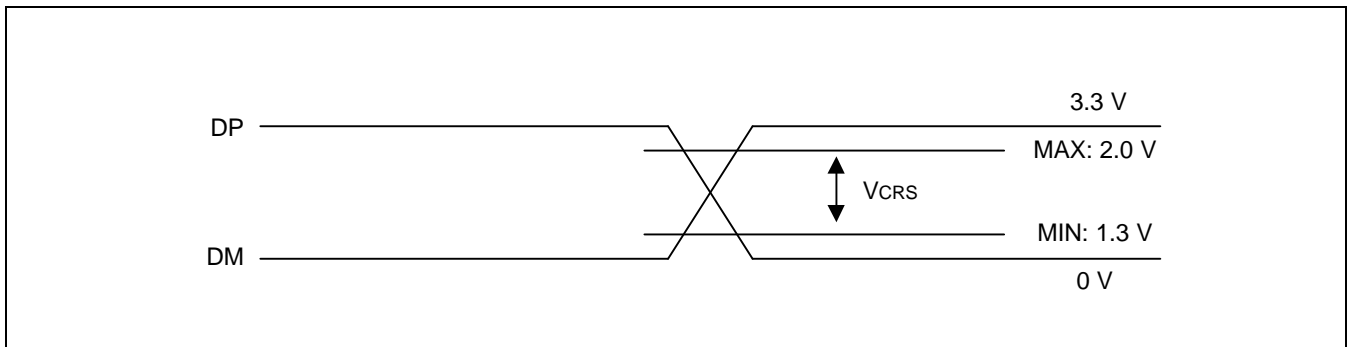


Figure 16-5. USB Output Signal Crossover Point Voltage

Table 16-9. A/D Converter Electrical Characteristics

(T_A = -40°C to +85°C, V_{DD} = 4.2 V to 5.25 V, V_{SS} = 0 V) S3C9664/P9664: 10-bit ADC

Parameter	Symbol	Test Conditions	Min	Typ	Max	Unit
Total accuracy		V _{DD} = 5.12 V CPU clock = 10 MHz V _{DD} = 5.12 V V _{SS} = 0 V	-	-	± 3	LSB
Integral linearity error	ILE	-		-	± 2	
Differential linearity error	DLE	-		-	± 1	
Offset error of top	EOT	-		- 1	± 3	
Offset error of bottom	EOB	-		- 1	± 2	
Conversion time ⁽¹⁾	t _{CON}	f _{cpu} = 10 MHz	-	50x4/ f _{OSC}	-	μs
Analog input voltage	V _{IAN}	-	V _{SS}	-	V _{DD}	V
Analog input impedance	R _{AN}	-	2	-	-	MΩ
Analog input current	I _{ADIN}	V _{DD} = 5 V	-	-	10	μA
ADC block current ⁽²⁾	I _{ADC}	V _{DD} = 5 V	-	1	3	mA
		V _{DD} = 5 V Power down mode	-	100	500	nA

NOTES:

1. 'Conversion time' is the time required from the moment a conversion operation starts until it ends.
2. I_{ADC} is operating current during A/D conversion.

17 MECHANICAL DATA

OVERVIEW

This section contains the following information about the device package:

- Package dimensions in millimeters
- Pad diagram

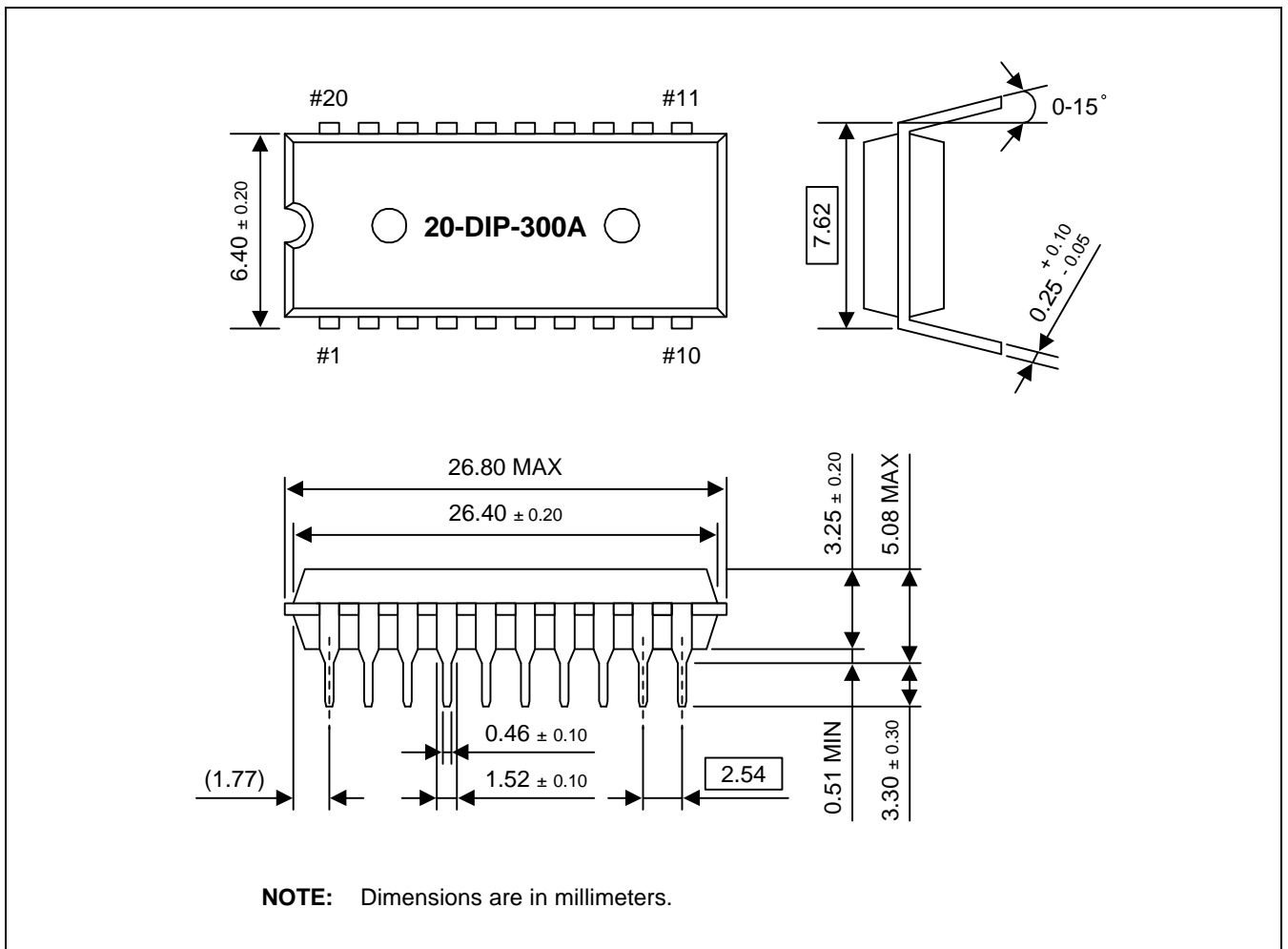


Figure 17-1. 20-DIP-300A Package Dimensions

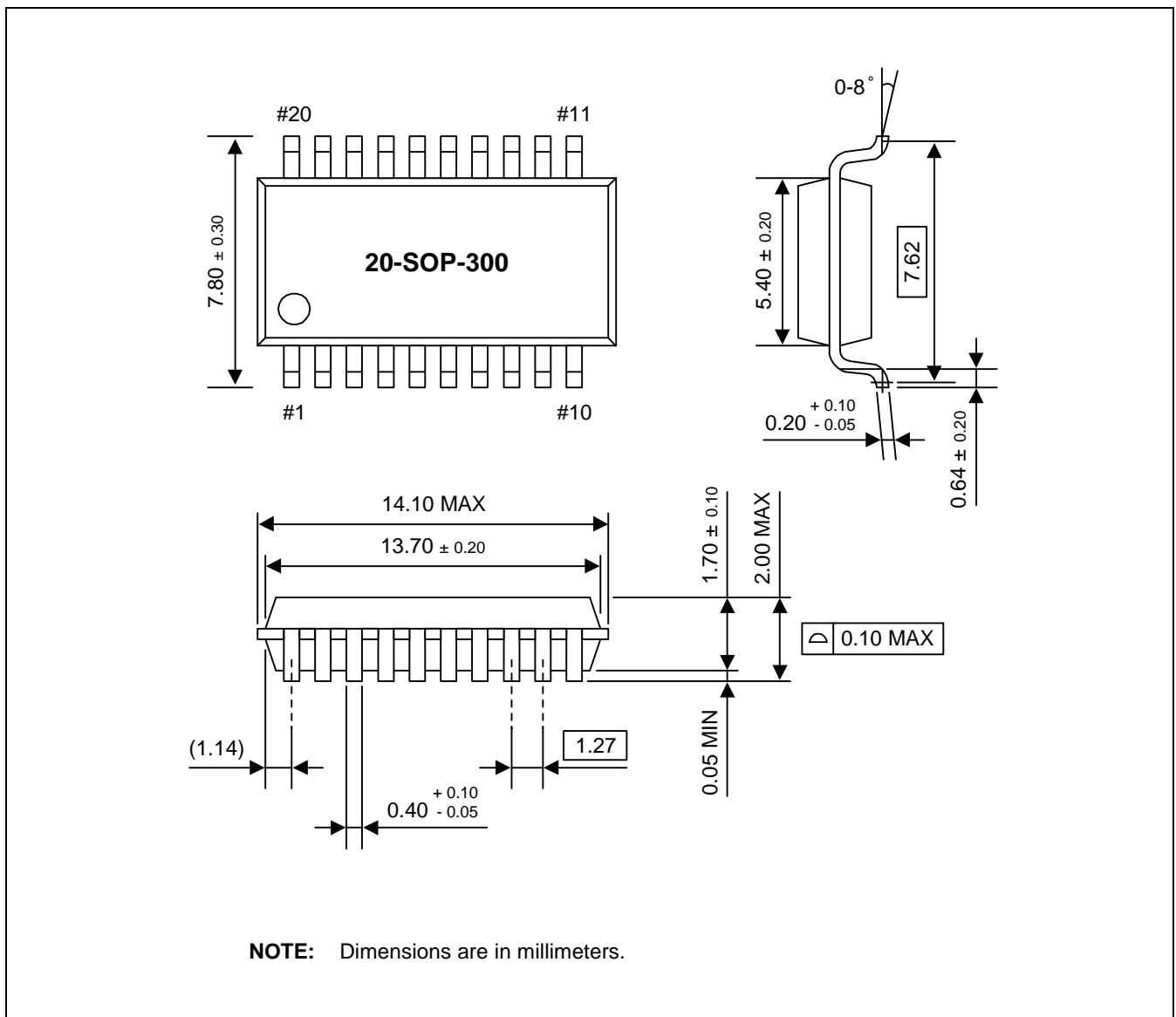


Figure 17-2. 20-SOP-300 Package Dimensions

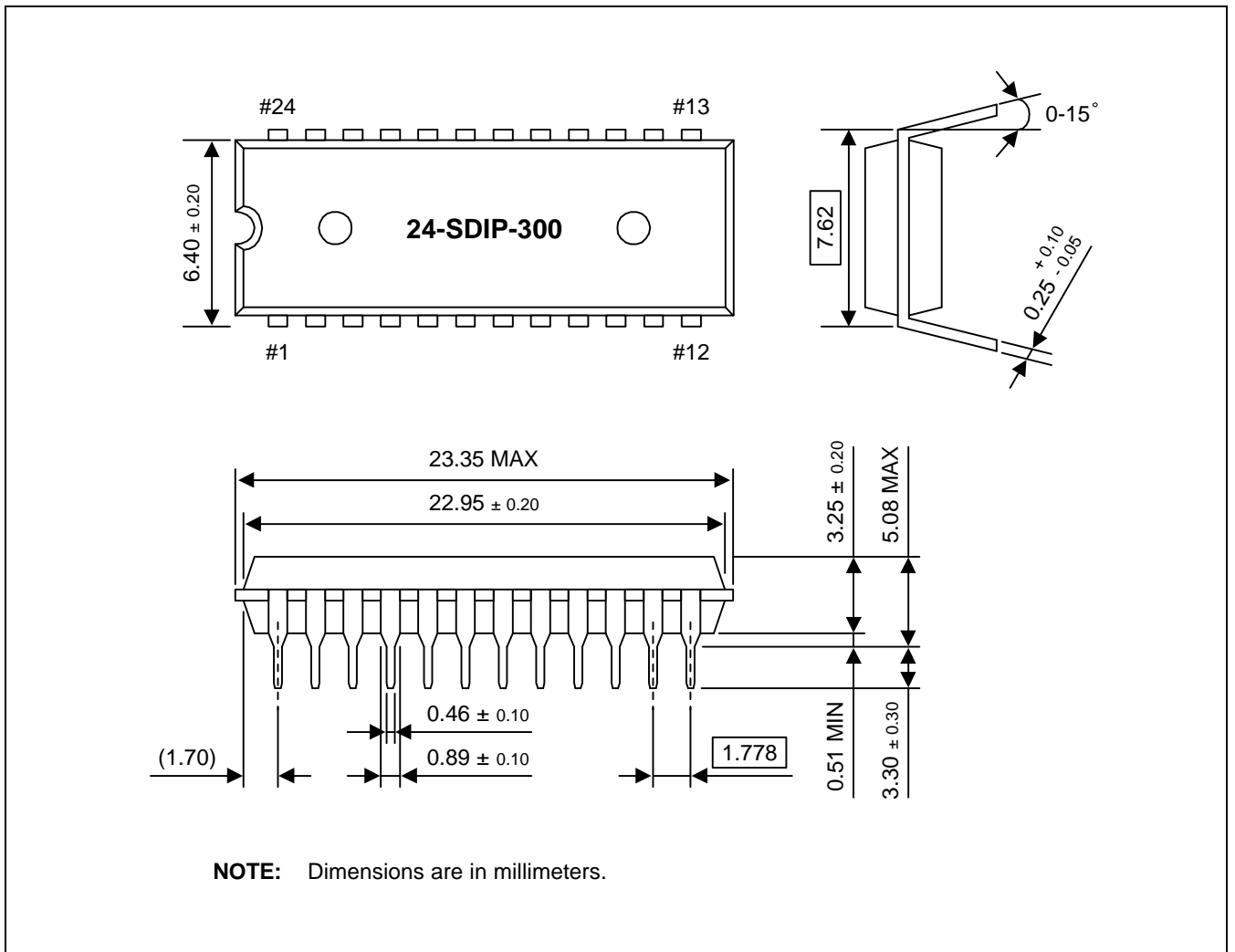


Figure 17-3. 24-SDIP-300 Package Dimensions

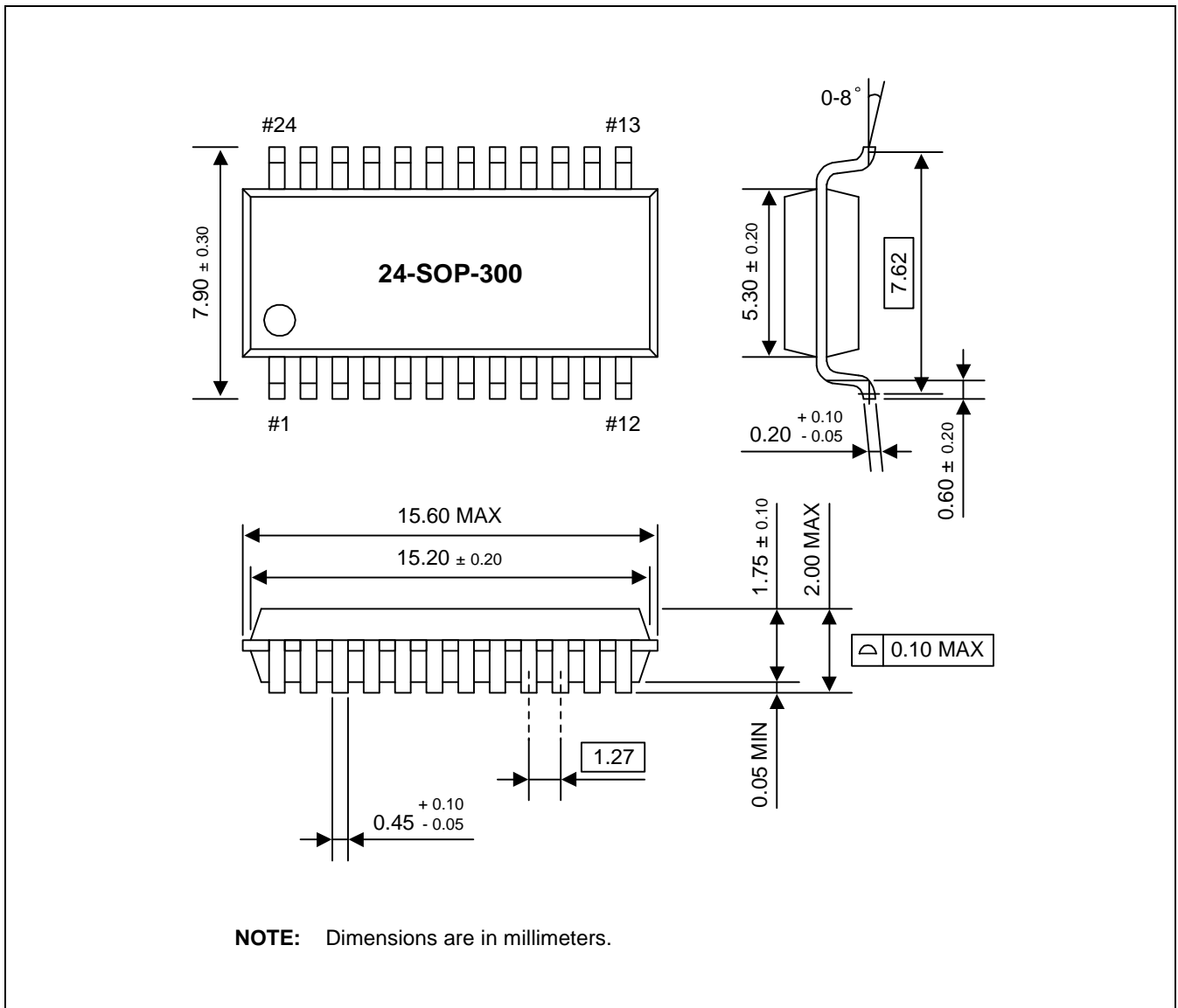


Figure 17-4. 24-SOP-300 Package Dimensions

18

S3P9664 OTP

OVERVIEW

The S3C9664 single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the S3C9664 microcontroller. It has an on-chip OTP ROM instead of masked ROM. The EPROM is accessed by serial data format.

The S3P9664 is fully compatible with the S3C9664, both in function and in pin configuration. Because of its simple programming requirements, the S3P9664 is ideal for use as an evaluation chip for the S3C9664.

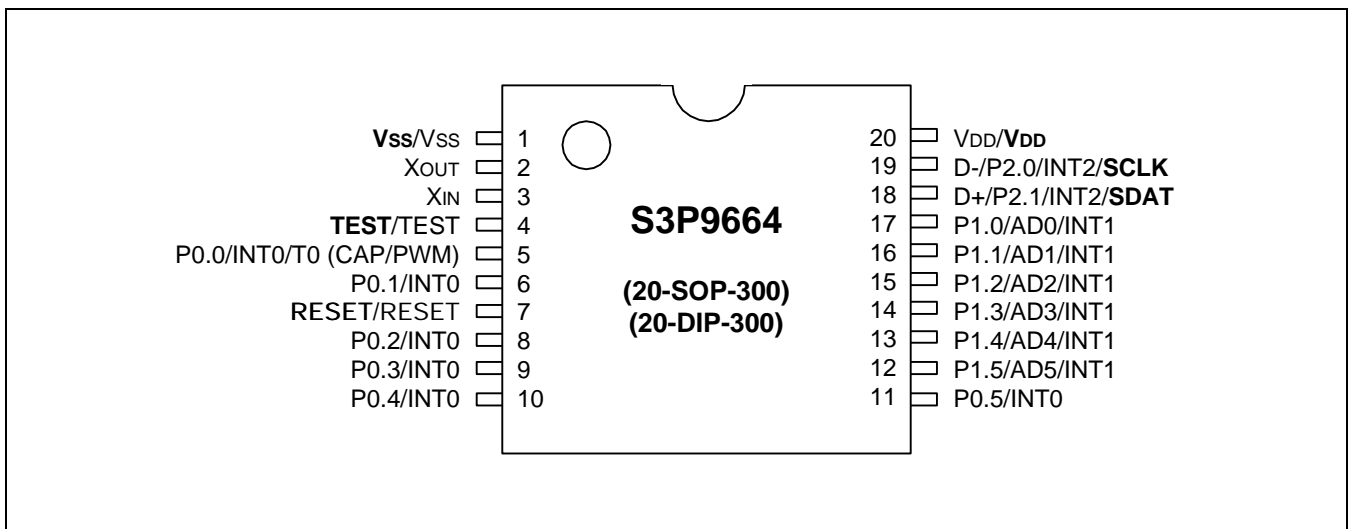


Figure 18-1. S3P9664 Pin Assignments (20-Pin Package)

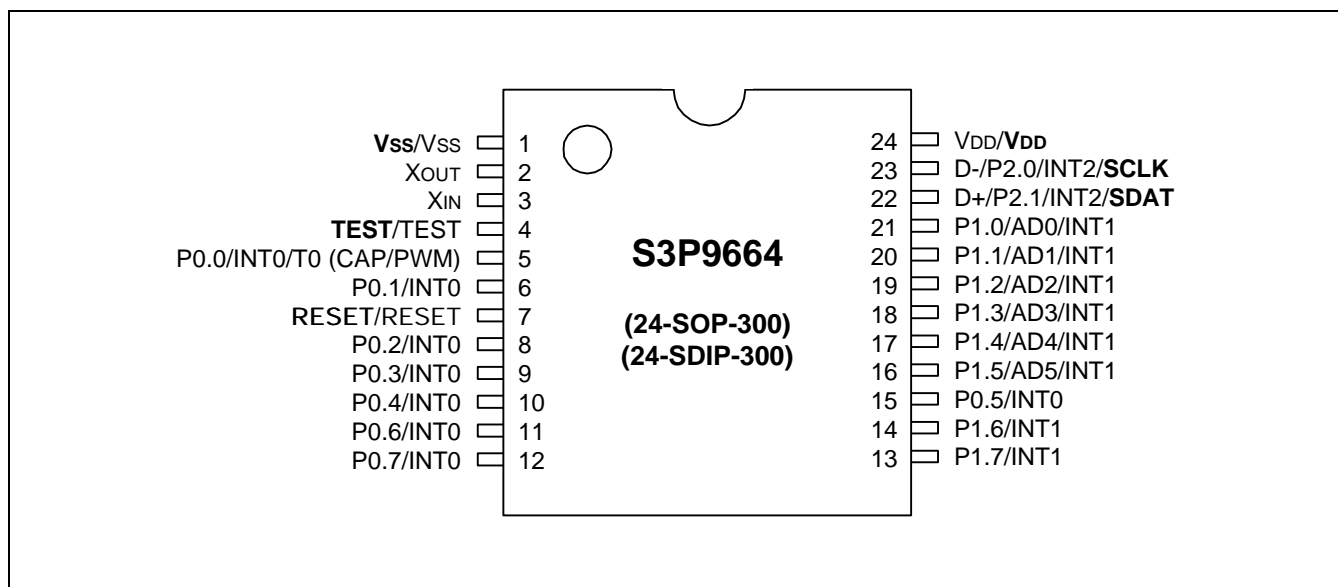


Figure 18-2. S3P9664 Pin Assignments (24-Pin Package)

Table 18-1. Descriptions of Pins Used to Read/Write the EPROM

Main Chip Pin Name	During Programming			
	Pin Name	Pin No. (24 DIP)	I/O	Function
P1.0	SDAT	23	I/O	Serial Data Pin (Output when reading, Input when writing) Input and Push-pull Output Port can be assigned
P1.1	SCLK	22	I/O	Serial Clock Pin (Input Only Pin)
TEST	V_{PP} (TEST)	5	I	0V : OTP write and test mode 5V : Operating mode
RESET	RESET	8	I	Chip Initialization and EPROM Cell Writing Power Supply Pin (Indicates OTP Mode Entering) When writing 12.5V is applied and when reading.
V_{DD}/V_{SS}	V_{DD}/V_{SS}	24/1	I	Logic Power Supply Pin.

Table 18-2. Comparison of S3P9664 and S3C9664 Features

Characteristic	S3P9664	S3C9664
Program Memory	4 K byte EPROM	4 K byte mask ROM
Operating Voltage (V_{DD})	4.0 V to 5.25 V	4.0 V to 5.25 V
OTP Programming Mode	$V_{DD} = 5 V$, V_{PP} (RESET) =12.5V	
Pin Configuration	20 SOP/20 DIP/24 SOP/24 SDIP	20 SOP/20 DIP/24 SOP/24SDIP
EPROM Programmability	User Program 1 time	Programmed at the factory

19

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.

SASM86

The SASM86 is an relocatable assembler for Samsung's S3C9-series microcontrollers. The SASM86 takes a source file containing assembly language statements and translates into a corresponding source code, object code and comments. The SASM86 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 S3C9-series microcontrollers. All required target system cables and adapters are included with the device-specific target board.

OTPs

One time programmable microcontrollers (OTPs) are under development for S3C9664/P9664 microcontroller.

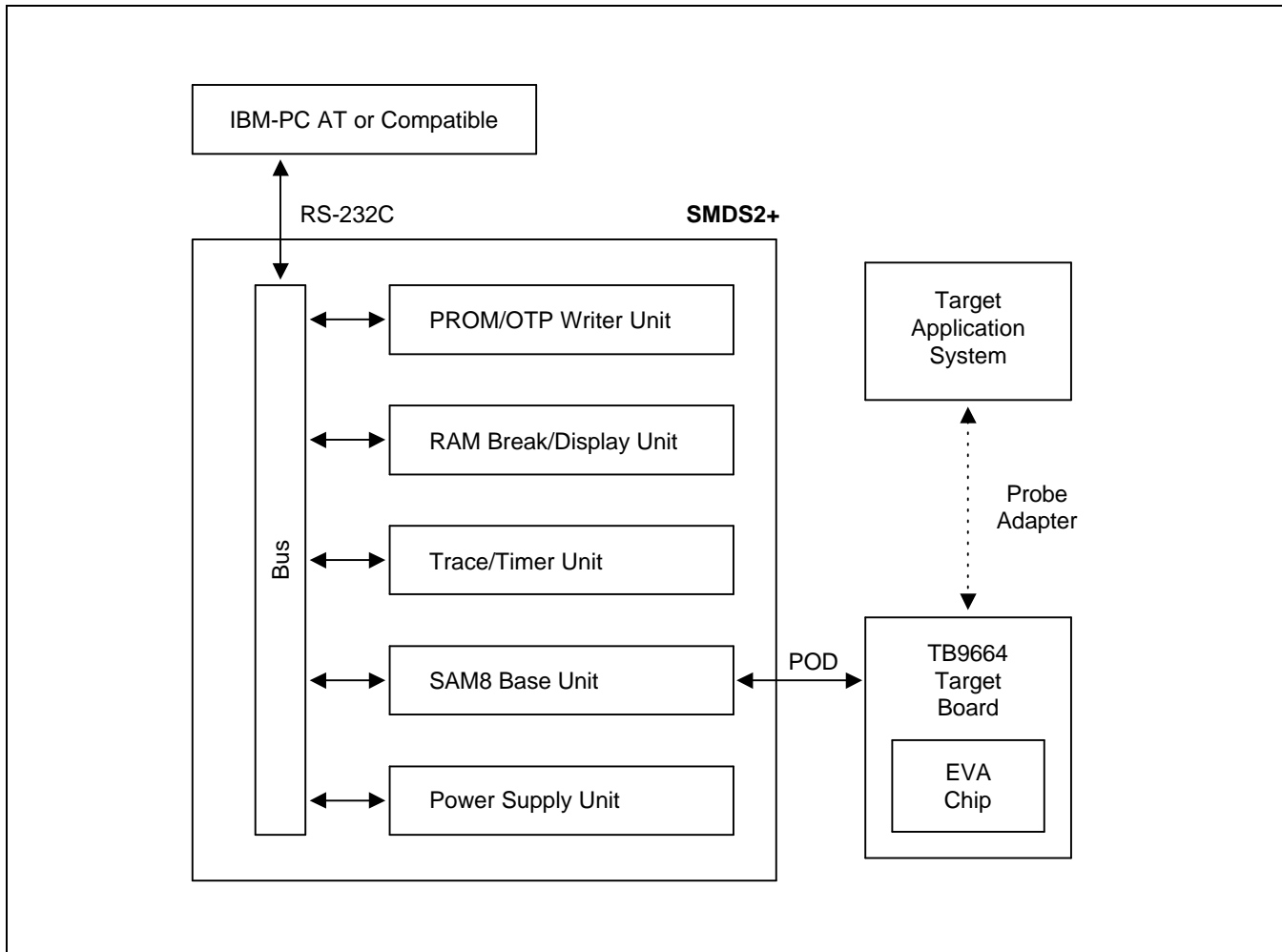


Figure 19-1. SMDS Product Configuration (SMDS2+)

TB9664 TARGET BOARD

The TB9664 target board is used for the S3C9664/P9664 microcontrollers. It is supported by the SMDS2+ development system.

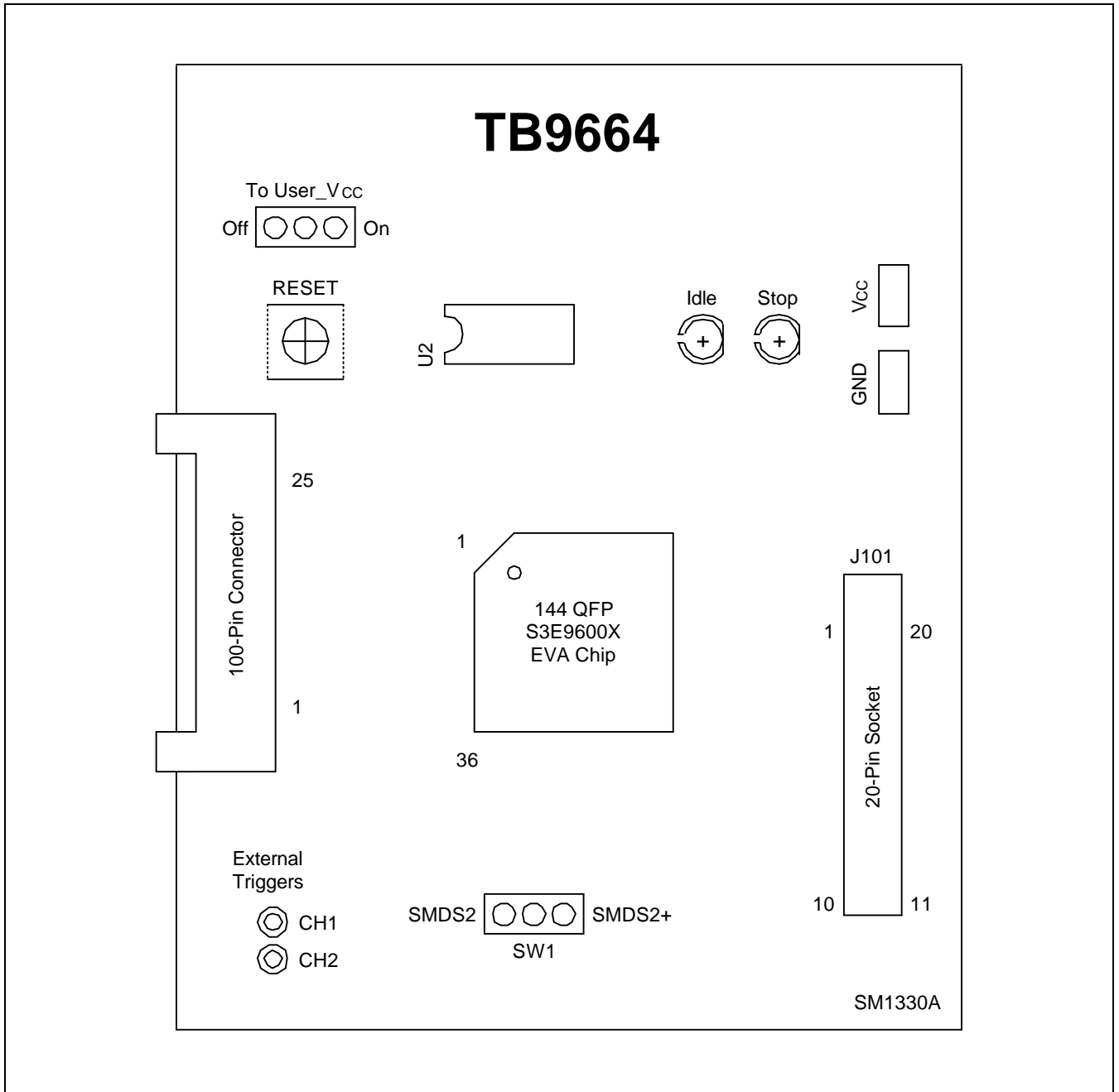
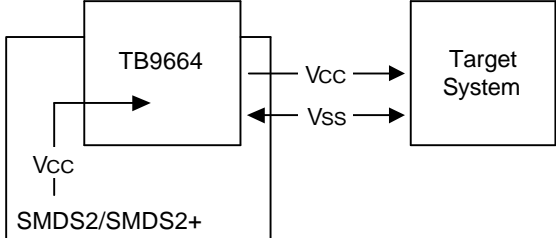
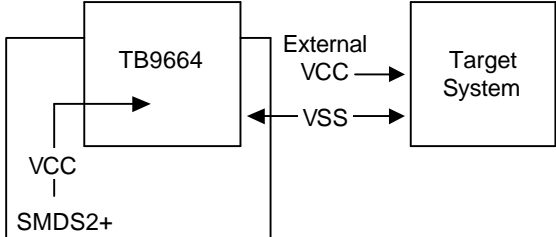


Figure 19-2. TB9664 Target Board Configuration

Table 19-1. Power Selection Settings for TB9664

'To User_Vcc' Settings	Operating Mode	Comments
To User_Vcc Off <input type="radio"/> <input checked="" type="radio"/> <input checked="" type="radio"/> On		SMDS2/SMDS2+ supplies V _{CC} to the target board (evaluation chip) and the target system.
To User_Vcc Off <input checked="" type="radio"/> <input checked="" type="radio"/> <input type="radio"/> On		SMDS2/SMDS2+ supplies V _{CC} only to the target board (evaluation chip). The target system must have a power supply of its own.

SMDS2+ Selection (SAM8)

In order to write data into program memory available in SMDS2+, the target board should be selected for SMDS2+ through a switch as follows. Otherwise, the program memory writing function is not available.

Table 19-2. The SMDS2+ Tool Selection Setting

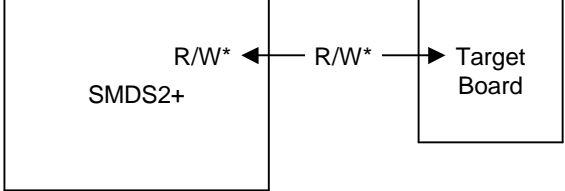
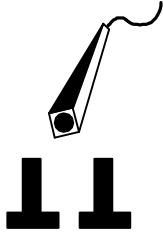
'SW1' Setting	Operating Mode
SMDS2 <input type="radio"/> <input checked="" type="radio"/> <input checked="" type="radio"/> SMDS2+	

Table 19-3. Using Single Header Pins as the Input Path for External Trigger Sources

Target Board Part	Comments
<p>External Triggers</p> <p>○ CH1</p> <p>○ CH2</p>	<div style="text-align: center;">  </div> <p>Connector from External Trigger Sources of the Application System</p> <p>You can connect an external trigger source to one of the two external trigger channels (CH1 or CH2) for the SMDS2+ breakpoint and trace functions.</p>

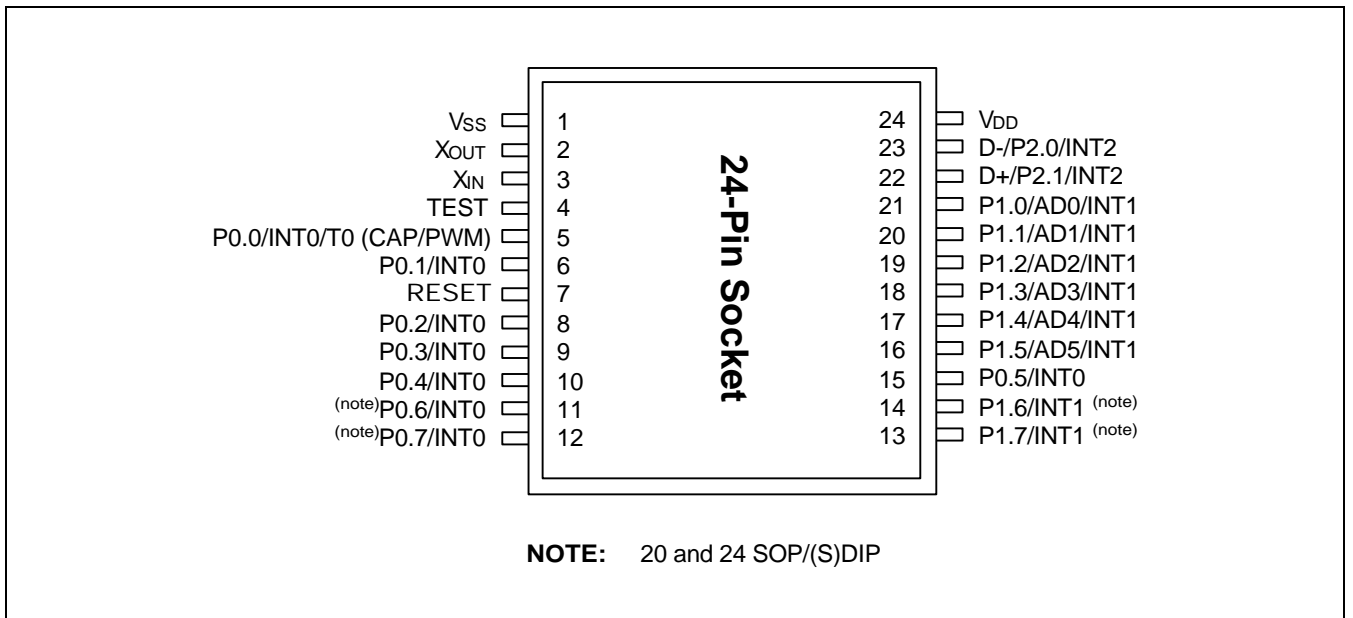


Figure 19-3. 24-Pin Socket for TB9664

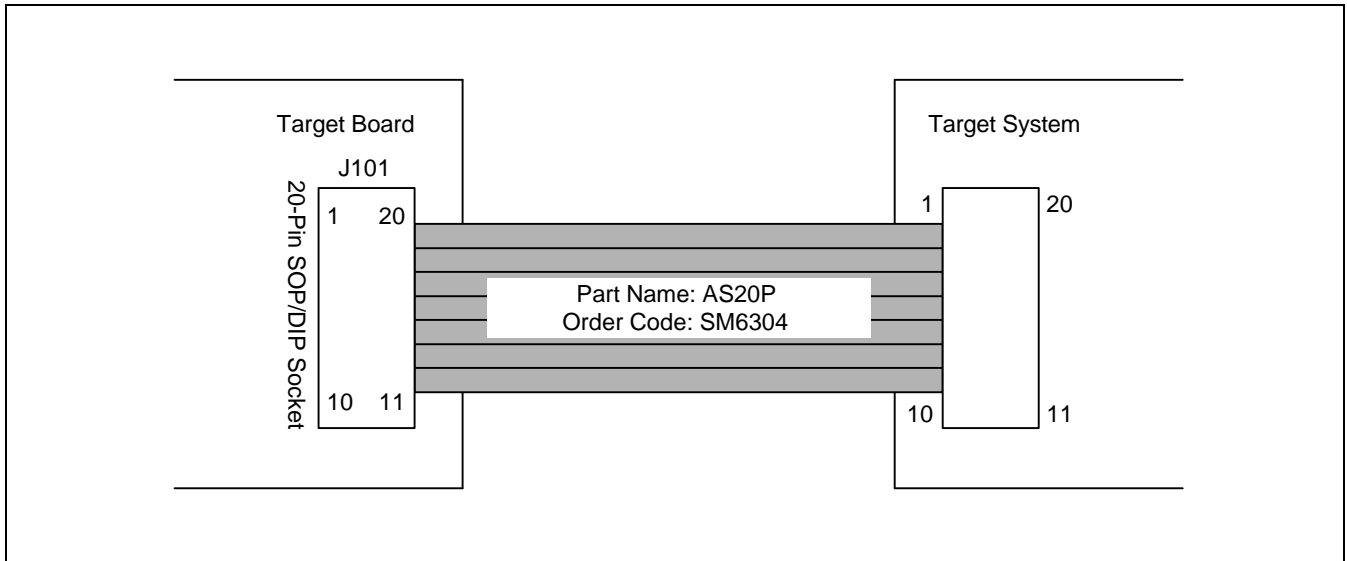


Figure 19-4. TB9664 Adapter Cable for 20-SOP/DIP Package

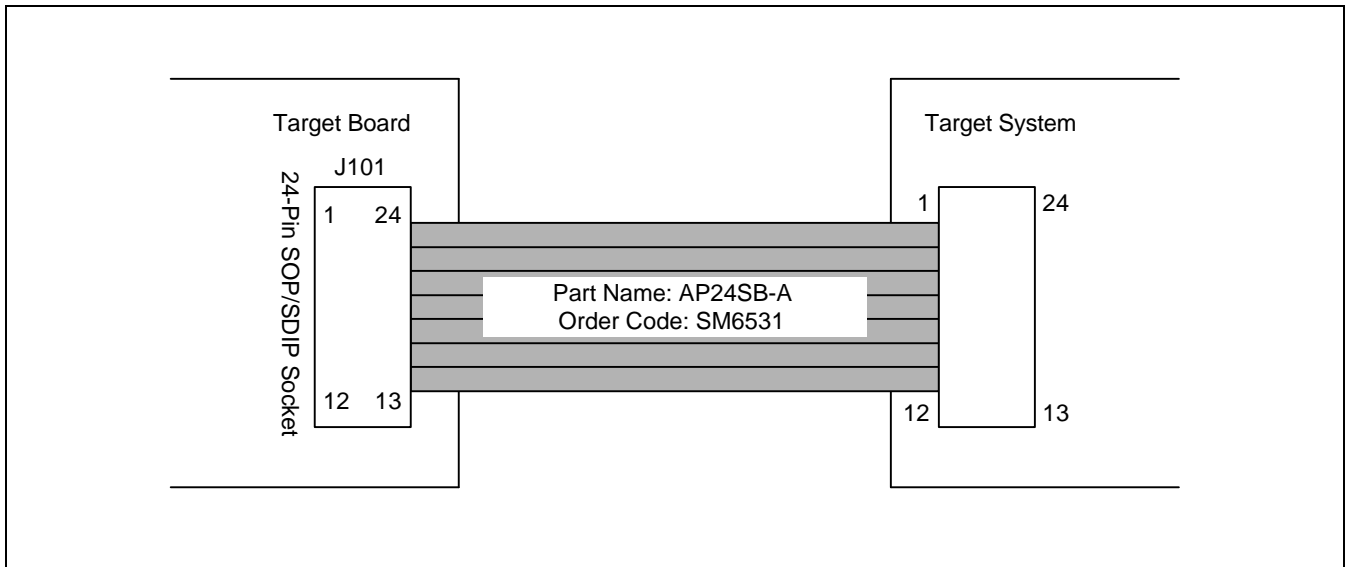


Figure 19-5. TB9664 Adapter Cable for 24-SOP/SDIP Package