21-S3-C9228/P9228-112002

USER'S MANUAL

S3C9228/P9228
8-Bit CMOS Microcontroller

Revision 1

SAMSUNG ELECTRONICS

S3C9228/P9228

8-BIT CMOS MICROCONTROLLERS USER'S MANUAL

Revision 1

SAMSUNG ELECTRONICS

Important Notice

The information in this publication has been carefully checked and is believed to be entirely accurate at the time of publication. Samsung assumes no responsibility, however, for possible errors or omissions, or for any consequences resulting from the use of the information contained herein.

Samsung reserves the right to make changes in its products or product specifications with the intent to improve function or design at any time and without notice and is not required to update this documentation to reflect such changes.

This publication does not convey to a purchaser of semiconductor devices described herein any license under the patent rights of Samsung or others.

Samsung makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Samsung assume any liability arising out of the application or use of any product or circuit and specifically disclaims any and all liability, including without limitation any consequential or incidental damages.

"Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by the customer's technical experts.

Samsung products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, for other applications intended to support or sustain life, or for any other application in which the failure of the Samsung product could create a situation where personal injury or death may occur.

Should the Buyer purchase or use a Samsung product for any such unintended or unauthorized application, the Buyer shall indemnify and hold Samsung and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, either directly or indirectly, any claim of personal injury or death that may be associated with such unintended or unauthorized use, even if such claim alleges that Samsung was negligent regarding the design or manufacture of said product.

S3C9228/P9228 8-Bit CMOS Microcontrollers
User's Manual, Revision 1
Publication Number: 21-S3-C9228/P9228-112002

© 2002 Samsung Electronics

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electric or mechanical, by photocopying, recording, or otherwise, without the prior written consent of Samsung Electronics.

Samsung Electronics' microcontroller business has been awarded full ISO-14001 certification (BVQI Certificate No 9330) All semiconductor products are designed and manufactured in accordance with the highest quality standards and objectives.

Samsung Electronics Co., Ltd.
San #24 Nongseo-Ri, Kiheung-Eup
Yongin-City, Kyunggi-Do, Korea
C.P.O. Box #37, Suwon 449-900
TEL: (82)-(331)-209-1907
FAX: (82)-(331)-209-1889
Home-Page URL: Http://www.samsungsemi.com
Printed in the Republic of Korea

Preface

The *S3C9228/P9228 Microcontrollers User's Manual* is designed for application designers and programmers who are using the S3C9228/P9228 microcontrollers for application development. It is organized in two main parts:

Part I Programming Model Part II Hardware Descriptions

Part I contains software-related information to familiarize you with the microcontroller's architecture, programming model, instruction set, and interrupt structure. It has six chapters:

Chapter 1	Product Overview	Chapter 4	Control Registers
Chapter 2	Address Spaces	Chapter 5	Interrupt Structure
Chapter 3	Addressing Modes	Chapter 6	SAM88RCRI Instruction Set

Chapter 1, "Product Overview," is a high-level introduction to the 100% with general product descriptions, as well as detailed information about individual pin characteristics and pin circuit types.

Chapter 2, "Address Spaces," explains the 100% program and data memory, internal register file, and mapped control register, and explains how to address them. Chapter 2 also describes working register addressing, as well as system stack and user-defined stack operations.

Chapter 3, "Addressing Modes," contains detailed descriptions of the addressing modes that are supported by the CPU.

Chapter 4, "Control Registers," contains overview tables for all mapped system and peripheral control register values, as well as detailed one-page descriptions in standard format. You can use these easy-to-read, alphabetically organized, register descriptions as a quick-reference source when writing programs.

Chapter 5, "Interrupt Structure," describes the 100% interrupt structure in detail and further prepares you for additional information presented in the individual hardware module descriptions in Part II.

Chapter 6, "SAM88RCRI Instruction Set," describes the features and conventions of the instruction set used for all S3C9-series microcontrollers. Several summary tables are presented for orientation and reference. Detailed descriptions of each instruction are presented in a standard format. Each instruction description includes one or more practical examples of how to use the instruction when writing an application program.

A basic familiarity with the information in Part I will help you to understand the hardware module descriptions in Part II. If you are not yet familiar with the SAM8 product family and are reading this manual for the first time, we recommend that you first read chapters 1–3 carefully. Then, briefly look over the detailed information in chapters 4, 5, and 6. Later, you can reference the information in Part I as necessary.

Part II "hardware Descriptions," has detailed information about specific hardware components of the S3C9228/P9228 microcontrollers. Also included in Part II are electrical, mechanical, OTP, and development tools data. It has 13 chapters:

Chapter 7	Clock Circuits	Chapter 14	10-bit ADC
Chapter 8	RESET and Power-Down	Chapter 15	Serial I/O Interface
Chapter 9	I/O Ports	Chapter 16	Electrical Data
Chapter 10	Basic Timer	Chapter 17	Mechanical Data
Chapter 11	Timer 1	Chapter 18	S3P9228 OTP
Chapter 12	Watch Timer	Chapter 19	Development Tools
Chapter 13	LCD Controller/Driver		

Two order forms are included at the back of this manual to facilitate customer order for S3C9228/P9228 microcontrollers: the Mask ROM Order Form, and the Mask Option Selection Form. You can photocopy these forms, fill them out, and then forward them to your local Samsung Sales Representative.

Table of Contents

Part I — Programming Model

Chapter 1 Product Overview

SAM88RCRI Product Family 1-1
S3C9228/P9228 Microcontroller 1-1
OTP 1-1
Features 1-2
Block Diagram 1-3
Pin Assignments 1-4
Pin Descriptions 1-6
Pin Circuit Diagrams 1-8

Chapter 2 Address Spaces

Overview 2-1
Program Memory (ROM) 2-2
Register Architecture 2-3
Common Working Register Area (C0H–CFH) 2-4
System Stack 2-5

Chapter 3 Addressing Modes

Overview 3-1
- Register Addressing Mode (R) 3-2
- Indirect Register Addressing Mode (IR) 3-3
- Indexed Addressing Mode (X) 3-7
- Direct Address Mode (DA) 3-10
- Relative Address Mode (RA) 3-12
- Immediate Mode (IM) 3-12

Table of Contents (Continued)

Chapter 4 Control Registers

Overview 4-1

Chapter 5 Interrupt Structure

Overview 5-1
Interrupt Processing Control Points 5-1
Enable/Disable Interrupt Instructions (EI, DI) 5-1
Interrupt Pending Function Types 5-2
Interrupt Priority 5-2
Interrupt Source Service Sequence 5-3
Interrupt Service Routines 5-3
Generating Interrupt Vector addresses 5-3
S3C9228/P9228 Interrupt Structure 5-4

Chapter 6 SAM88RCRI Instruction Set

Overview 6-1
Register Addressing 6-1
Addressing Modes 6-1
Flags Register (FLAGS) 6-4
Flag Descriptions 6-4
Instruction Set Notation 6-5
Condition Codes 6-9
Instruction Descriptions 6-10

Table of Contents (Continued)

Part II — Hardware Descriptions

Chapter 7 Clock Circuit

Overview 7-1
- System Clock Circuit 7-1
- CPU Clock Notation 7-1
- Main Oscillator Circuits 7-2
- Sub Oscillator Circuits 7-2
- Clock Status During Power-Down Modes 7-3
- System Clock Control Register (CLKCON) 7-4
- Oscillator Control Register (OSCCON) 7-5
- Switching The CPU Clock 7-6
- Stop Control Register (STPCON) 7-7

Chapter 8 RESET and Power-Down

System Reset 8-1
- Overview 8-1

Power-Down Modes 8-2
- Stop Mode 8-2
- Idle Mode 8-3
- Hardware Reset Values 8-4

Chapter 9 I/O Ports

Overview 9-1
- Port Data Registers 9-2
- Port 0 9-3
- Port 1 9-6
- Port 2 9-9
- Port 3 9-11
- Port 4 9-14
- Port 5 9-15
- Port 6 9-16

Chapter 10 Basic Timer

Overview 10-1
- Basic Timer Control Register (BTCON) 10-2
- Basic Timer Function Description 10-3

Table of Contents (Continued)

Chapter 11 Timer 1

One 16-Bit Timer Mode (Timer 1) 11-1
- Overview 11-1
- Function Description 11-1

Two 8-Bit Timers Mode (Timer A and B) 11-4
- Overview 11-4
- Function Description 11-7

Chapter 12 Watch Timer

Overview 12-1
- Watch Timer Control Register (WTCON) 12-2

Watch Timer Circuit Diagram 12-3

Chapter 13 LCD Controller/Driver

Overview 13-1
- LCD Circuit Diagram 13-2
- LCD RAM Address Area 13-3
- LCD Mode Control Register (LMOD) 13-4
- LCD Port Control Register 13-5
- LCD Voltage Dividing Resistors 13-6
- Common (COM) Signals 13-6
- Segment (SEG) Signals 13-6

Chapter 14 10-Bit A/D Converter

Overview 14-1
Function Description 14-1
- Conversion Timing 14-2
- A/D Converter Control Register (ADCON) 14-2
- Internal Reference Voltage Levels 14-3

Block Diagram 14-3

Chapter 15 Serial I/O Interface

Overview 15-1
- Programming Procedure 15-1
- SIO Control Registers (SIOCON) 15-2
- SIO Pre-Scaler Register (SIOPS) 15-3

SIO Block Diagram 15-3
- Serial I/O Timing Diagram (SIO) 15-4

Table of Contents (Concluded)

Chapter 16 Electrical Data

Overview 16-1

Chapter 17 Mechanical Data

Overview 17-1

Chapter 18 S3P9228 OTP

Overview 18-1
Operating Mode Characteristics 18-3

Chapter 19 Development Tools

Overview 19-1
SHINE 19-1
SAMA Assembler 19-1
SASM86 19-1
HEX2ROM 19-1
Target Boards 19-1
TB9228 Target Board 19-3
Idle LED 19-5
Stop LED 19-5

List of Figures

Figure Number	Title	Page Number
1-1	Block Diagram	1-3
1-2	S3C9228 44-QFP Pin Assignments	1-4
1-3	S3C9228 42-SDIP Pin Assignments	1-5
1-4	Pin Circuit Type B	1-8
1-5	Pin Circuit Type C	1-8
1-7	Pin Circuit Type E-4	1-9
1-8	Pin Circuit Type F-16A	1-9
1-9	Pin Circuit Type H-23	1-10
1-10	Pin Circuit Type H-32	1-11
1-11	Pin Circuit Type H-32A	1-11
1-12	Pin Circuit Type H-32B	1-12
2-1	S3C9228/P9228 Program Memory Address Space	2-2
2-2	Internal Register File Organization	2-3
2-3	16-Bit Register Pairs	2-4
2-4	Stack Operations	2-5
3-1	Register Addressing	3-2
3-2	Working Register Addressing	3-2
3-3	Indirect Register Addressing to Register File	3-3
3-4	Indirect Register Addressing to Program Memory	3-4
3-5	Indirect Working Register Addressing to Register File	3-5
3-6	Indirect Working Register Addressing to Program or Data Memory	3-6
3-7	Indexed Addressing to Register File	3-7
3-8	Indexed Addressing to Program or Data Memory with Short Offset	3-8
3-9	Indexed Addressing to Program or Data Memory with Long Offset	3-9
3-10	Direct Addressing for Load Instructions	3-10
3-11	Direct Addressing for Call and Jump Instructions	3-11
3-12	Relative Addressing	3-12
3-13	Immediate Addressing	3-12
4-1	Register Description Format	4-4

List of Figures (Continued)

Figure Number	Title	Page Number
5-1	S3C9-Series Interrupt Type	5-1
5-2	Interrupt Function Diagram	5-2
5-3	S3C9228/P9228 Interrupt Structure	5-5
6-1	System Flags Register (FLAGS)	6-4
7-1	Crystal/Ceramic Oscillator	7-2
7-2	External Oscillator	7-2
7-3	RC Oscillator	7-2
7-4	Crystal/Ceramic Oscillator	7-2
7-5	External Oscillator	7-2
7-6	System Clock Circuit Diagram	7-3
7-7	System Clock Control Register (CLKCON)	7-4
7-8	Oscillator Control Register (OSCCON)	7-5
7-9	STOP Control Register (STPCON)	7-7
9-1	S3C9228 I/O Port Data Register Format	9-2
9-2	Port 0 Control Register (P0CON)	9-4
9-3	Port 0 Interrupt Control Register (P0INT)	9-4
9-4	Port 0 Interrupt Pending Bits (INTPND1.3-.0)	9-5
9-5	Port 0 Interrupt Edge Selection Register (P0EDGE)	9-5
9-6	Port 0 Pull-up Control Register (P0PUR)	9-5
9-7	Port 1 Control Register (P1CON)	9-6
9-8	Port 1 Interrupt Control Register (P1INT)	9-7
9-9	Port 1 Interrupt Pending Bits (INTPND1.7-.4)	9-7
9-10	Port 1 Interrupt Edge Selection Register (P1EDGE)	9-8
9-11	Port 1 Pull-up Control Register (P1PUR)	9-8
9-12	Port 2 Control Register (P2CON)	9-9
9-13	Port 2 Pull-up Control Register (P2PUR)	9-10
9-14	Port 3 Control Register (P3CON)	9-11
9-15	Port 3 Interrupt Control Register (P3INT)	9-12
9-16	Port 3 Interrupt Pending Bits (INTPND2.5-.4)	9-12
9-17	Port 3 Interrupt Edge Selection Register (P3EDGE)	9-13
9-18	Port 3 Pull-up Control Register (P3PUR)	9-13
9-19	Port 4 High-Byte Control Register (P4CONH)	9-14
9-20	Port 4 Low-Byte Control Register (P4CONL)	9-14
9-21	Port 5 High-Byte Control Register (P5CONH)	9-15
9-22	Port 5 Low-Byte Control Register (P5CONL)	9-15
9-23	Port 6 Control Register (P6CON)	9-16

List of Figures (Continued)

Figure Number	Title	Page Number
10-1	Basic Timer Control Register (BTCON)	10-2
10-2	Basic Timer Block Diagram	10-4
11-1	Timer 1 Control Register (TACON)	11-2
11-2	Timer 1 Block Diagram (One 16-bit Mode)	11-3
11-3	Timer A Control Register (TACON)	11-5
11-4	Timer B Control Register (TBCON)	11-6
11-5	Timer A Block Diagram (Two 8-bit Timers Mode)	11-8
11-6	Timer B Block Diagram (Two 8-bit Timers Mode)	11-9
12-1	Watch Timer Control Register (WTCON)	12-2
12-2	Watch Timer Circuit Diagram	12-3
13-1	LCD Function Diagram	13-1
13-2	LCD Circuit Diagram	13-2
13-3	LCD Display Data RAM Organization	13-3
13-4	LCD Mode Control Register (LMOD)	13-4
13-5	LCD Port Control Register	13-5
13-6	Internal Voltage Dividing Resistor Connection	13-6
13-7	LCD Signal Waveforms (1/8 Duty, 1/4 Bias)	13-7
13-8	LCD Signal Waveforms (1/4 Duty, 1/3 Bias)	13-8
13-9	LCD Signal Waveforms (1/3 Duty, 1/3 Bias)	13-9
14-1	A/D Converter Control Register (ADCON)	14-2
14-2	A/D Converter Data Register (ADDATAH/ADDATAL)	14-3
14-3	A/D Converter Functional Block Diagram	14-3
14-4	Recommended A/D Converter Circuit for Highest Absolute Accuracy	14-4
15-1	Serial I/O Module Control Register (SIOCON)	15-2
15-2	SIO Prescaler Register (SIOPS)	15-3
15-3	SIO Functional Block Diagram	15-3
15-4	Serial I/O Timing in Transmit/Receive Mode (Tx at falling, SIOCON.4 = 0)	15-4
15-5	Serial I/O Timing in Transmit/Receive Mode (Tx at rising, SIOCON.4 = 1)	15-4

List of Figures (Concluded)

Figure Number	Title	Page Number
16-1	Stop Mode Release Timing When Initiated by an External Interrupt	16-5
16-2	Stop Mode Release Timing When Initiated by a RESET	16-6
16-3	Input Timing for External Interrupts	16-8
16-4	Input Timing for RESET	16-9
16-5	Serial Data Transfer Timing	16-9
16-6	Clock Timing Measurement at X_{IN}	16-11
16-7	Clock Timing Measurement at XT_{IN}	16-12
16-8	Operating Voltage Range	16-13
17-1	42-SDIP-600 Package Dimensions	17-1
17-2	44-QFP-1010B Package Dimensions	17-2
18-1	S3P9228 44-QFP Pin Assignments	18-1
18-2	S3P9228 42-SDIP Pin Assignments	18-2
18-3	Standard Operating Voltage Range	18-5
19-1	SMDS Product Configuration (SMDS2+)	19-2
19-2	TB9228 Target Board Configuration	19-3
19-3	Connectors (J101, J102) for TB9228	19-6
19-4	S3C9228 Probe Adapter for 42-SDIP Package	19-7
19-5	S3C9228 Probe Adapter for 44-QFP Package	19-7

List of Tables

Table Number	Title	Page Number
1-1	Pin Descriptions	1-6
6-1	Instruction Group Summary	6-2
6-2	Flag Notation Conventions	6-5
6-3	Instruction Set Symbols	6-5
6-4	Instruction Notation Conventions	6-6
6-5	Opcode Quick Reference	6-7
6-6	Condition Codes	6-9
8-1	Register Values after RESET	8-4
9-1	S3C9228 Port Configuration Overview	9-1
9-2	Port Data Register Summary	9-2
13-1	Common and Segment Pins per Duty Cycle	13-3
16-1	Absolute Maximum Ratings	16-2
16-2	D.C. Electrical Characteristics	16-2
16-3	Data Retention Supply Voltage in Stop Mode	16-5
16-4	Input/Output Capacitance	16-6
16-5	A.C. Electrical Characteristics	16-7
16-6	A/D Converter Electrical Characteristics	16-8
16-7	Main Oscillation Characteristics	16-10
16-8	Sub Oscillation Characteristics	16-10
16-9	Main Oscillation Stabilization Time	16-11
16-10	Sub Oscillation Stabilization Time	16-12
18-1	Descriptions of Pins Used to Read/Write the EPROM	18-3
18-2	Comparison of S3P9228 and S3C9228 Features	18-3
18-3	Operating Mode Selection Criteria	18-3
18-4	D.C. Electrical Characteristics	18-4
19-1	Power Selection Settings for TB9228	19-4
19-2	The SMDS2+ Tool Selection Setting	19-5
19-3	Using Single Header Pins as the Input Path for External Trigger Sources	19-5

List of Programming Tips

Description **Page Number**

Chapter 2: Address Spaces

Addressing the Common Working Register Area 2-4
Standard Stack Operations Using PUSH and POP 2-6

Chapter 5: Interrupt Structure

How to clear an interrupt pending bit..... 5-6

Chapter 7: Clock Circuits

Switching the CPU clock..... 7-6
How to Use Stop Instruction..... 7-7

List of Register Descriptions

Register Identifier	Full Register Name	Page Number
ADCON	A/D Converter Control Register	4-5
BTCON	Basic Timer Control Register	4-6
CLKCON	System Clock Control Register	4-7
FLAGS	System Flags Register	4-8
INTPND1	Interrupt Pending Register 1	4-9
INTPND2	Interrupt Pending Register 2	4-10
LMOD	LCD Mode Control Register	4-11
LPOT	LCD Port Control Register	4-12
OSSCON	Oscillator Control Register	4-13
P0CON	Port 0 Control Register	4-14
P0INT	Port 0 Interrupt Enable Register	4-15
P0PUR	Port 0 Pull-up Resistors Enable Register	4-16
P0EDGE	Port 0 Interrupt Edge Selection Register	4-17
P1CON	Port 1 Control Register	4-18
P1INT	Port 1 Interrupt Enable Register	4-19
P1PUR	Port 1 Pull-up Resistors Enable Register	4-20
P1EDGE	Port 1 Interrupt Edge Selection Register	4-21
P2CON	Port 2 Control Register	4-22
P2PUR	Port 2 Pull-up Resistors Enable Register	4-23
P3CON	Port 3 Control Register	4-24
P3INT	Port 2 Interrupt Enable Register	4-25
P3PUR	Port 3 Pull-up Resistors Enable Register	4-26
P3EDGE	Port 3 Interrupt Edge Selection Register	4-27
P4CONH	Port 4 Control Register	4-28
P4CONL	Port 4 Control Register Low Byte	4-29
P5CONH	Port 5 Control Register High Byte	4-30
P5CONL	Port 5 Control Register Low Byte	4-31
P6CON	Port 6 Control Register	4-32
SIOCON	SIO Control Register	4-33
STPCON	Stop Control Register	4-34
SYM	System Mode Register	4-35
TACON	Timer 1/A Control Register	4-36
TBCON	Timer B Control Register	4-37
WTCON	Watch Timer Control Register	4-38

List of Instruction Descriptions

Instruction Mnemonic	Full Instruction Name	Page Number
ADC	Add With Carry	6-11
ADD	Add	6-12
AND	Logical AND	6-13
CALL	Call Procedure	6-14
CCF	Complement Carry Flag	6-15
CLR	Clear	6-16
COM	Complement	6-17
CP	Compare	6-18
DEC	Decrement	6-19
DI	Disable Interrupts	6-20
EI	Enable Interrupts	6-21
IDLE	Idle Operation	6-22
INC	Increment	6-23
IRET	Interrupt Return	6-24
JP	Jump	6-25
JR	Jump Relative	6-26
LD	Load	6-27
LDC/LDE	Load Memory	6-29
LDCD/LDED	Load Memory and Decrement	6-31
LDCI/LDEI	Load Memory and Increment	6-32
NOP	No Operation	6-33
OR	Logical OR	6-34
POP	Pop From Stack	6-35
PUSH	Push To Stack	6-36
RCF	Reset Carry Flag	6-37
RET	Return	6-38
RL	Rotate Left	6-39
RLC	Rotate Left Through Carry	6-40
RR	Rotate Right	6-41
RRC	Rotate Right Through Carry	6-42
SBC	Subtract With Carry	6-43
SCF	Set Carry Flag	6-44
SRA	Shift Right Arithmetic	6-45
STOP	Stop Operation	6-46
SUB	Subtract	6-47
TCM	Test Complement Under Mask	6-48
TM	Test Under Mask	6-49
XOR	Logical Exclusive OR	6-50

1 PRODUCT OVERVIEW

SAM88RCRI 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 supports 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.

S3C9228/P9228 MICROCONTROLLER

The S3C9228 can be used for dedicated control functions in a variety of applications, and is especially designed for application with FRS or etc.

The S3C9228/P9228 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 S3C9228/P9228 has 8K-byte of program ROM, and 264-byte of RAM (including 16-byte of working register and 20-byte LCD display RAM).

Using the SAM88RCRI design approach, the following peripherals were integrated with the SAM88RCRI core:

— 7 configurable I/O ports including ports shared with segment/common drive outputs
— 10-bit programmable pins for external interrupts
— One 8-bit basic timer for oscillation stabilization and watch-dog functions
— Two 8-bit timer/counters with selectable operating modes
— Watch timer for real time
— 4 channel A/D converter
— 8-bit serial I/O interface

OTP

The S3C9228 microcontroller is also available in OTP (One Time Programmable) version. S3P9228 microcontroller has an on-chip 8K-byte one-time-programmable EPROM instead of masked ROM. The S3P9228 is comparable to S3C9228, both in function and in pin configuration.

FEATURES

CPU

- SAM88RCRI CPU core

Memory

- 8192 × 8 bits program memory (ROM)
- 264 × 8 bits data memory (RAM)
  (Including LCD data memory)

Instruction Set

- 41 instructions
- Idle and Stop instructions added for power-down modes

36 I/O Pins

- I/O: 34 pins (44-pin QFP, 42-pin SDIP)
- Output only: 2 pins (44-pin QFP)

Interrupts

- 14 interrupt source and 1 vector
- One interrupt level

8-Bit Basic Timer

- Watchdog timer function
- 3 kinds of clock source

Two 8-Bit Timer/Counters

- The programmable 8-bit timer/counters
- External event counter function
- Configurable as one 16-bit timer/counters

Watch Timer

- Interval time: 3.91mS, 0.25S, 0.5S, and 1S at 32.768 kHz
- 0.5/1/2/4 kHz Selectable buzzer output
- Clock source generation for LCD

LCD Controller/Driver

- 16 segments and 8 common terminals
- 3, 4, and 8 common selectable
- Internal resistor circuit for LCD bias

8-bit Serial I/O Interface

- 8-bit transmit/receive mode
- 8-bit receive mode
- LSB-first or MSB-first transmission selectable
- Internal or external clock source

A/D Converter

- 10-bit converter resolution
- 50us conversion speed at 1MHz f_{ADC} clock
- 4-channel

Two Power-Down Modes

- Idle: only CPU clock stops
- Stop: system clock and CPU clock stop

Oscillation Sources

- Crystal, ceramic, or RC for main clock
- Main clock frequency: 0.4 MHz - 8MHz
- 32.768 kHz crystal oscillation circuit for sub clock

Instruction Execution Times

- 500nS at 8MHz fx(minimum)

Operating Voltage Range

- 2.0 V to 5.5 V at 0.4 - 4.2MHz
- 2.7 V to 5.5 V at 0.4 - 8MHz

Operating Temperature Range

- -25 °C to +85 °C

Package Type

- 44-pin QFP, 42-pin SDIP

BLOCK DIAGRAM

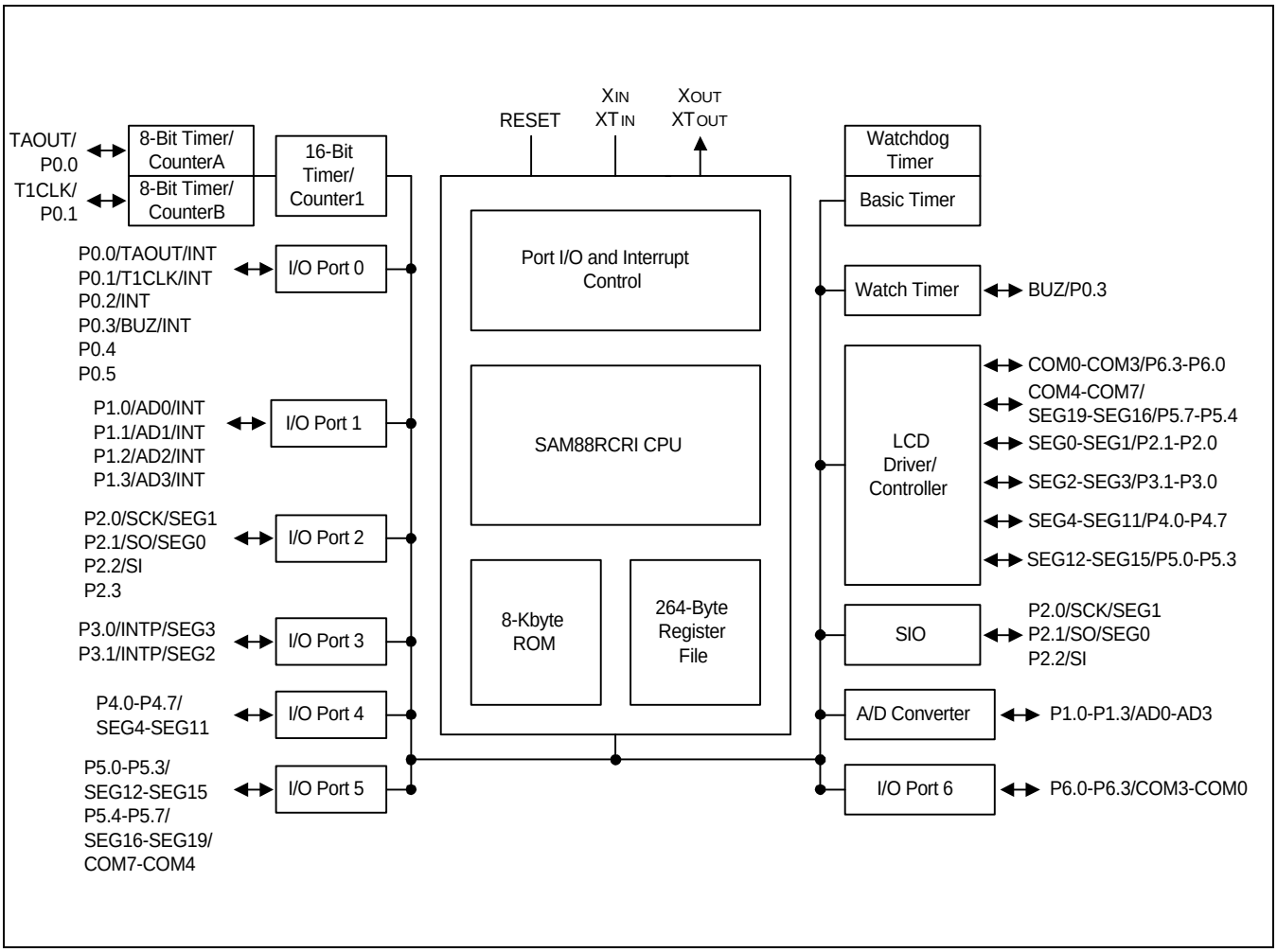


Figure 1-1. Block Diagram

PIN ASSIGNMENTS

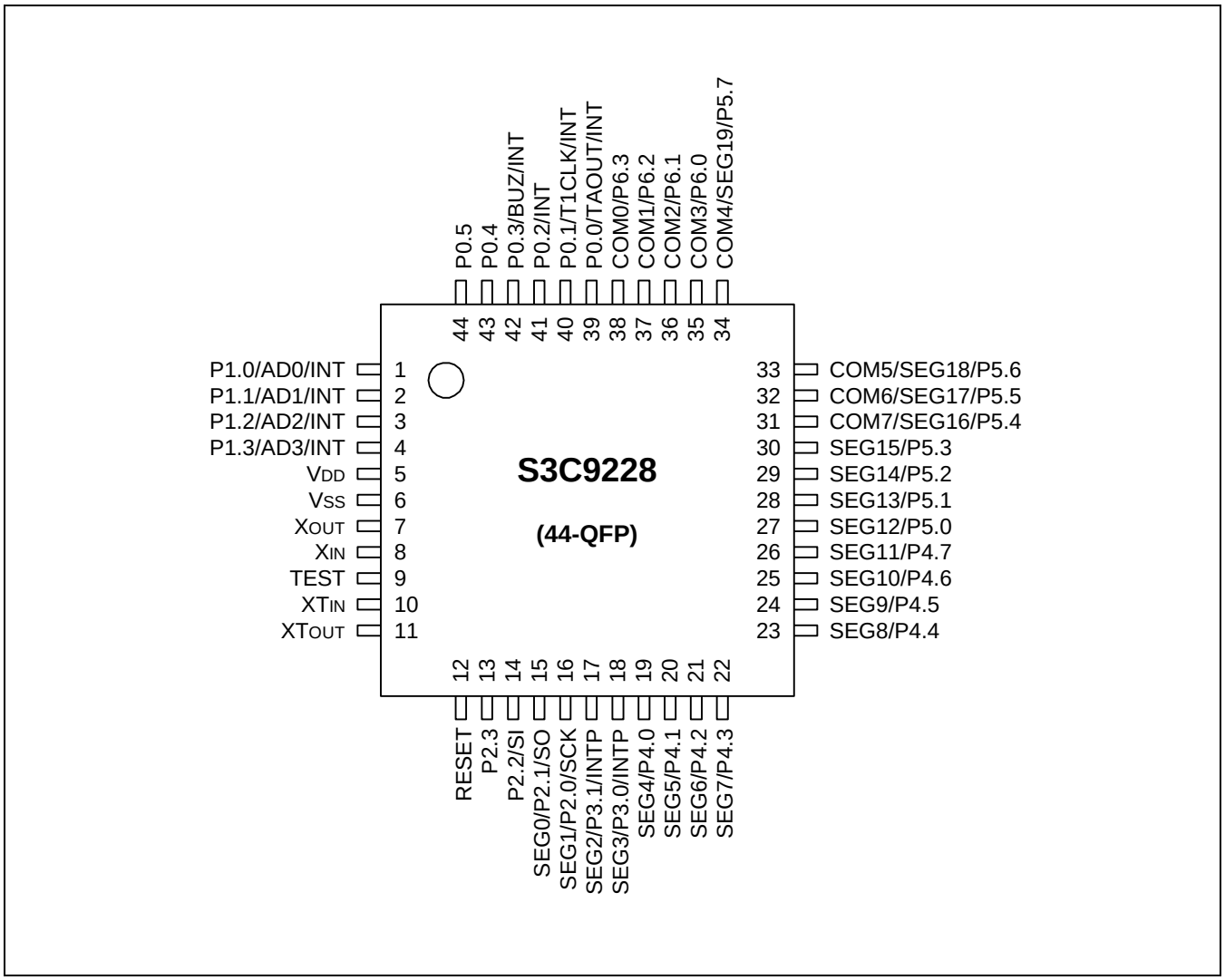


Figure 1-2. S3C9228 44-QFP Pin Assignments

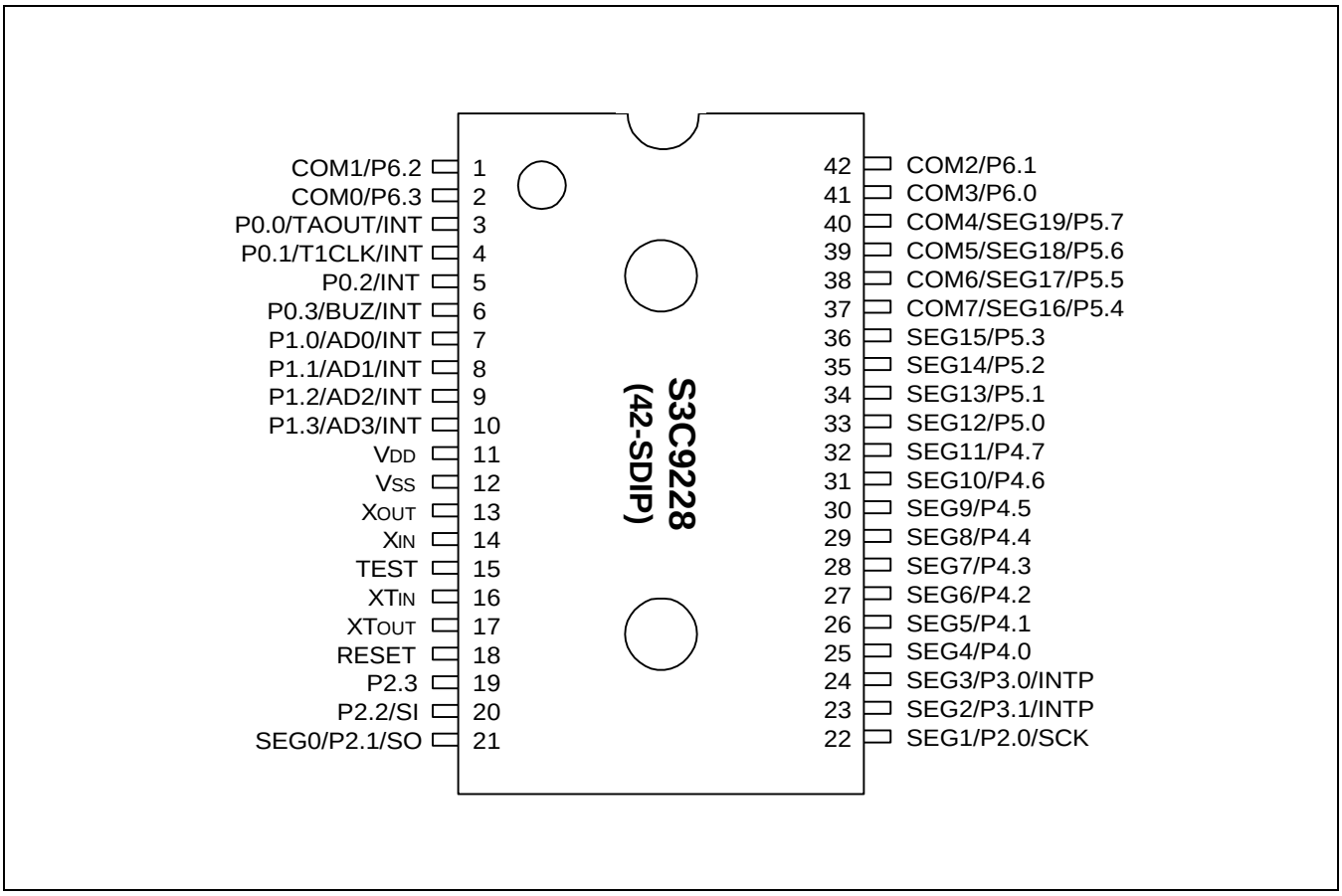


Figure 1-3. S3C9228 42-SDIP Pin Assignments

PIN DESCRIPTIONS

Table 1-1. Pin Descriptions

Pin Names	Pin Type	Pin Description	Circuit Number	Pin Numbers	Share Pins
P0.0 P0.1 P0.2 P0.3	I/O	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	E-4	39(3) 40(4) 41(5) 42(6)	TAOUT/INT T1CLK/INT INT BUZ/INT
P0.4-P0.5	O	1-bit programmable output port.	C	43-44	
P1.0 P1.1 P1.2 P1.3	I/O	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	F-16A	1(7) 2(8) 3(9) 4(10)	AD0/INT AD1/INT AD2/INT AD3/INT
P2.0 P2.1	I/O	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain	H-32A	16(22) 15(21)	SCK/SEG1 SO/SEG0
P2.2 P2.3	I/O	output and software assignable pull-ups.	E-4	14(20) 13(19)	SI –
P3.0 P3.1	I/O	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	H-32B	18(24) 17(23)	INTP/SEG3 INTP/SEG2
P4.0–P4.7	I/O	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups.	H-32	19-26(25-32)	SEG4-SEG11
P5.0–P5.3	I/O	1-bit programmable I/O port. Input or push-pull, open-drain output and	H-32	27-30(33-36)	SEG12-SEG15
P5.4–P5.7		software assignable pull-ups.		31-34(37-40)	SEG16-SEG19 /COM7-COM4
P6.0-P6.3	I/O	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups.	H-32	35-38 (41-42,1-2)	COM3-COM0

NOTE: Parentheses indicate pin number for 42-SDIP-600 package.

Table 1-1. Pin Descriptions (Continued)

Pin Names	Pin Type	Pin Description	Circuit Number	Pin Numbers	Share Pins
V_{DD}, V_{SS}	–	Power input pins for internal power block	–	5,6(11,12)	–
X_{OUT}, X_{IN}	–	Main oscillator pins for main clock	–	7,8(13,14)	
XT_{OUT}, XT_{IN}	–	Sub oscillator pins for sub clock	–	11,10(17,16)	–
TEST	–	Chip test input pin Hold GND when the device is operating	–	9(15)	–
RESET	I	RESET signal input pin. Schmitt trigger input with internal pull-up resistor.	B	12(18)	–
INT	I/O	External interrupts input.	E-4 F-16A	39-42(3-6) 1-4(7-10)	P0.0-P0.3 P1.0-P1.3
INTP	I/O	Key scan interrupts inputs.	H-32B	17-18(23-24)	P3.1-P3.0
T1CLK	I/O	Timer 1/A external clock input.	E-4	40(4)	P0.1
TAOUT	I/O	Timer 1/A clock output.	E-4	39(3)	P0.0
AD0-AD3	I/O	Analog input pins for A/D converts module.	F-16A	1-4(7-10)	P1.0-P1.3
BUZ	I/O	Buzzer signal output.	E-4	42(6)	P0.3
SCK SO	I/O	Serial clock, serial data output, serial data input	H-32A	16-15(22-21)	P2.0-P2.1
SI			E-4	14(20)	P2.2
SEG0-SEG1	I/O	LCD segment signal output	H-32A	15-16(21-22)	P2.1-P2.0
SEG2-SEG3			H-32B	17-18(23-24)	P3.1-P3.0
SEG4-SEG19			H-32	19-34(25-40)	P4.0-P4.7 P5.0-P5.7
COM0-COM7	I/O	LCD common signal output	H-32	38-31 (2-1,42-37)	P6.3-P6.0 P5.7-P5.4

NOTE: Parentheses indicate pin number for 42-SDIP-600 package.

PIN CIRCUIT DIAGRAMS

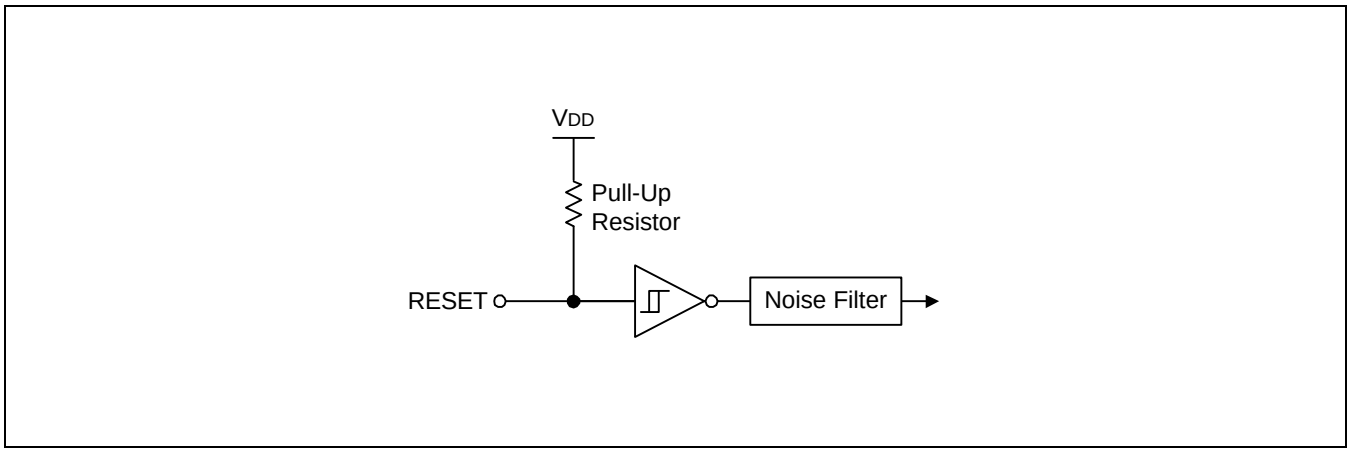


Figure 1-4. Pin Circuit Type B

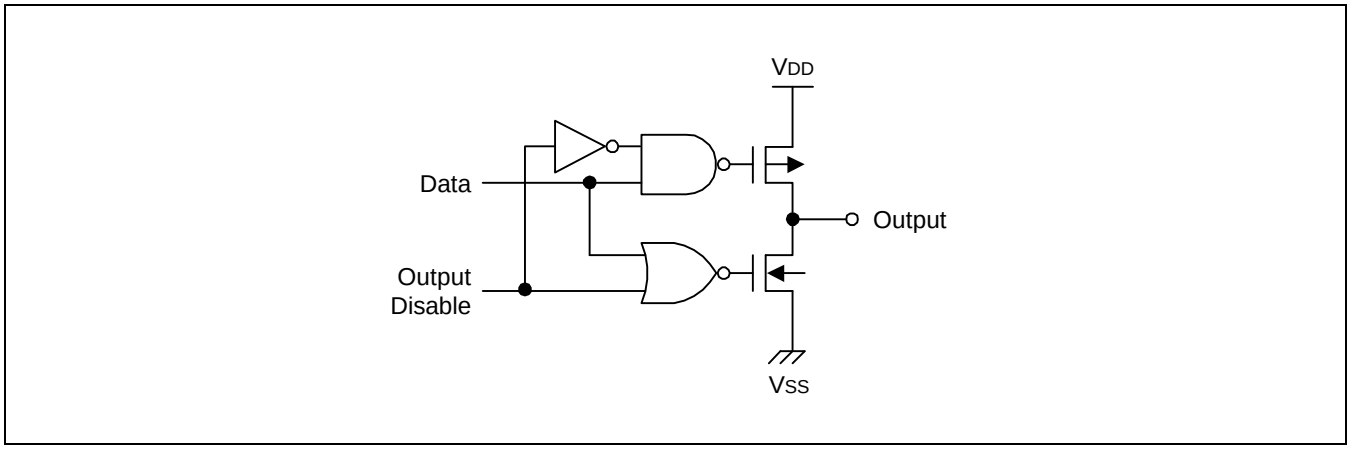


Figure 1-5. Pin Circuit Type C

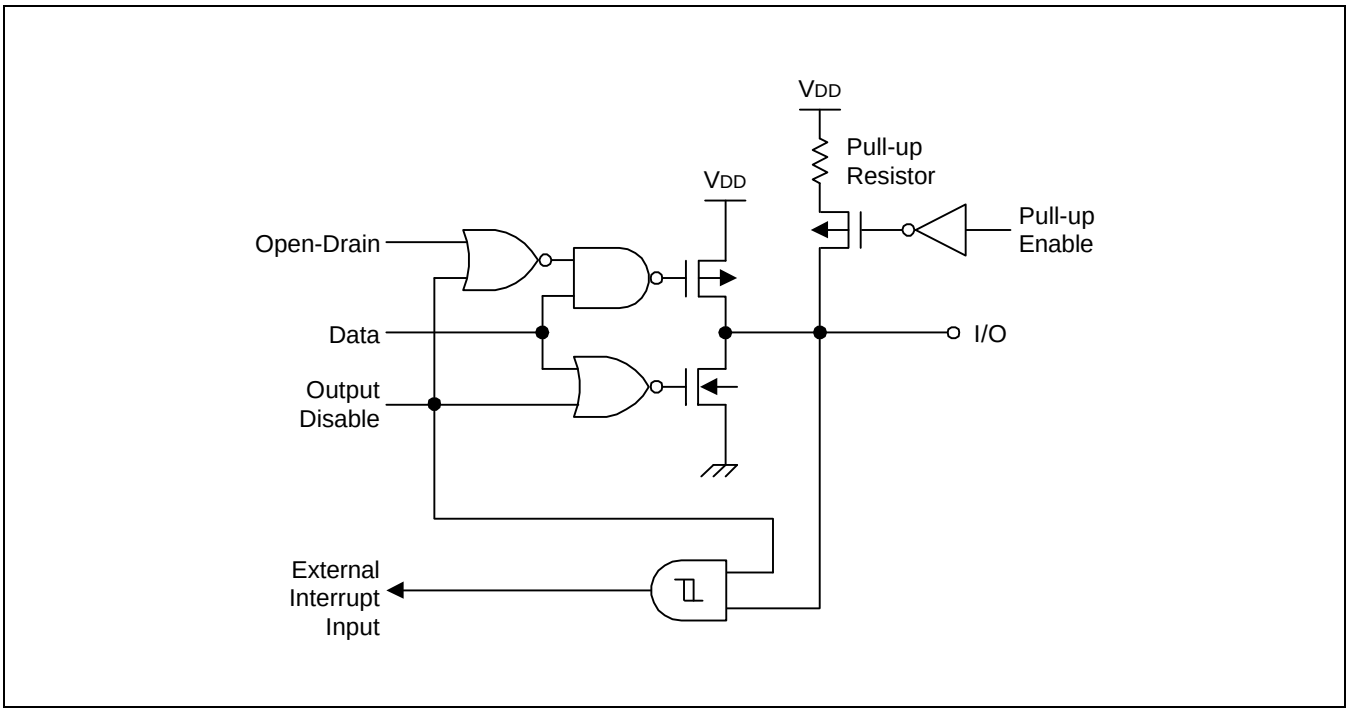


Figure 1-7. Pin Circuit Type E-4

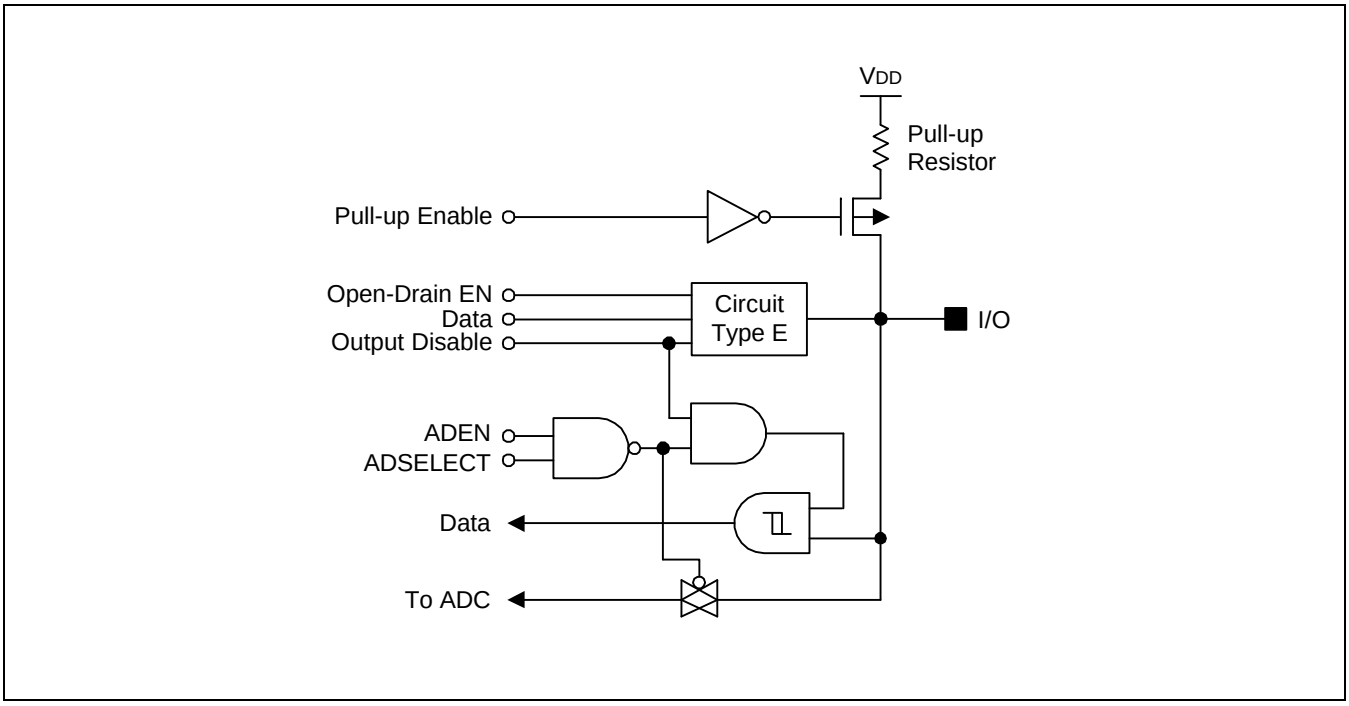


Figure 1-8. Pin Circuit Type F-16A

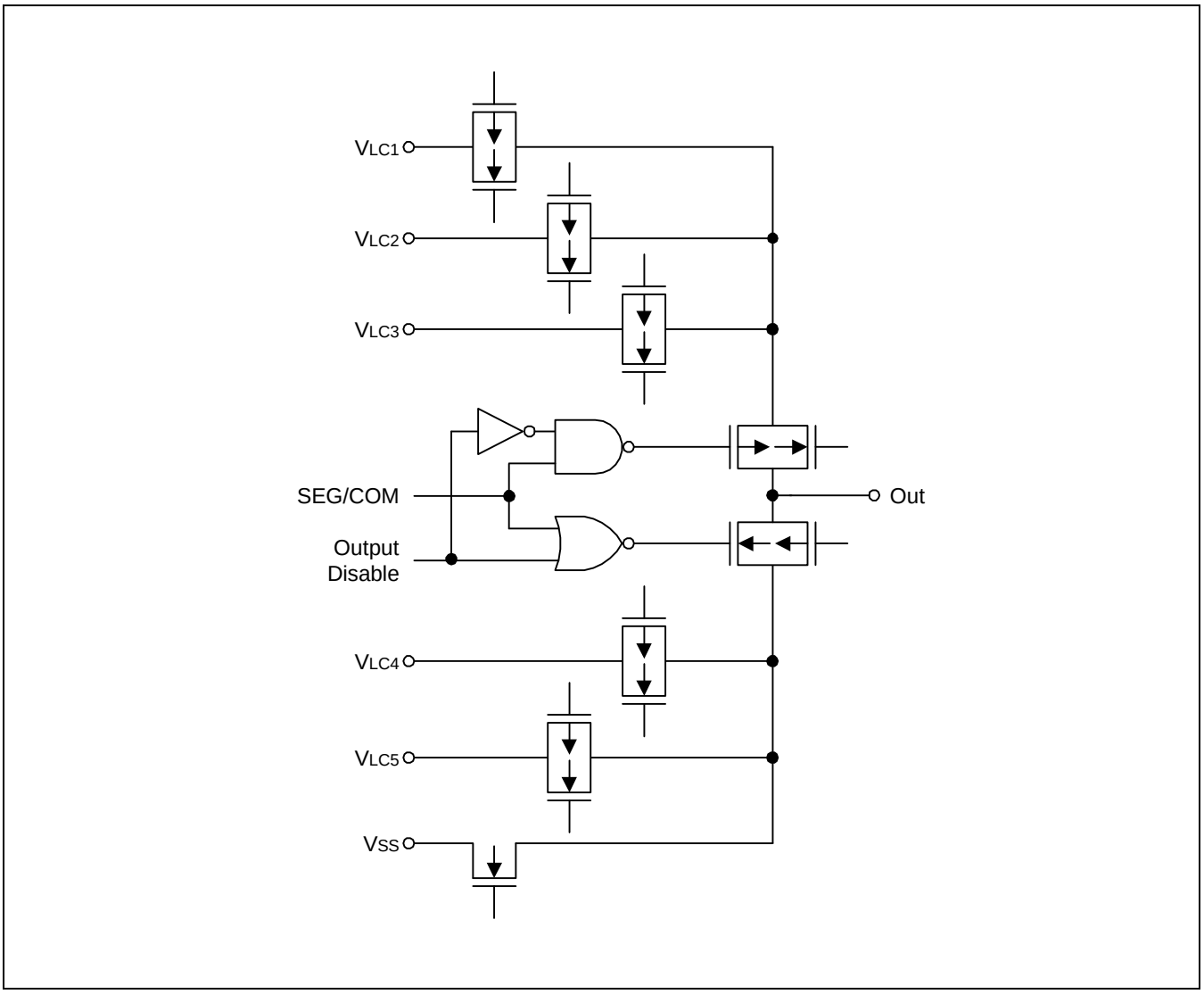


Figure 1-9. Pin Circuit Type H-23

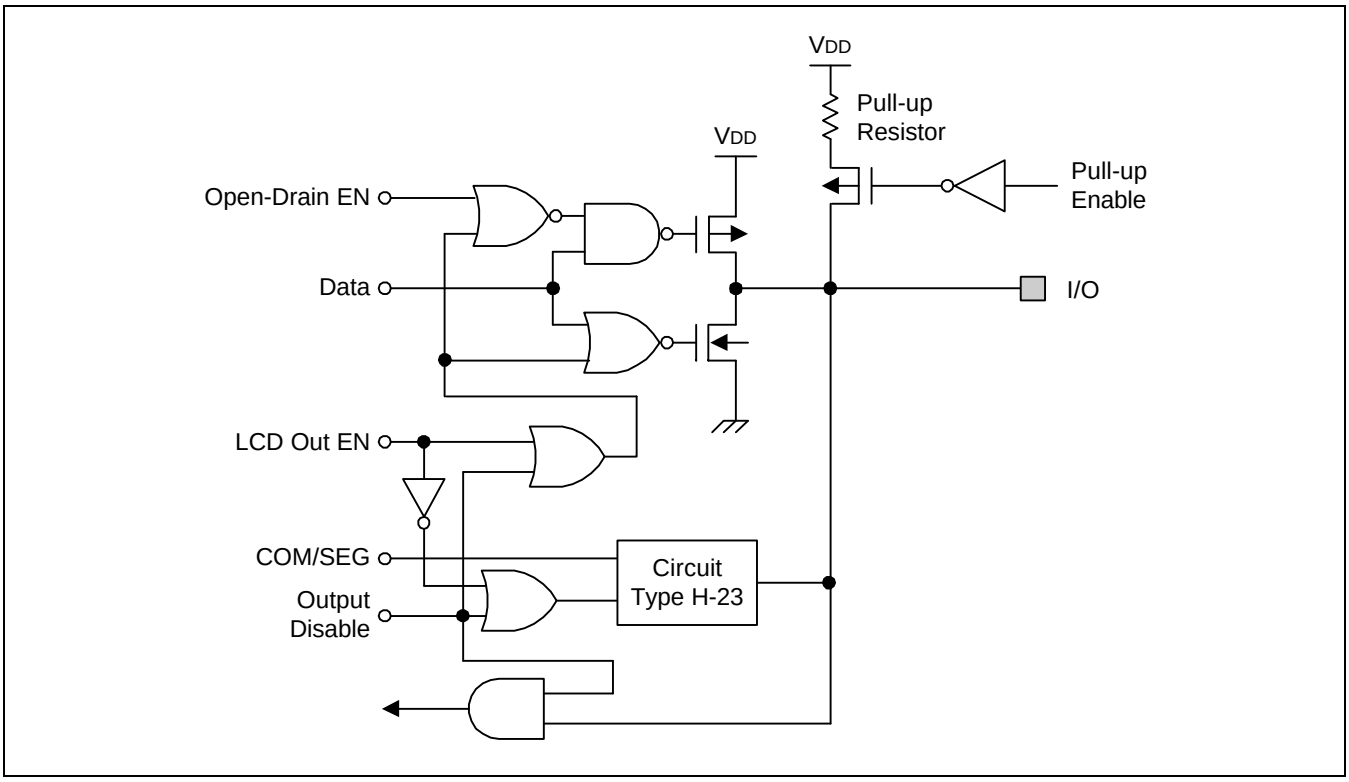


Figure 1-10. Pin Circuit Type H-32

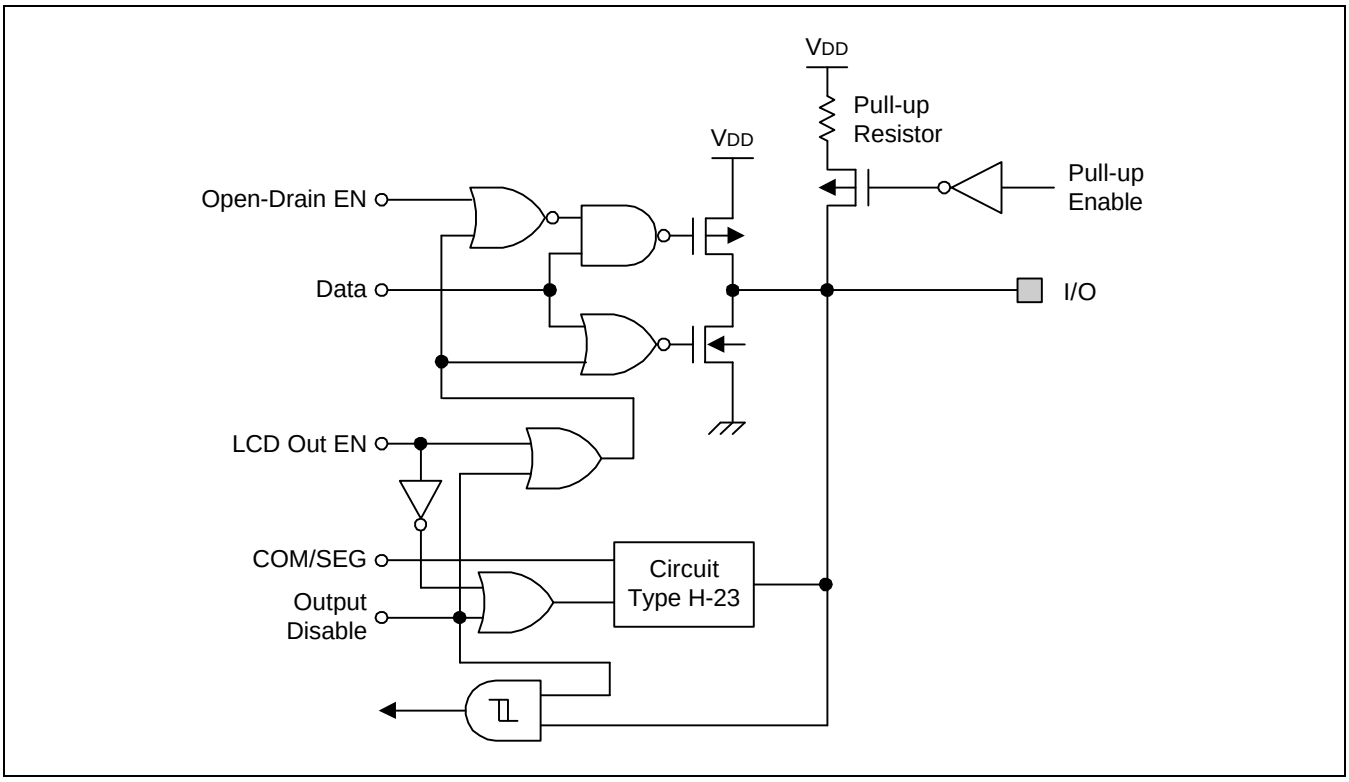


Figure 1-11. Pin Circuit Type H-32A

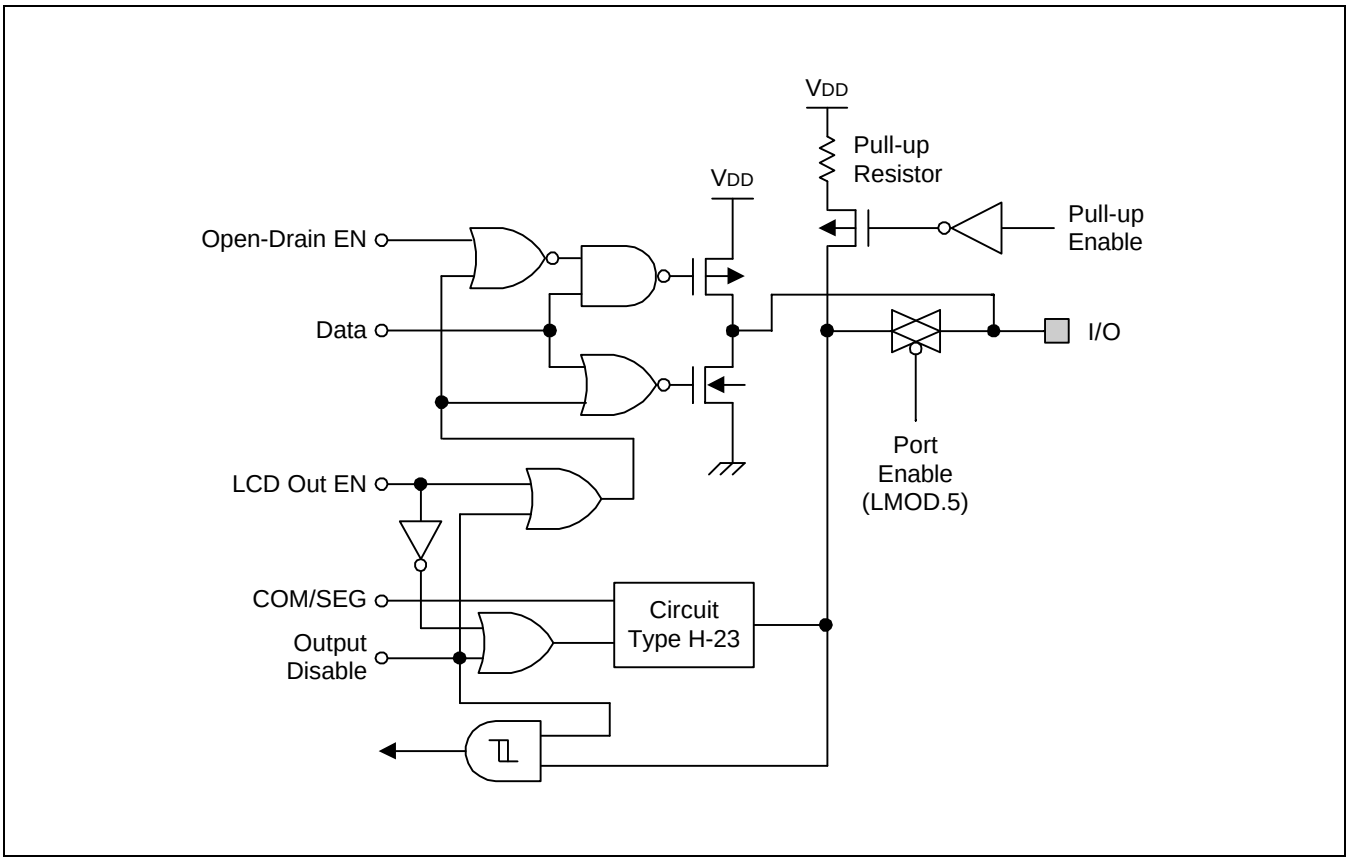


Figure 1-12. Pin Circuit Type H-32B

2 ADDRESS SPACES

OVERVIEW

The S3C9228/P9228 microcontroller has three kinds of address space:

— Program memory (ROM)
— Internal register file
— LCD display register file

A 16-bit address bus supports program memory operations. Special instructions and related internal logic determine when the 16-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 S3C9228 has 8K bytes of mask-programmable program memory on-chip. The S3C9228/P9228 microcontroller has 244 bytes general-purpose registers in its internal register file and the 20 bytes for LCD display memory is implemented in the internal register file too. Fifty-six bytes in the register file are mapped for system and peripheral control functions.

PROGRAM MEMORY (ROM)

Program memory (ROM) stores program code or table data. The S3C9228 has 8K bytes of mask-programable program memory. The program memory address range is therefore 0H-1FFFH. The first 2 bytes of the ROM (0000H–0001H) are an interrupt vector address. The program reset address in the ROM is 0100H.

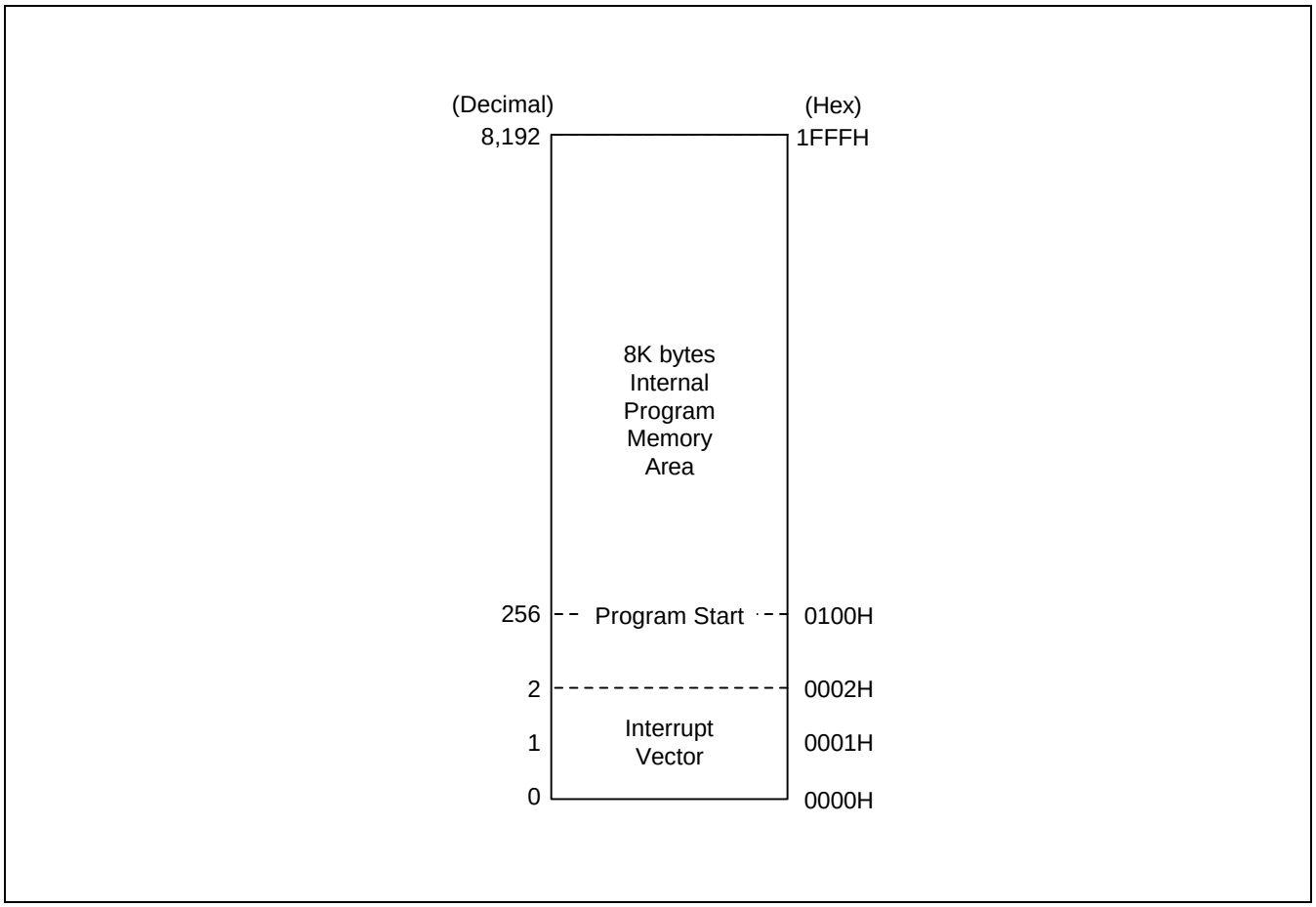


Figure 2-1. S3C9228/P9228 Program Memory Address Space

REGISTER ARCHITECTURE

The upper 72 bytes of the S3C9228/P9228's internal register file are addressed as working registers, system control registers and peripheral control registers. The lower 184 bytes of internal register file (00H–B7H) 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 implemented by page 1 in the S3C9228/P9228. The page 1 (20 × 8 bits) is for LCD display register and can be used as general-purpose registers.

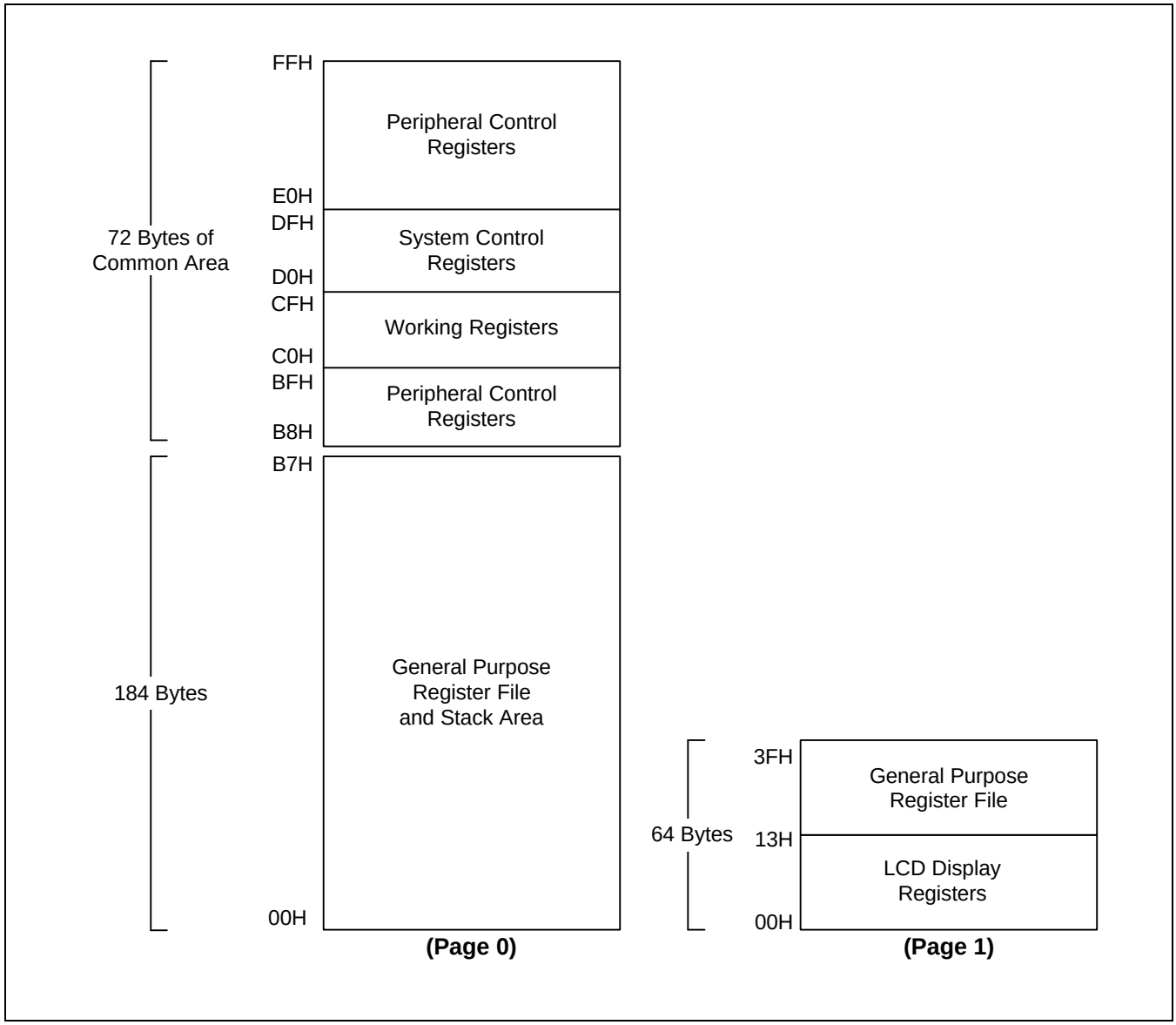


Figure 2-2. Internal Register File Organization

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.

This16-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.

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.

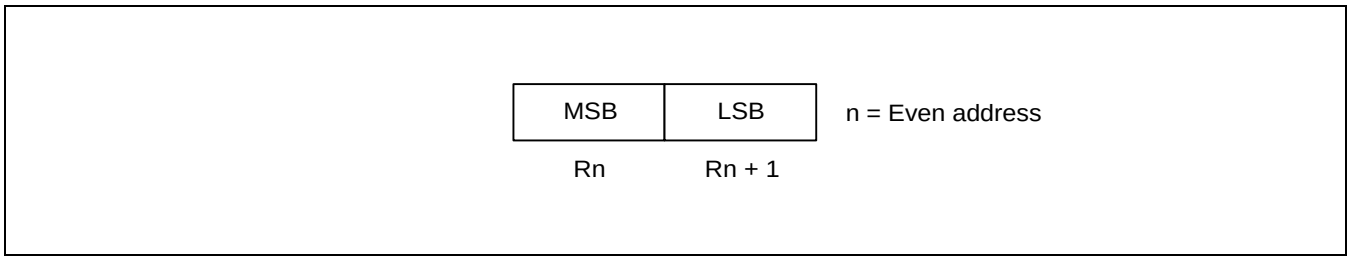


Figure 2-3. 16-Bit Register Pairs

☞ 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 S3C9228/P9228 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.

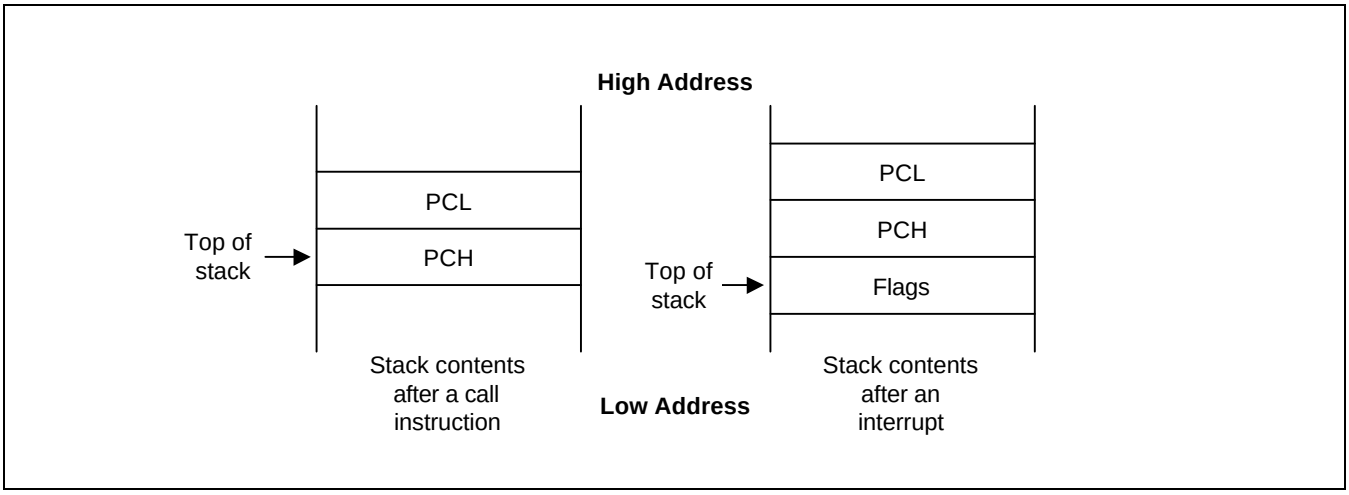


Figure 2-4. Stack Operations

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 S3C9228/P9228, the SP must be initialized to an 8-bit value in the range 00H–B7H.

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,#0B8H        ; SP ← B8H (Normally, the SP is set to 0B8H by the
                        ; initialization routine)
•
•
•
PUSH    SYM             ; Stack address 0B7H ← SYM
PUSH    WTCON           ; Stack address 0B6H ← WTCON
PUSH    20H             ; Stack address 0B5H ← 20H
PUSH    R3              ; Stack address 0B4H ← R3
•
•
•
POP     R3              ; R3 ← Stack address 0B4H
POP     20H             ; 20H ← Stack address 0B5H
POP     WTCON           ; WTCON ← Stack address 0B6H
POP     SYM             ; SYM ← Stack address 0B7H
```

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).

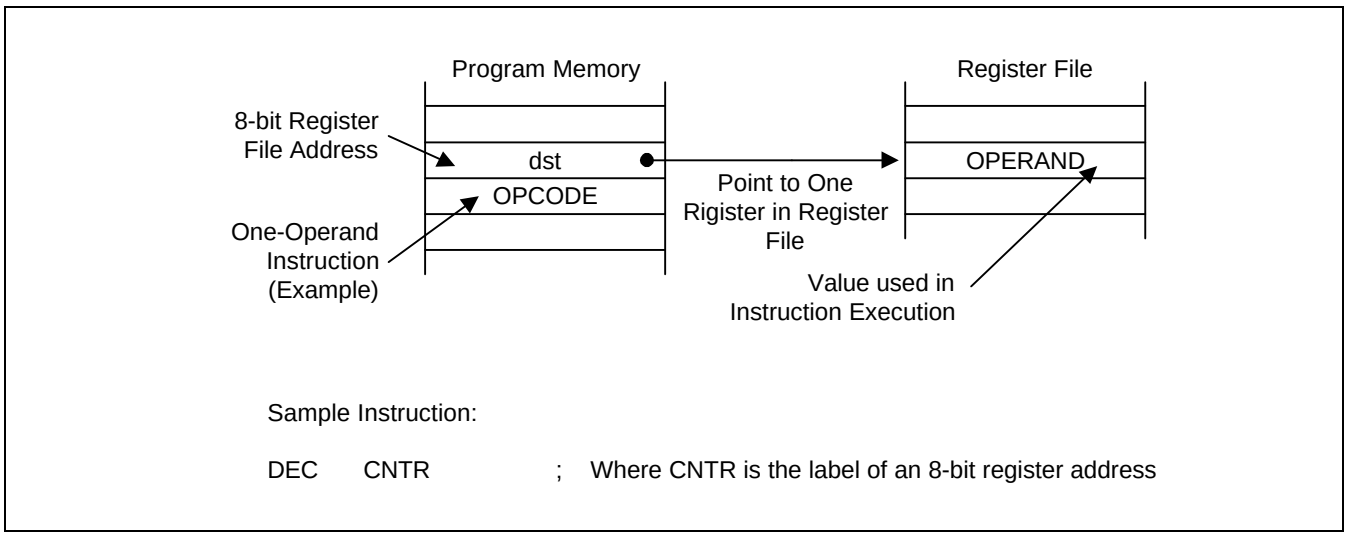


Figure 3-1. Register Addressing

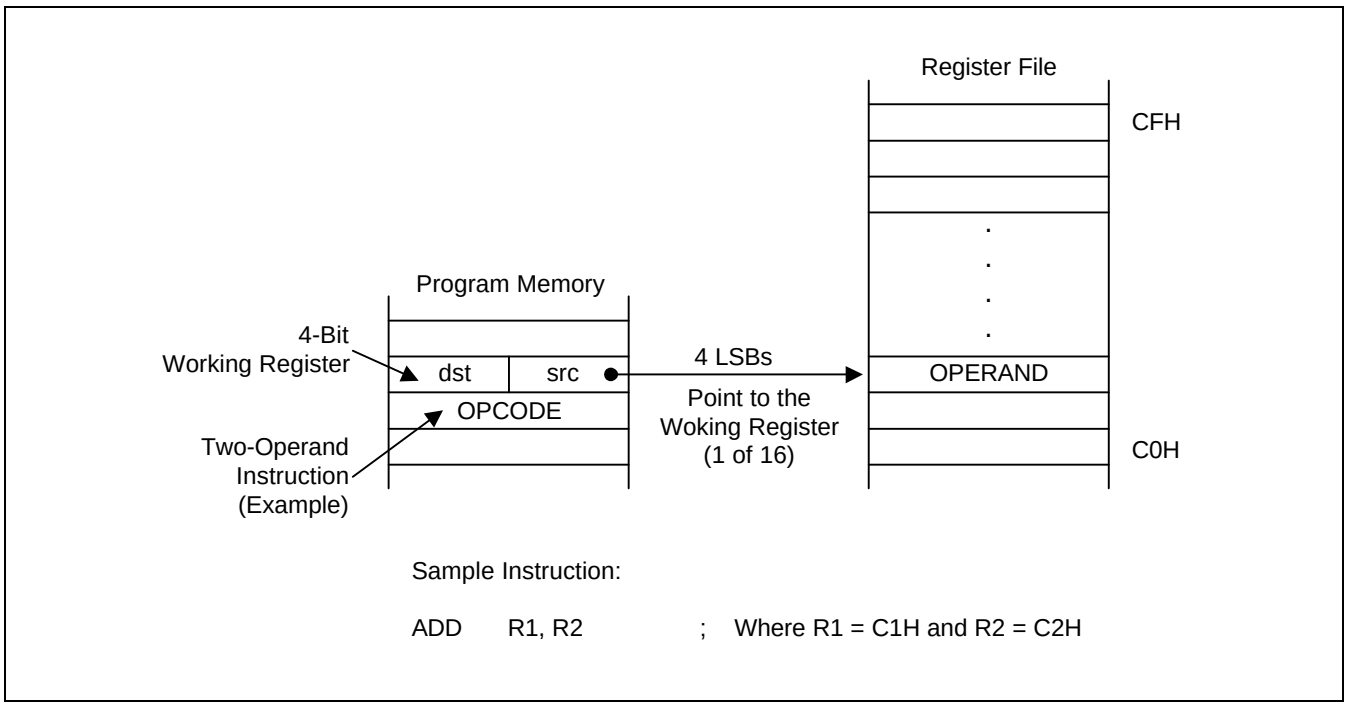


Figure 3-2. Working Register Addressing

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.

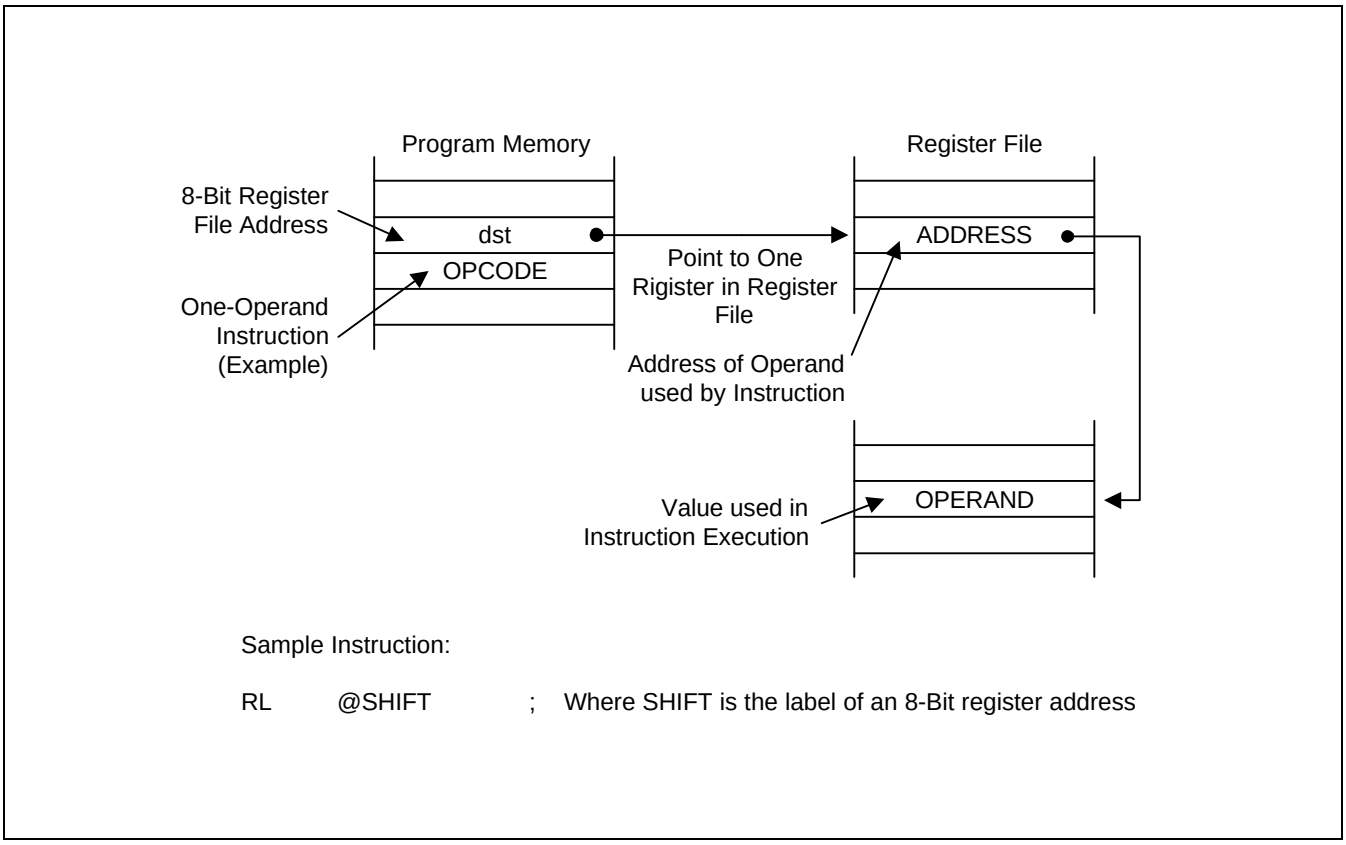


Figure 3-3. Indirect Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Continued)

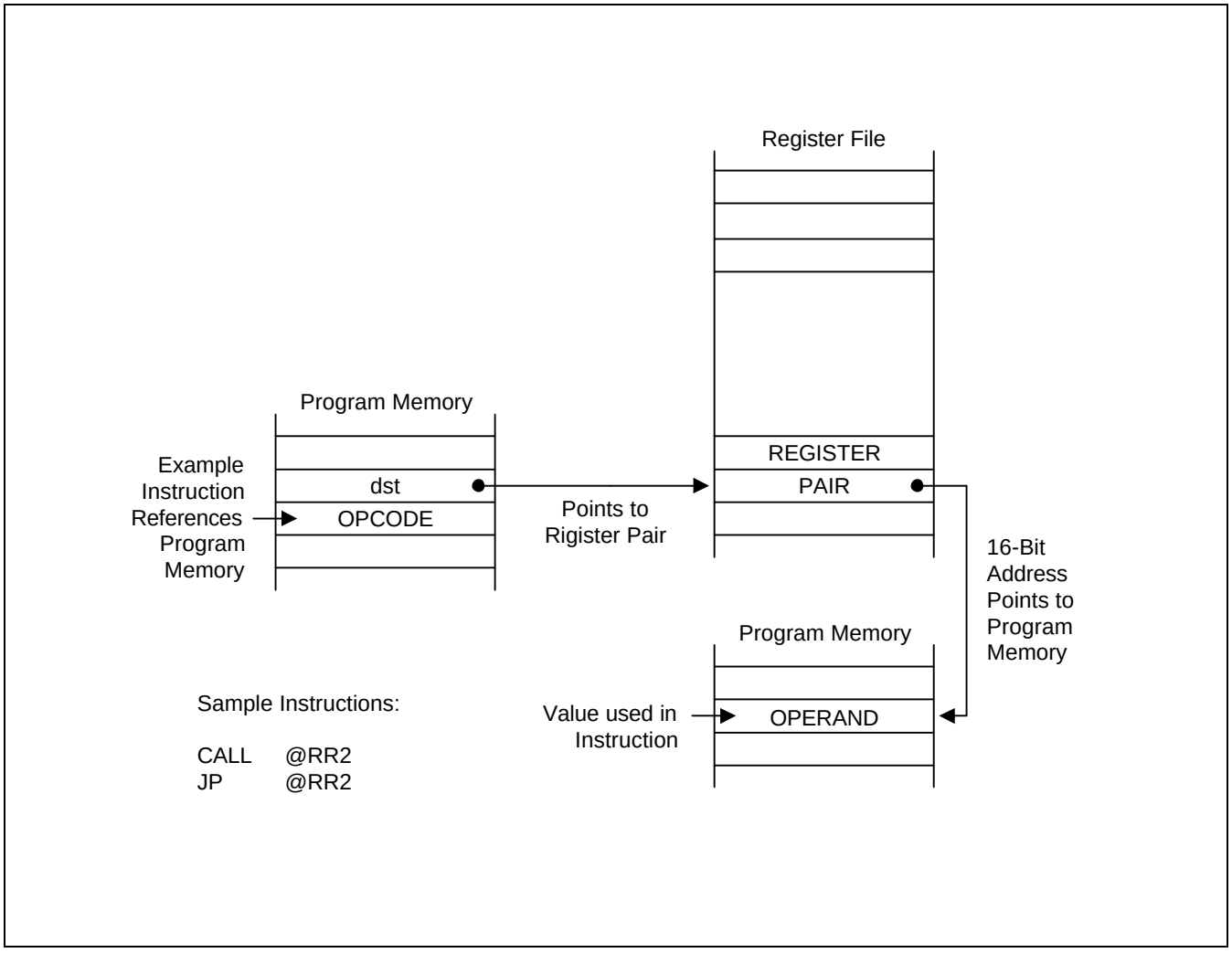


Figure 3-4. Indirect Register Addressing to Program Memory

INDIRECT REGISTER ADDRESSING MODE (Continued)

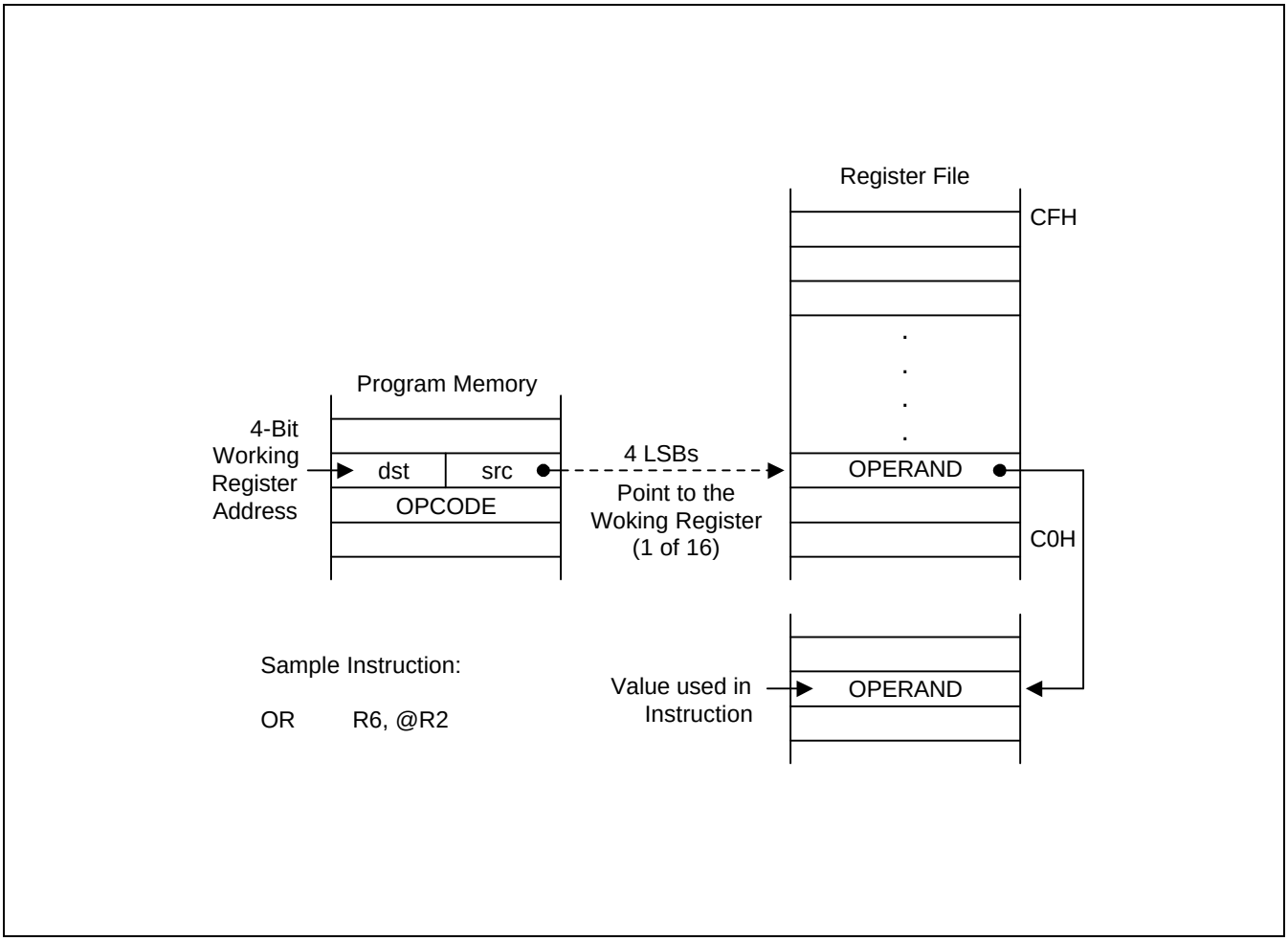


Figure 3-5. Indirect Working Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Concluded)

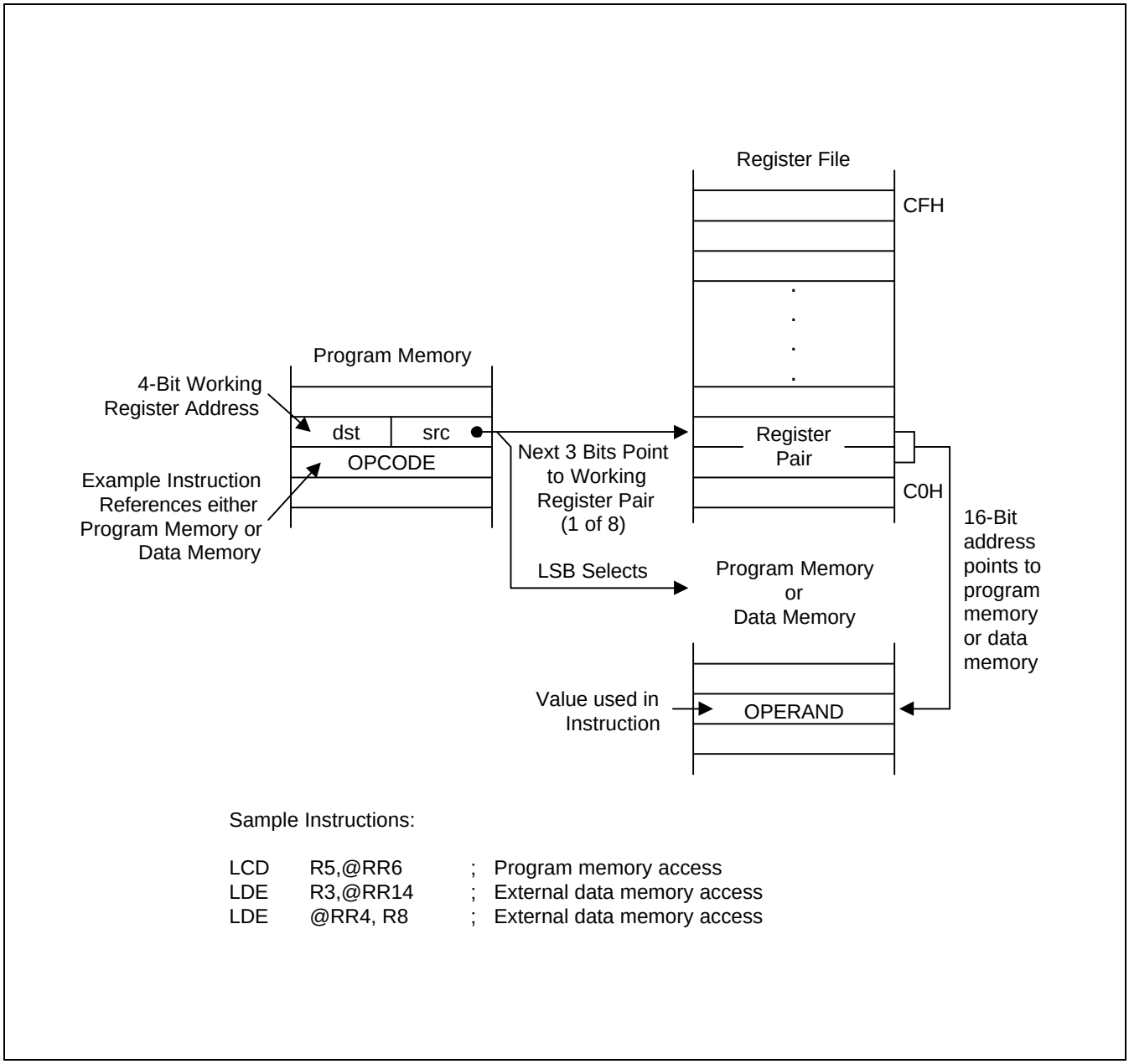


Figure 3-6. Indirect Working Register Addressing to Program or Data Memory

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.

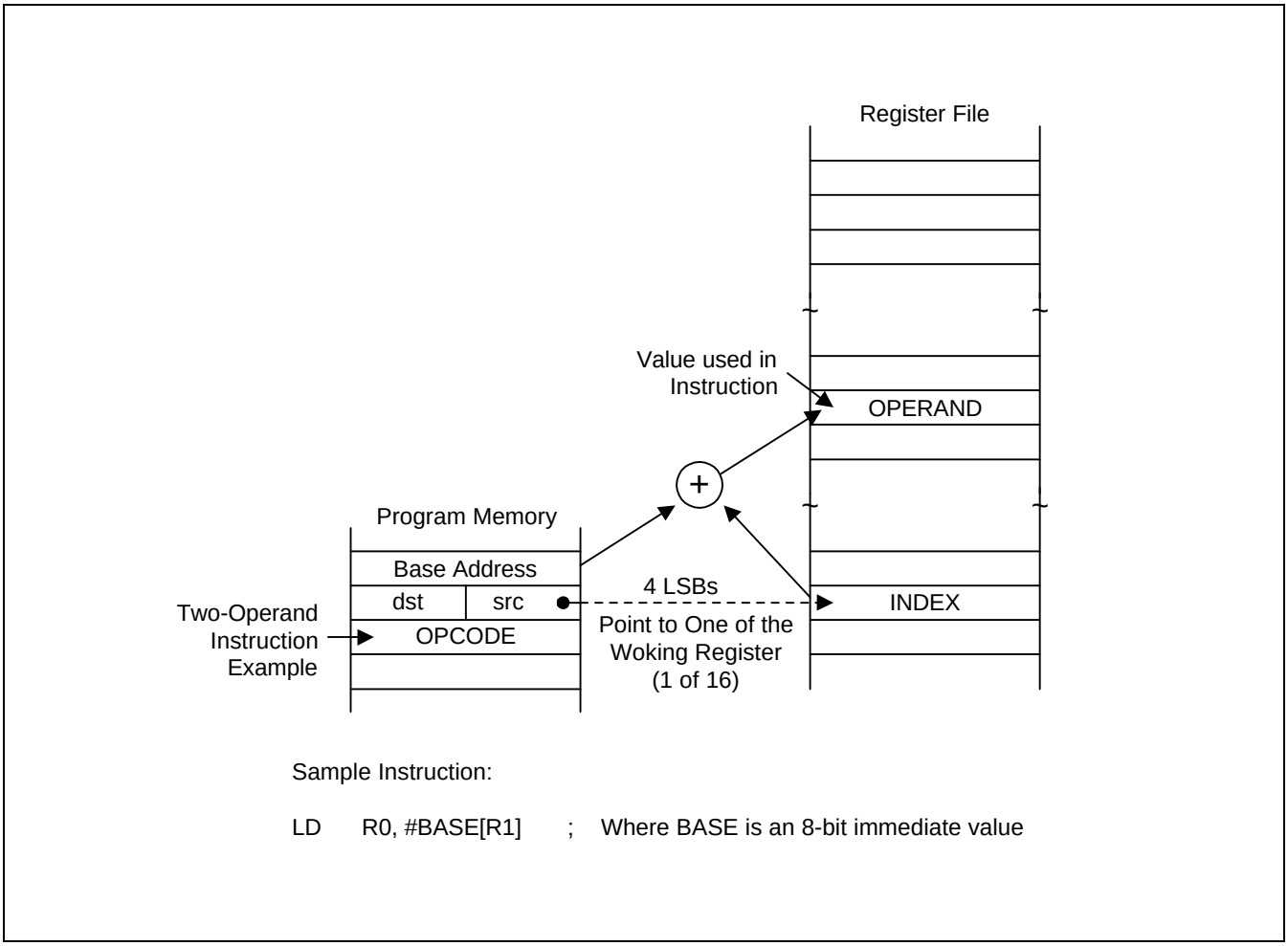
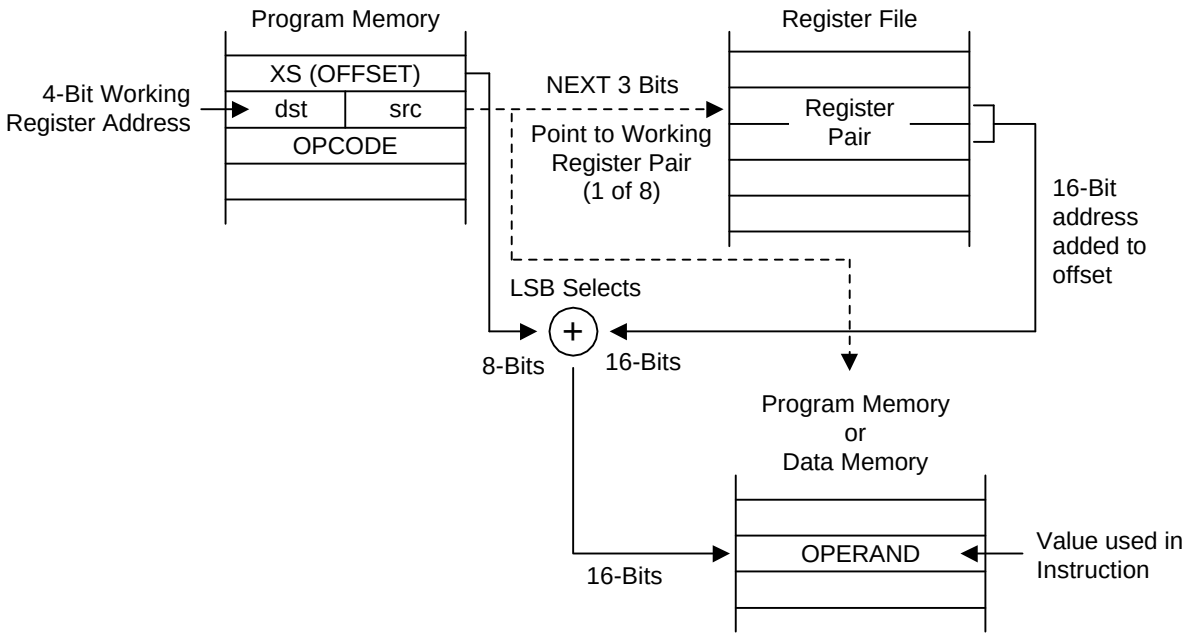


Figure 3-7. Indexed Addressing to Register File

INDEXED ADDRESSING MODE (Continued)

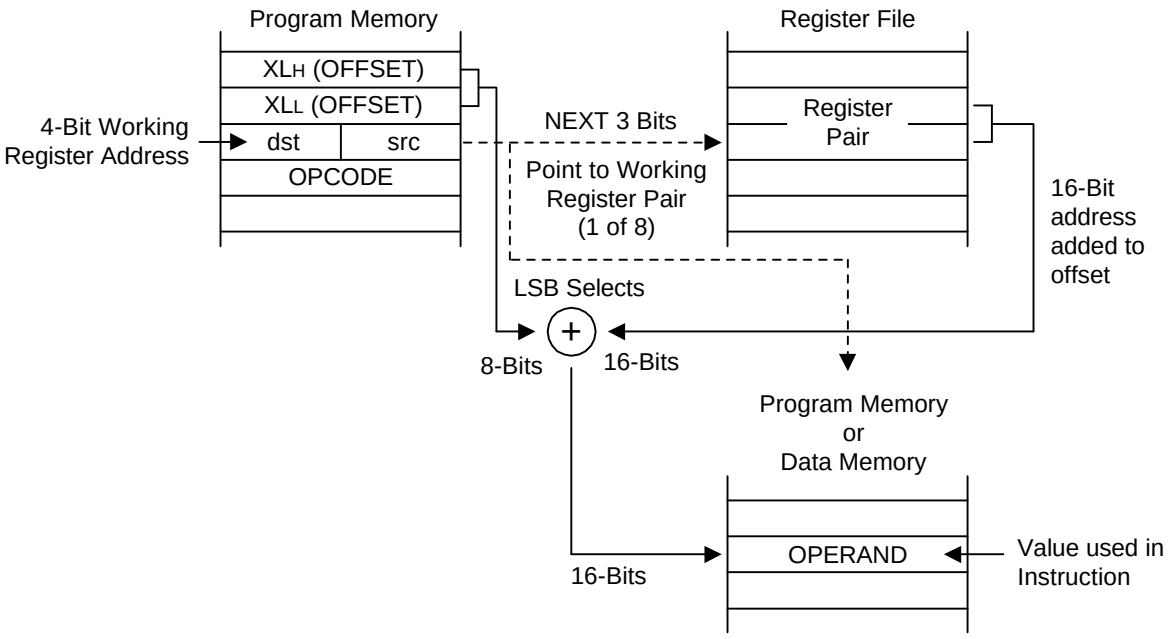


Sample Instructions:

```
LDC     R4, #04H[RR2]     ; The values in the program address (RR2 + #04H)
                          ; are loaded into register R4.
LDE     R4,#04H[RR2]      ; Identical operation to LDC example, except that
                          ; external program memory is accessed.
```

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

INDEXED ADDRESSING MODE (Concluded)



Sample Instructions:

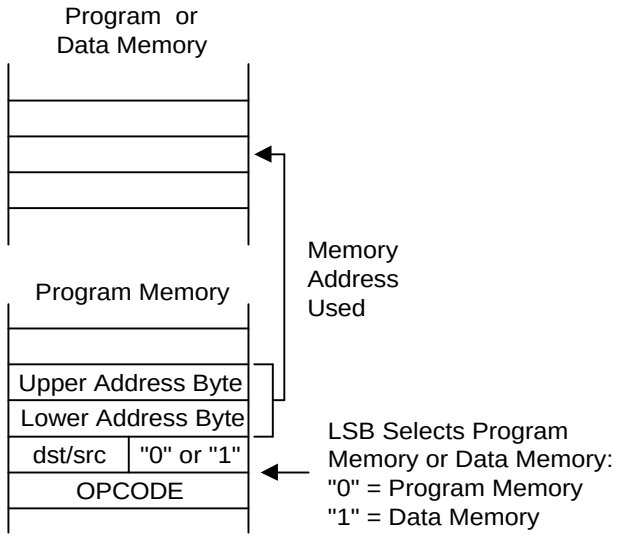
```
LDC     R4, #1000H[RR2]     ; The values in the program address (RR2 +
#1000H)
                            ; are loaded into register R4.
LDE     R4, #1000H[RR2]     ; Identical operation to LDC example, except that
                            ; external program memory is accessed.
```

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.

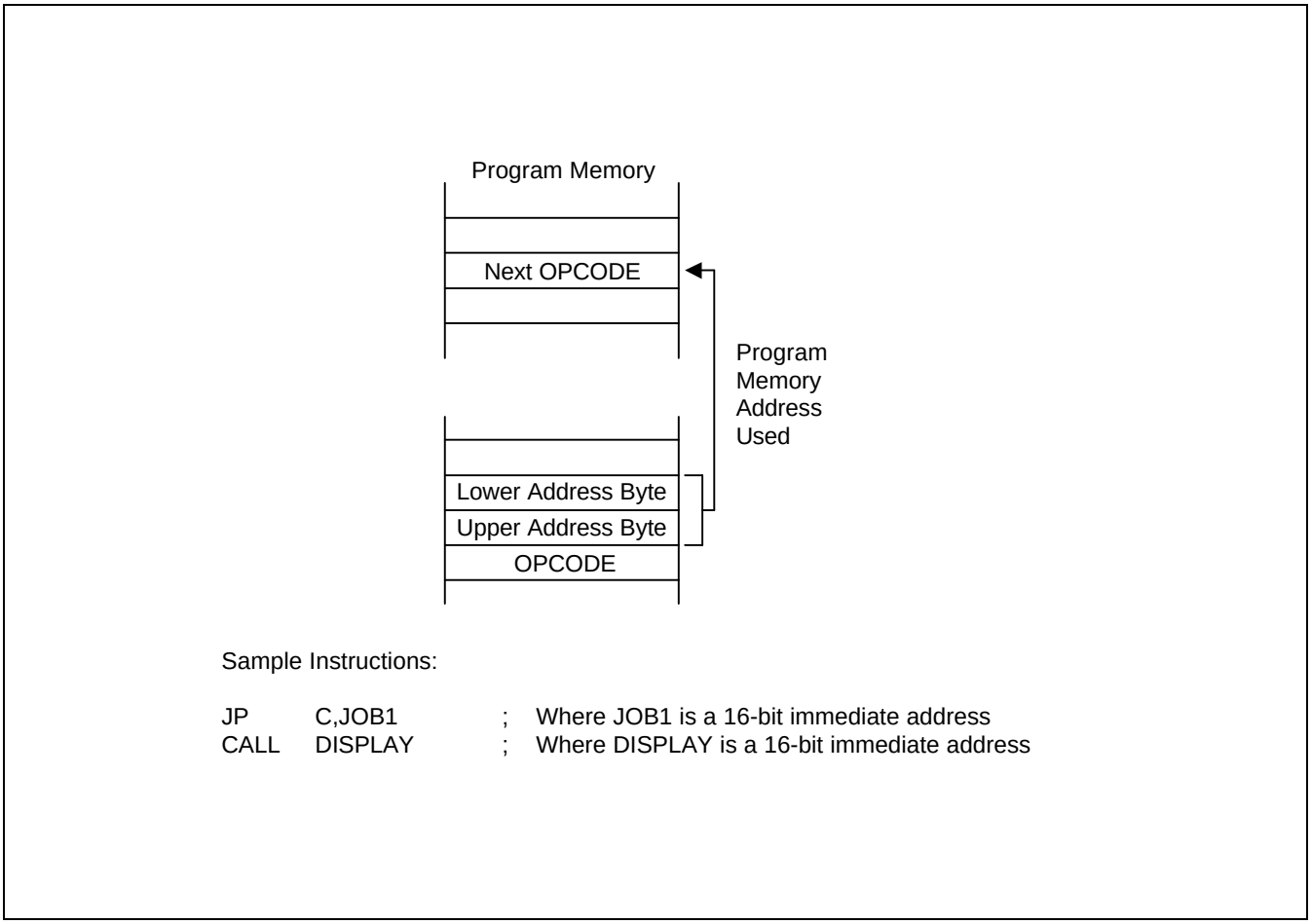


Sample Instructions:

```
LDC     R5,1234H        ;   The values in the program address (1234H)
                            are loaded into register R5.
LDE     R5,1234H        ;   Identical operation to LDC example, except that
                            external program memory is accessed.
```

Figure 3-10. Direct Addressing for Load Instructions

DIRECT ADDRESS MODE (Continued)



Sample Instructions:

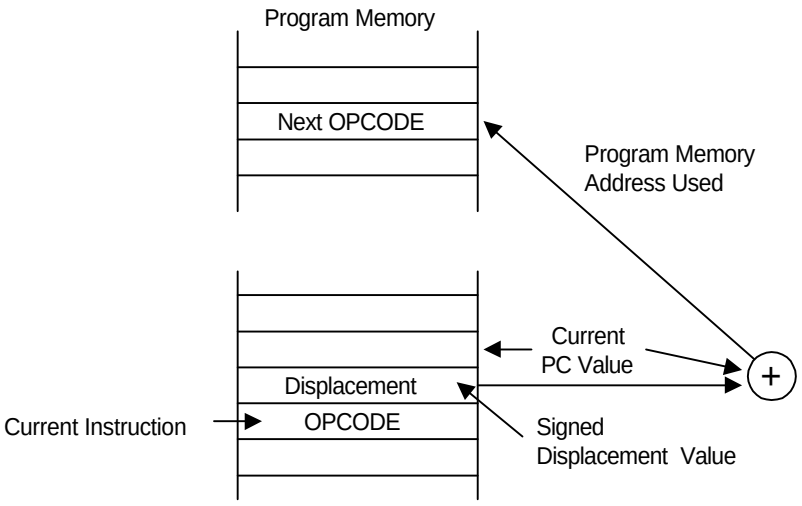
```
JP      C,JOB1      ;   Where JOB1 is a 16-bit immediate address
CALL    DISPLAY     ;   Where DISPLAY is a 16-bit immediate address
```

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.



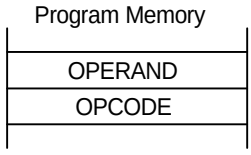
Sample Instructions:

```
JR      ULT,$ + OFFSET      ;   Where OFFSET is a value in the range + 127 to - 128
```

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.



(The Operand value is in the instruction)

Sample Instruction:

```
LD      R0,#0AAH
```

Figure 3-13. Immediate Addressing

4 CONTROL REGISTERS

OVERVIEW

In this section, detailed descriptions of the S3C9228/P9228 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. System and Peripheral Control Registers (Page 0)

Register Name	Mnemonic	Address (Page 0)		R/W
		Decimal	Hex	
Port 0 Control Register	P0CON	235	EBH	R/W
Port 0 Pull-up Resistor Enable Register	P0PUR	236	ECH	R/W
Port 0 Interrupt Control Register	P0INT	237	EDH	R/W
Port 0 Interrupt Edge Selection Register	P0EDGE	238	EEH	R/W
Port 1 Control Register	P1CON	239	EFH	R/W
Port 1 Pull-up Resistor Enable Register	P1PUR	240	F0H	R/W
Port 1 Interrupt Control Register	P1INT	241	F1H	R/W
Port 1 Interrupt Edge Selection Register	P1EDGE	242	F2H	R/W
Port 2 Control Register	P2CON	243	F3H	R/W
Port 2 Pull-up Resistor Enable Register	P2PUR	244	F4H	R/W
Port 3 Control Register	P3CON	245	F5H	R/W
Port 3 Pull-up Resistor Enable Register	P3PUR	246	F6H	R/W
Port 3 Interrupt Control Register	P3INT	247	F7H	R/W
Port 3 Interrupt Edge Selection Register	P3EDGE	248	F8H	R/W
Port 4 Control Register (High Byte)	P4CONH	249	F9H	R/W
Port 4 Control Register (Low Byte)	P4CONL	250	FAH	R/W
Port 5 Control Register (High Byte)	P5CONH	251	FBH	R/W
Port 5 Control Register (Low Byte)	P5CONL	252	FCH	R/W
Port 6 Control Register	P6CON	253	FDH	R/W
LCD Mode Register	LMOD	254	FEH	R/W
Location FFH is not mapped.				

Table 4-1. System and Peripheral Control Registers (Page 0)

Register Name	Mnemonic	Address (Page 0)		R/W
		Decimal	Hex	
Locations D8H-B9H are not mapped.				
Timer B Control Register	TBCON	202	BAH	R/W
Timer 1/A Control Register	TACON	203	BBH	R/W
Timer B Data Register	TBDATA	204	BCH	R/W
Timer A Data Register	TADATA	205	BDH	R/W
Timer B Counter	TBCNT	206	BEH	R
Timer A Counter	TACNT	207	BFH	R
A/D Converter Control Register	ADCON	208	D0H	R/W
A/D Converter Data Register (High Byte)	ADDATAH	209	D1H	R/W
A/D Converter Data Register (Low Byte)	ADDATAL	210	D2H	R/W
Oscillator Control Register	OSCCON	211	D3H	R/W
System Clock Control Register	CLKCON	212	D4H	R/W
System Flags Register	FLAGS	213	D5H	R/W
Interrupt Pending Register 1	INTPND1	214	D6H	R/W
Interrupt Pending Register 2	INTPND2	215	D7H	R/W
LCD Port Control Register	LPOT	216	D8H	R/W
Stack Pointer	SP	217	D9H	R/W
Watch Timer Control Register	WTCON	218	DAH	R/W
Location DBH is not mapped.				
Basic Timer Control Register	BTCON	220	DCH	R/W
Basic Timer Counter	BTCNT	221	DDH	R
Location DEH is not mapped.				
System Mode Register	SYM	223	DFH	R/W
STOP Control Register	STPCON	224	E0H	R/W
SIO Control Register	SIOCON	225	E1H	R/W
SIO Data Register	SIODATA	226	E2H	R/W
SIO Prescaler Register	SIOPS	227	E3H	R/W
Port 0 Data Register	P0	228	E4H	R/W
Port 1 Data Register	P1	229	E5H	R/W
Port 2 Data Register	P2	230	E6H	R/W
Port 3 Data Register	P3	231	E7H	R/W
Port 4 Data Register	P4	232	E8H	R/W
Port 5 Data Register	P5	233	E9H	R/W
Port 6 Data Register	P6	234	EAH	R/W

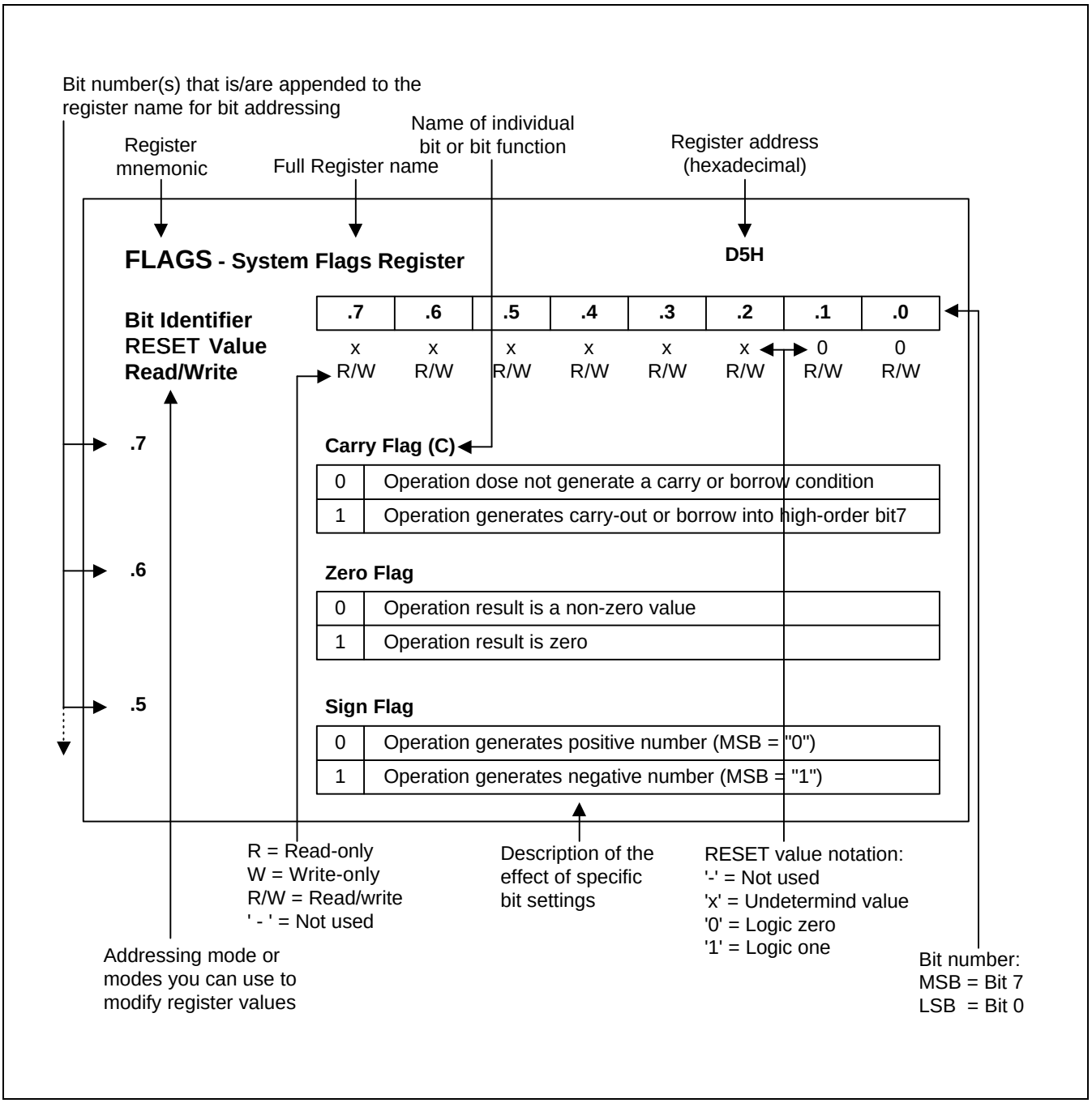


Figure 4-1. Register Description Format

ADCON — A/D Converter Control Register D0H

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	R/W	R/W	R/W

.7-.6 Not used for the S3C9228/P9228

.5-.4 A/D Input Pin Selection Bits

0	0	AD0 (P1.0)
0	1	AD1 (P1.1)
1	0	AD2 (P1.2)
1	1	AD3 (P1.3)

.3 End of Conversion Bit (Read-only)

0	Conversion not complete
1	Conversion complete

.2-.1 Clock Source Selection Bits

0	0	fxx/16
0	1	fxx/8
1	0	fxx/4
1	1	fxx

.0 Start or Enable Bit

0	Disable operation
1	Start operation (automatically disable operation after conversion complete)

BTCON — Basic Timer Control Register 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	fxx/4096
0	1	fxx/1024
1	0	fxx/128
1	1	fxx/16

.1 Basic Timer Counter Clear Bit (1)

0	No effect
1	Clear the basic timer counter value (BTCNT)

.0 Clock Frequency Divider Clear Bit for Basic Timer and Timer/Counters (2)

0	No effect
1	Clear clock frequency dividers

NOTES

1. When "1" is written to BTCON.1, the basic timer counter value is cleared to "00H". Immediately following the write operation, the BTCON.1 value is automatically cleared to "0".
2. When "1" is written to BTCON.0, the corresponding frequency divider is cleared to "00H". Immediately following the write operation, the BTCON.0 value is automatically cleared to "0".

CLKCON — System Clock Control Register D4H

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 Oscillator IRQ Wake-up Function Bit

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

.6-.5 Bits 6-5

0	Always logic zero

.4-.3 CPU Clock (System Clock) Selection Bits

0	0	Divide by 16 (fxx/16)
0	1	Divide by 8 (fxx /8)
1	0	Divide by 2 (fxx /2)
1	1	Non-divided clock (fxx)

.2-.0 Bits 2-0

0	Always logic zero

FLAGS — System Flags Register D5H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	–	–	–	–
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 $\geq$ –128
1	Operation result is $\geq$ +127 or $\leq$ –128

.3-.0 Not used for S3C9228/P9228

INTPND1 — Interrupt Pending Register 1 D6H

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 P1.3's Interrupt Pending Bit

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

.6 P1.2's Interrupt Pending Bit

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

.5 P1.1's Interrupt Pending Bit

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

.4 P1.0's Interrupt Pending Bit

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

.3 P0.3's Interrupt Pending Bit

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

.2 P0.2's Interrupt Pending Bit

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

.1 P0.1's Interrupt Pending Bit

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

.0 P0.0's Interrupt Pending Bit

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

NOTE: Refer to Page 5-6 to clear any pending bits.

INTPND2 — Interrupt Pending Register 2 D7H

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-.6 Not used for S3C9228/P9228

.5 P3.1 (INTP) Interrupt Pending Bit

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

.4 P3.0 (INTP) Interrupt Pending Bit

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

.3 Watch Timer Interrupt Pending Bit

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

.2 SIO Interrupt Pending Bit

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

.1 Timer B Interrupt Pending Bit

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

.0 Timer 1/A Interrupt Pending Bit

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

NOTE: Refer to Page 5-6 to clear any pending bits.

LMOD — LCD Mode Control Register FEH

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 S3C9228/P9228

.6 COM Pins High Impedance Control Bit

0	Normal COMs signal output
1	COM pins are at high impedance

.5 Port3 High Impedance Control Bit

0	Normal I/O
1	High impedance input

.4 LCD Display Control Bit

0	Display off (cut off the LCD voltage dividing resistors)
1	Normal display on

.3-.2 LCD Duty and Bias Selection Bits

0	0	1/3 duty, 1/3 bias; COM0–COM2/SEG0–SEG19
0	1	1/4 duty, 1/3 bias; COM0–COM3/SEG0–SEG19
1	0	1/8 duty, 1/4 bias; COM0–COM7/SEG0–SEG15
1	1	1/8 duty, 1/5 bias; COM0–COM7/SEG0–SEG15

.1-.0 LCD Clock Selection Bits

0	0	$fw/2^7$ (256 Hz when fw is 32.768 kHz)
0	1	$fw/2^6$ (512 Hz when fw is 32.768 kHz)
1	0	$fw/2^5$ (1,024 Hz when fw is 32.768 kHz)
1	1	$fw/2^4$ (2,048 Hz when fw is 32.768 kHz)

LPOT — LCD Port Control Register 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 S3C9228/P9228

.6-.4 SEG4-SEG19 and COM0-COM3 Selection Bit

			SEG4-7	SEG8-11	SEG12-15	SEG16-19/ COM7-COM4	COM0-3
			P4.0-P4.3	P4.4-P4.7	P5.0-P5.3	P5.4-P5.7	P6.0-P6.3
0	0	0	SEG	SEG	SEG	SEG/COM	COM
0	0	1	Port	SEG	SEG	SEG/COM	COM
0	1	0	Port	Port	SEG	SEG/COM	COM
0	1	1	Port	Port	Port	SEG/COM	COM
1	0	0	Port	Port	Port	Port	COM
1	0	1	Port	Port	Port	Port	Port

.3 SEG3/P3.0 Selection Bit

0	SEG port
1	Normal I/O port

.2 SEG2/P3.1 Selection Bit

0	SEG port
1	Normal I/O port

.1 SEG1/P2.0 Selection Bit

0	SEG port
1	Normal I/O port

.0 SEG0/P2.1 Selection Bit

0	SEG port
1	Normal I/O port

NOTE: SEG16-SEG19 are shared with COM4-COM7.

OSCCON — Oscillator Control Register D3H

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

.7-.4 Not used for S3C9228/P9228

.3 Main Oscillator Control Bit

0	Main oscillator RUN
1	Main oscillator STOP

.2 Sub Oscillator Control Bit

0	Sub oscillator RUN
1	Sub oscillator STOP

.1 Not used for S3C9228/P9228

.0 System Clock Selection Bit

0	Select main oscillator for system clock
1	Select sub oscillator for system clock

P0CON – Port 0 Control Register EBH

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 P0.3/BUZ/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (BUZ output)

.5-.4 P0.2/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.3-.2 P0.1/T1CLK/INT Configuration Bits

0	0	Schmitt trigger input (T1CLK input)
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.1-.0 P0.0/TAOUT/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (TAOUT output)

P0INT –Port 0 Interrupt Enable Register EDH

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 S3C9228/P9228

.3 P0.3's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.2 P0.2's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.1 P0.1's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.0 P0.0's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

P0PUR –Port 0 Pull-up Resistors Enable Register ECH

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 S3C9228/P9228

.3 P0.3's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.2 P0.2's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.1 P0.1's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.0 P0.0's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

P0EDGE –Port 0 Interrupt Edge Selection Register EEH

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 S3C9228/P9228

.3 P0.3's Interrupt Edge Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.2 P0.2's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.1 P0.1's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.0 P0.0's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

P1CON – Port 1 Control Register 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-.6 P1.3/AD3/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (ADC mode)

.5-.4 P1.2/AD2/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (ADC mode)

.3-.2 P1.1/AD1/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (ADC mode)

.1-.0 P1.0/AD0/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (ADC mode)

P1INT –Port 1 Interrupt Enable Register F1H

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 S3C9228/P9228

.3 **P1.3's Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.2 **P1.2's Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.1 **P1.1's Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.0 **P1.0's Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

P1PUR –Port 1 Pull-up Resistors Enable Register F0H

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 S3C9228/P9228

.3 P1.3's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.2 P1.2's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.1 P1.1's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.0 P1.0's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

P1EDGE –Port 1 Interrupt Edge Selection Register F2H

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 S3C9228/P9228

.3 P1.3's Interrupt Edge Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.2 P1.2's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.1 P1.1's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.0 P1.0's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

P2CON – Port 2 Control Register F3H

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 P2.3 Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.5-.4 P2.2/SI Configuration Bits

0	0	Schmitt trigger input (SI)
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.3-.2 P2.1/SO Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (SO out)

.1-.0 P2.0/SCK Configuration Bits

0	0	Schmitt trigger input (SCK in)
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (SCK out)

P2PUR –Port 2 Pull-up Resistors Enable Register F4H

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 S3C9228/P9228

.3 P2.3's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.2 P2.2's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.1 P2.1's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.0 P2.0's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

P3CON – Port 3 Control Register F5H

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 S3C9228/P9228

.3-.2 P3.1/SEG2/INTP Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.1-.0 P3.0/SEG3/INTP Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

P3INT –Port 3 Interrupt Enable Register F7H

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

.7-.2

Not used for S3C9228/P9228

.1 P3.1's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.0 P3.0's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

P3PUR –Port 3 Pull-up Resistors Enable Register F6H

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

.7-.2

Not used for S3C9228/P9228

.1 **P3.1's Pull-up Resistor Enable Bit**

0	Disable pull-up resistor
1	Enable pull-up resistor

.0 **P3.0's Pull-up Resistor Enable Bit**

0	Disable pull-up resistor
1	Enable pull-up resistor

P3EDGE –Port 3 Interrupt Edge Selection Register F8H

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

.7-.4

Not used for S3C9228/P9228

.1 P3.1's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.0 P3.0's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

P4CONH – Port 4 Control Register High Byte 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

.7-.6 P4.7/SEG11 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P4.6/SEG10 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P4.5/SEG9 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P4.4/SEG8 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

P4CONL–Port 4 Control Register Low Byte FAH

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 P4.3/SEG7 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P4.2/SEG6 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P4.1/SEG5 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P4.0/SEG4 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

P5CONH – Port 5 Control Register High Byte FBH

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 P5.7/SEG19/COM4 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P5.6/SEG18/COM5 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P5.5/SEG17/COM6 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P5.4/SEG16/COM7 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

P5CONL – Port 5 Control Register Low Byte FCH

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 P5.3/SEG15 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P5.2/SEG14 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P5.1/SEG13 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P5.0/SEG12 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

P6CON – Port 6 Control Register High Byte FDH

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 P6.3/COM0 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P6.2/COM1 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P6.1/COM2 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P6.0/COM3 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

SIOCON — SIO Control Register E1H

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 SIO Shift Clock Selection Bit

0	Internal clock (P.S clock)
1	External clock (SCK)

.6 Data Direction Control Bit

0	MSB-first mode
1	LSB-first mode

.5 SIO Mode Selection Bit

0	Receive-only mode
1	Transmit/receive mode

.4 Shift Clock Edge Selection Bit

0	Tx at falling edges, Rx at rising edges
1	Tx at rising edges, Rx at falling edges

.3 SIO Counter Clear and Shift Start Bit

0	No action
1	Clear 3-bit counter and start shifting

.2 SIO Shift Operation Enable Bit

0	Disable shifter and clock counter
1	Enable shifter and clock counter

.1 SIO Interrupt Enable Bit

0	Disable SIO interrupt
1	Enable SIO interrupt

.0

Not used for S3C9228/P9228

STPCON – Stop Control Register E0H

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 Stop Control Bits

1	0	1	0	0	1	0	1	Enable Stop instruction
Other values								Disable Stop instruction

NOTE: Before executing the STOP instruction, the STPCON register must be set to "10100101B". Otherwise the STOP instruction will not execute.

SYM — System Mode Register 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 S3C9228/P9228

.3 Global Interrupt Enable Bit

0	Global interrupt processing disable (DI instruction)
1	Global interrupt processing enable (EI instruction)

.2-.0 Page Selection Bits

0	0	0	Page 0
0	0	1	Page 1
Other values			Not available

TACON — Timer 1/A Control Register BBH

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 Timer 1 Mode Selection Bit

0	Two 8-bit timers mode (Timer A/B)
1	One 16-bit timer mode (Timer 1)

.6-.4 Timer 1/A Clock Selection Bits

0	0	0	fxx/512
0	0	1	fxx/256
0	1	0	fxx/64
0	1	1	fxx/8
1	0	0	fxx (system clock)
1	0	1	fxt (sub clock)
1	1	0	T1CLK (external clock)
1	1	1	Not available

.3 Timer 1/A Counter Clear Bit

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

.2 Timer 1/A Counter Enable Bit

0	Disable counting operation
1	Enable counting operation

.1 Timer 1/A Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.0 Bit 0

Not used for S3C9228/P9228

TBCON — Timer B Control Register BAH

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 Not used for S3C9228/P9228

.6-.4 Timer B Clock Selection Bits

0	0	0	fxx/512
0	0	1	fxx/256
0	1	0	fxx/64
0	1	1	fxx/8
1	0	0	fxx (system clock)
1	0	1	fxt (sub clock)

.3 Timer B Counter Clear Bit

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

.2 Timer B Counter Enable Bit

0	Disable counting operation
1	Enable counting operation

.1 Timer B Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.0 Not used for S3C9228/P9228

WTCON — Watch Timer Control Register DAH

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 Watch Timer Clock Selection Bit

0	Select main clock divided by 2^7 (fx/128)
1	Select sub clock (fxt)

.6 Watch Timer Interrupt Enable Bit

0	Disable watch timer interrupt
1	Enable watch timer interrupt

.5-.4 Buzzer Signal Selection Bits

0	0	0.5 kHz
0	1	1 kHz
1	0	2 kHz
1	1	4 kHz

.3-.2 Watch Timer Speed Selection Bits

0	0	Set watch timer interrupt to 1s
0	1	Set watch timer interrupt to 0.5s
1	0	Set watch timer interrupt to 0.25s
1	1	Set watch timer interrupt to 3.91ms

.1 Watch Timer Enable Bit

0	Disable watch timer; Clear frequency dividing circuits
1	Enable watch timer

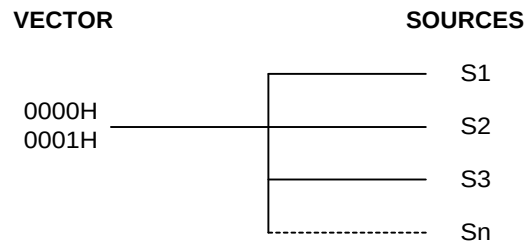
.0

Not used for S3C9228/P9228

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.



NOTES:
1. The SAM88RCRI interrupt has only one vector address (0000H-0001H).
2. The number of Sn value is expandable.

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 SAM87RCRI, the order of service is determined by a sequence of source which is executed in interrupt service routine.

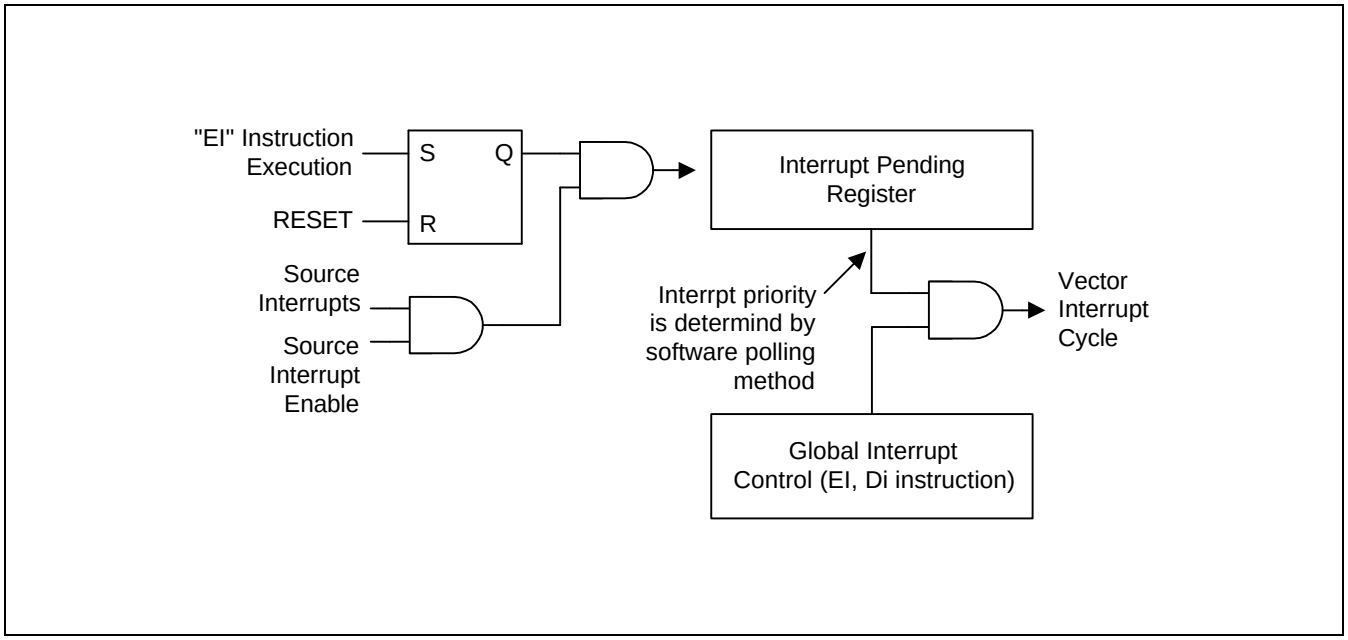


Figure 5-2. Interrupt Function Diagram

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.

S3C9228/P9228 INTERRUPT STRUCTURE

The S3C9228/P9228 microcontroller has fourteen peripheral interrupt sources:

— Timer 1/A interrupt
— Timer B interrupt
— SIO interrupt
— Watch Timer interrupt
— Four external interrupts for port 0
— Four external interrupts for port 1
— Two external interrupts for port 3

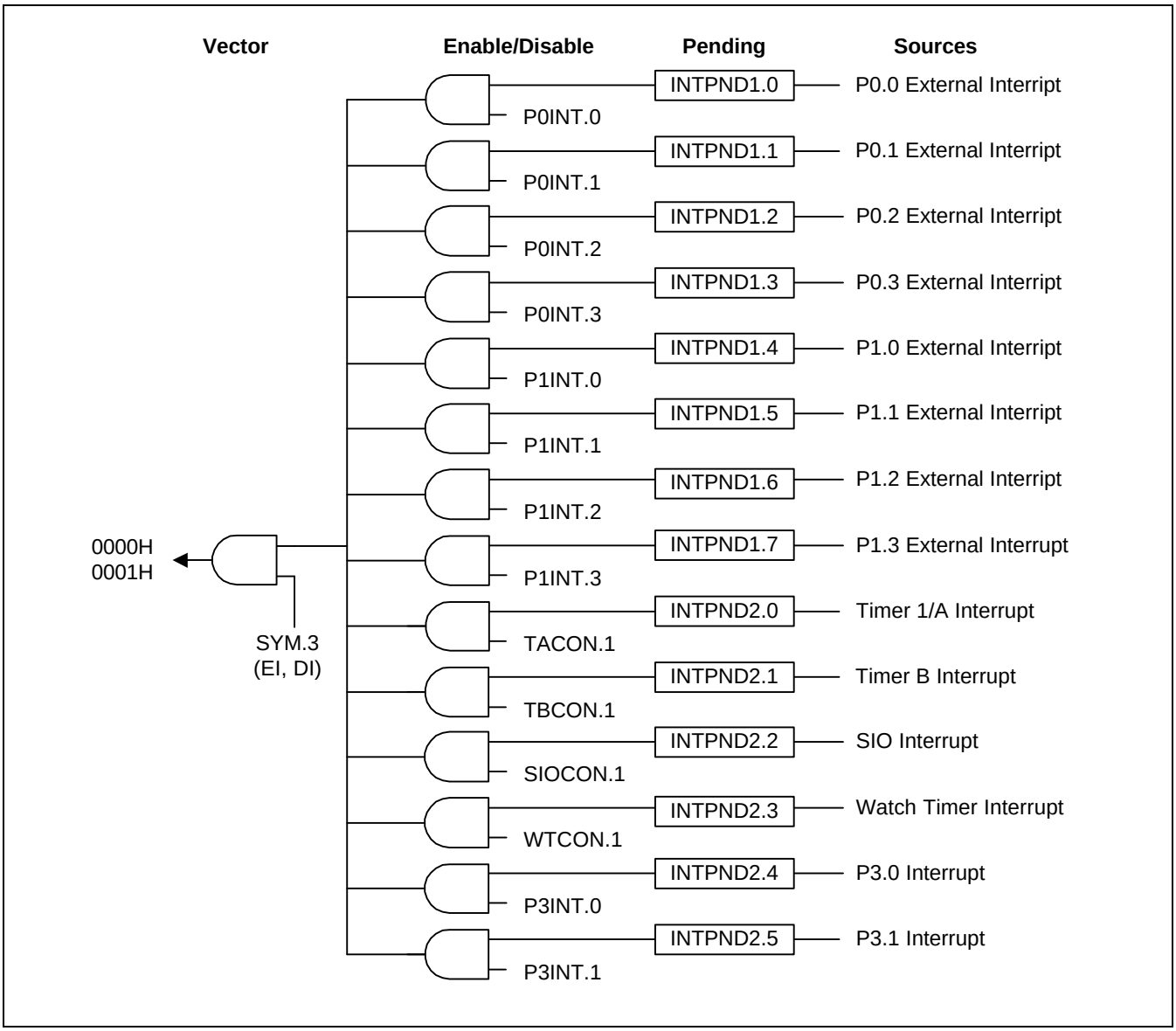


Figure 5-3. S3C9228/P9228 Interrupt Structure

Programming Tip — How to clear an interrupt pending bit

As the following examples are shown, a load instruction should be used to clear an interrupt pending bit.

Examples:

```
1.    LD      INTPND1, #11111011B     ; Clear P0.2's interrupt pending bit
      •
      •
      •
      IRET

2.    LD      INTPND2, #11110111B     ; Clear watch timer interrupt pending bit
      •
      •
      •
      IRET
```

6 SAM88RCRI INSTRUCTION SET

OVERVIEW

The SAM88RCRI instruction set is designed to support the large register file. It includes a full complement of 8-bit arithmetic and logic operations. There are 41 instructions. No special I/O instructions are necessary because I/O control and data registers are mapped directly into the register file. Flexible instructions for bit addressing, rotate, and shift operations complete the powerful data manipulation capabilities of the SAM88RCRI instruction set.

REGISTER ADDRESSING

To access an individual register, an 8-bit address in the range 0-255 or the 4-bit address of a working register is specified. Paired registers can be used to construct 16-bit program memory or data memory addresses. For detailed information about register addressing, please refer to Section 2, "Address Spaces".

ADDRESSING MODES

There are six addressing modes: Register (R), Indirect Register (IR), Indexed (X), Direct (DA), Relative (RA), and Immediate (IM). For detailed descriptions of these addressing modes, please refer to Section 3, "Addressing Modes".

Table 6-1. Instruction Group Summary

Mnemonic	Operands	Instruction
Load Instructions		
CLR	dst	Clear
LD	dst,src	Load
LDC	dst,src	Load program memory
LDE	dst,src	Load external data memory
LDCD	dst,src	Load program memory and decrement
LDED	dst,src	Load external data memory and decrement
LDCI	dst,src	Load program memory and increment
LDEI	dst,src	Load external data memory and increment
POP	dst	Pop from stack
PUSH	src	Push to stack
Arithmetic Instructions		
ADC	dst,src	Add with carry
ADD	dst,src	Add
CP	dst,src	Compare
DEC	dst	Decrement
INC	dst	Increment
SBC	dst,src	Subtract with carry
SUB	dst,src	Subtract
Logic Instructions		
AND	dst,src	Logical AND
COM	dst	Complement
OR	dst,src	Logical OR
XOR	dst,src	Logical exclusive OR

Table 6-1. Instruction Group Summary (Continued)

Mnemonic	Operands	Instruction
Program Control Instructions		
CALL	dst	Call procedure
IRET		Interrupt return
JP	cc,dst	Jump on condition code
JP	dst	Jump unconditional
JR	cc,dst	Jump relative on condition code
RET		Return
Bit Manipulation Instructions		
TCM	dst,src	Test complement under mask
TM	dst,src	Test under mask
Rotate and Shift Instructions		
RL	dst	Rotate left
RLC	dst	Rotate left through carry
RR	dst	Rotate right
RRC	dst	Rotate right through carry
SRA	dst	Shift right arithmetic
CPU Control Instructions		
CCF		Complement carry flag
DI		Disable interrupts
EI		Enable interrupts
IDLE		Enter Idle mode
NOP		No operation
RCF		Reset carry flag
SCF		Set carry flag
STOP		Enter Stop mode

FLAGS REGISTER (FLAGS)

The FLAGS register contains eight bits that describe the current status of CPU operations. Four of these bits, FLAGS.4 – FLAGS.7, can be tested and used with conditional jump instructions;

FLAGS register can be set or reset by instructions as long as its outcome does not affect the flags, such as, Load instruction. Logical and Arithmetic instructions such as, AND, OR, XOR, ADD, and SUB can affect the Flags register. For example, the AND instruction updates the Zero, Sign and Overflow flags based on the outcome of the AND instruction. If the AND instruction uses the Flags register as the destination, then simultaneously, two write will occur to the Flags register producing an unpredictable result.

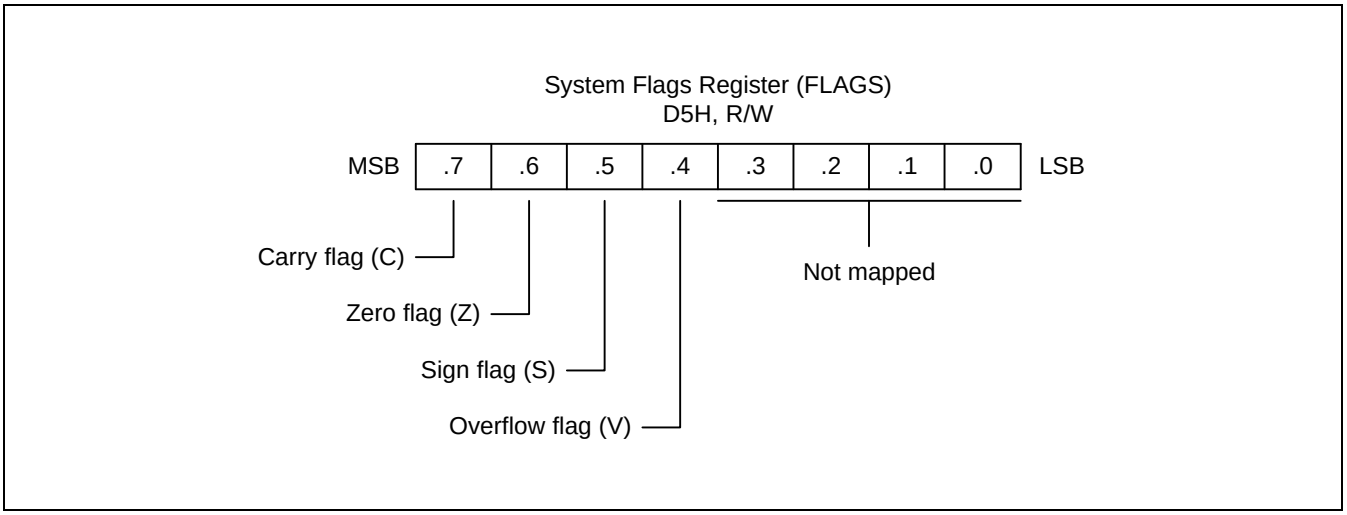


Figure 6-1. System Flags Register (FLAGS)

FLAG DESCRIPTIONS

Overflow Flag (FLAGS.4, V)

The V flag is set to "1" when the result of a two's-complement operation is greater than + 127 or less than – 128. It is also cleared to "0" following logic operations.

Sign Flag (FLAGS.5, S)

Following arithmetic, logic, rotate, or shift operations, the sign bit identifies the state of the MSB of the result. A logic zero indicates a positive number and a logic one indicates a negative number.

Zero Flag (FLAGS.6, Z)

For arithmetic and logic operations, the Z flag is set to "1" if the result of the operation is zero. For operations that test register bits, and for shift and rotate operations, the Z flag is set to "1" if the result is logic zero.

Carry Flag (FLAGS.7, C)

The C flag is set to "1" if the result from an arithmetic operation generates a carry-out from or a borrow to the bit 7 position (MSB). After rotate and shift operations, it contains the last value shifted out of the specified register. Program instructions can set, clear, or complement the carry flag.

INSTRUCTION SET NOTATION

Table 6-2. Flag Notation Conventions

Flag	Description
C	Carry flag
Z	Zero flag
S	Sign flag
V	Overflow flag
0	Cleared to logic zero
1	Set to logic one
*	Set or cleared according to operation
–	Value is unaffected
x	Value is undefined

Table 6-3. Instruction Set Symbols

Symbol	Description
dst	Destination operand
src	Source operand
@	Indirect register address prefix
PC	Program counter
FLAGS	Flags register (D5H)
#	Immediate operand or register address prefix
H	Hexadecimal number suffix
D	Decimal number suffix
B	Binary number suffix
opc	Opcode

Table 6-4. Instruction Notation Conventions

Notation	Description	Actual Operand Range
cc	Condition code	See list of condition codes in Table 6-6.
r	Working register only	Rn (n = 0–15)
rr	Working register pair	RRp (p = 0, 2, 4, ..., 14)
R	Register or working register	reg or Rn (reg = 0–255, n = 0–15)
RR	Register pair or working register pair	reg or RRp (reg = 0–254, even number only, where p = 0, 2, ..., 14)
Ir	Indirect working register only	@Rn (n = 0–15)
IR	Indirect register or indirect working register	@Rn or @reg (reg = 0–255, n = 0–15)
Irr	Indirect working register pair only	@RRp (p = 0, 2, ..., 14)
IRR	Indirect register pair or indirect working register pair	@RRp or @reg (reg = 0–254, even only, where p = 0, 2, ..., 14)
X	Indexed addressing mode	#reg[Rn] (reg = 0–255, n = 0–15)
XS	Indexed (short offset) addressing mode	#addr[RRp] (addr = range –128 to +127, where p = 0, 2, ..., 14)
xl	Indexed (long offset) addressing mode	#addr [RRp] (addr = range 0–8191, where p = 0, 2, ..., 14)
da	Direct addressing mode	addr (addr = range 0–8191)
ra	Relative addressing mode	addr (addr = number in the range +127 to –128 that is an offset relative to the address of the next instruction)
im	Immediate addressing mode	#data (data = 0–255)

Table 6-5. Opcode Quick Reference

OPCODE MAP									
	LOWER NIBBLE (HEX)								
	–	0	1	2	3	4	5	6	7
U	0	DEC R1	DEC IR1	ADD r1,r2	ADD r1,Ir2	ADD R2,R1	ADD IR2,R1	ADD R1,IM	
P	1	RLC R1	RLC IR1	ADC r1,r2	ADC r1,Ir2	ADC R2,R1	ADC IR2,R1	ADC R1,IM	
P	2	INC R1	INC IR1	SUB r1,r2	SUB r1,Ir2	SUB R2,R1	SUB IR2,R1	SUB R1,IM	
E	3	JP IRR1		SBC r1,r2	SBC r1,Ir2	SBC R2,R1	SBC IR2,R1	SBC R1,IM	
R	4			OR r1,r2	OR r1,Ir2	OR R2,R1	OR IR2,R1	OR R1,IM	
	5	POP R1	POP IR1	AND r1,r2	AND r1,Ir2	AND R2,R1	AND IR2,R1	AND R1,IM	
N	6	COM R1	COM IR1	TCM r1,r2	TCM r1,Ir2	TCM R2,R1	TCM IR2,R1	TCM R1,IM	
I	7	PUSH R2	PUSH IR2	TM r1,r2	TM r1,Ir2	TM R2,R1	TM IR2,R1	TM R1,IM	
B	8								LD r1, x, r2
B	9	RL R1	RL IR1						LD r2, x, r1
L	A			CP r1,r2	CP r1,Ir2	CP R2,R1	CP IR2,R1	CP R1,IM	LDC r1, Irr2, xL
E	B	CLR R1	CLR IR1	XOR r1,r2	XOR r1,Ir2	XOR R2,R1	XOR IR2,R1	XOR R1,IM	LDC r2, Irr2, xL
	C	RRC R1	RRC IR1		LDC r1,Irr2				LD r1, Ir2
H	D	SRA R1	SRA IR1		LDC r2,Irr1			LD IR1,IM	LD Ir1, r2
E	E	RR R1	RR IR1	LDCD r1,Irr2	LDCI r1,Irr2	LD R2,R1	LD R2,IR1	LD R1,IM	LDC r1, Irr2, xs
X	F					CALL IRR1	LD IR2,R1	CALL DA1	LDC r2, Irr1, xs

Table 6-5. Opcode Quick Reference (Continued)

OPCODE MAP									
	LOWER NIBBLE (HEX)								
	–	8	9	A	B	C	D	E	F
U	0	LD r1,R2	LD r2,R1		JR cc,RA	LD r1,IM	JP cc,DA	INC r1	
P	1	↓	↓		↓	↓	↓	↓	
P	2								
E	3								
R	4								
	5								
N	6								IDLE
I	7	↓	↓		↓	↓	↓	↓	STOP
B	8								DI
B	9								EI
L	A								RET
E	B								IRET
	C								RCF
H	D	↓	↓		↓	↓	↓	↓	SCF
E	E								CCF
X	F	LD r1,R2	LD r2,R1		JR cc,RA	LD r1,IM	JP cc,DA	INC r1	NOP

CONDITION CODES

The opcode of a conditional jump always contains a 4-bit field called the condition code (cc). This specifies under which conditions it is to execute the jump. For example, a conditional jump with the condition code for "equal" after a compare operation only jumps if the two operands are equal. Condition codes are listed in Table 6-6.

The carry (C), zero (Z), sign (S), and overflow (V) flags are used to control the operation of conditional jump instructions.

Table 6-6. Condition Codes

Binary	Mnemonic	Description	Flags Set
0000	F	Always false	–
1000	T	Always true	–
0111 (1)	C	Carry	C = 1
1111 (1)	NC	No carry	C = 0
0110 (1)	Z	Zero	Z = 1
1110 (1)	NZ	Not zero	Z = 0
1101	PL	Plus	S = 0
0101	MI	Minus	S = 1
0100	OV	Overflow	V = 1
1100	NOV	No overflow	V = 0
0110 (1)	EQ	Equal	Z = 1
1110 (1)	NE	Not equal	Z = 0
1001	GE	Greater than or equal	(S XOR V) = 0
0001	LT	Less than	(S XOR V) = 1
1010	GT	Greater than	(Z OR (S XOR V)) = 0
0010	LE	Less than or equal	(Z OR (S XOR V)) = 1
1111 (1)	UGE	Unsigned greater than or equal	C = 0
0111 (1)	ULT	Unsigned less than	C = 1
1011	UGT	Unsigned greater than	(C = 0 AND Z = 0) = 1
0011	ULE	Unsigned less than or equal	(C OR Z) = 1

NOTES:

1. Indicate condition codes that are related to two different mnemonics but which test the same flag. For example, Z and EQ are both true if the zero flag (Z) is set, but after an ADD instruction, Z would probably be used; after a CP instruction, however, EQ would probably be used.
2. For operations involving unsigned numbers, the special condition codes UGE, ULT, UGT, and ULE must be used.

INSTRUCTION DESCRIPTIONS

This section contains detailed information and programming examples for each instruction in the SAM88RCRI instruction set. Information is arranged in a consistent format for improved readability and for fast referencing. The following information is included in each instruction description:

— Instruction name (mnemonic)
— Full instruction name
— Source/destination format of the instruction operand
— Shorthand notation of the instruction's operation
— Textual description of the instruction's effect
— Specific flag settings affected by the instruction
— Detailed description of the instruction's format, execution time, and addressing mode(s)
— Programming example(s) explaining how to use the instruction

ADC — Add With Carry

ADC dst,src

Operation: dst ¨ dst + src + c

The source operand, along with the setting of the carry flag, is added to the destination operand and the sum is stored in the destination. The contents of the source are unaffected. Two's-complement addition is performed. In multiple precision arithmetic, this instruction permits the carry from the addition of low-order operands to be carried into the addition of high-order operands.

Flags:

C: Set if there is a carry from the most significant bit of the result; cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurs, that is, if both operands are of the same sign and the result is of the opposite sign; cleared otherwise.
D: Always cleared to "0".
H: Set if there is a carry from the most significant bit of the low-order four bits of the result; cleared otherwise.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	dst | src		2	4	12	r	r
				6	13	r	lr
opc	src	dst	3	6	14	R	R
				6	15	R	IR
opc	dst	src	3	6	16	R	IM

Examples: Given: R1 = 10H, R2 = 03H, C flag = "1", register 01H = 20H, register 02H = 03H, and register 03H = 0AH:

ADC	R1,R2	→	R1 = 14H, R2 = 03H
ADC	R1,@R2	→	R1 = 1BH, R2 = 03H
ADC	01H,02H	→	Register 01H = 24H, register 02H = 03H
ADC	01H,@02H	→	Register 01H = 2BH, register 02H = 03H
ADC	01H,#11H	→	Register 01H = 32H

In the first example, destination register R1 contains the value 10H, the carry flag is set to "1", and the source working register R2 contains the value 03H. The statement "ADC R1,R2" adds 03H and the carry flag value ("1") to the destination value 10H, leaving 14H in register R1.

ADD — Add

ADD dst,src

Operation: dst ¨ dst + src

The source operand is added to the destination operand and the sum is stored in the destination. The contents of the source are unaffected. Two's-complement addition is performed.

Flags:
C: Set if there is a carry from the most significant bit of the result; cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if both operands are of the same sign and the result is of the opposite sign; cleared otherwise.
D: Always cleared to "0".
H: Set if a carry from the low-order nibble occurred.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc | dst | src	2	4	02	r	r
		6	03	r	lr
opc | src | dst	3	6	04	R	R
		6	05	R	IR
opc | dst | src	3	6	06	R	IM

Examples: Given: R1 = 12H, R2 = 03H, register 01H = 21H, register 02H = 03H, register 03H = 0AH:

ADD	R1,R2	→	R1 = 15H, R2 = 03H
ADD	R1,@R2	→	R1 = 1CH, R2 = 03H
ADD	01H,02H	→	Register 01H = 24H, register 02H = 03H
ADD	01H,@02H	→	Register 01H = 2BH, register 02H = 03H
ADD	01H,#25H	→	Register 01H = 46H

In the first example, destination working register R1 contains 12H and the source working register R2 contains 03H. The statement "ADD R1,R2" adds 03H to 12H, leaving the value 15H in register R1.

AND — Logical AND

AND dst,src

Operation: dst ¨ dst AND src

The source operand is logically ANDed with the destination operand. The result is stored in the destination. The AND operation results in a "1" bit being stored whenever the corresponding bits in the two operands are both logic ones; otherwise a "0" bit value is stored. The contents of the source are unaffected.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always cleared to "0".
D: Unaffected.
H: Unaffected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc	dst | src		2	4	52	r	r
				6	53	r	lr
opc	src	dst	3	6	54	R	R
				6	55	R	IR
opc	dst	src	3	6	56	R	IM

Examples: Given: R1 = 12H, R2 = 03H, register 01H = 21H, register 02H = 03H, register 03H = 0AH:

AND	R1,R2	→	R1 = 02H, R2 = 03H
AND	R1,@R2	→	R1 = 02H, R2 = 03H
AND	01H,02H	→	Register 01H = 01H, register 02H = 03H
AND	01H,@02H	→	Register 01H = 00H, register 02H = 03H
AND	01H,#25H	→	Register 01H = 21H

In the first example, destination working register R1 contains the value 12H and the source working register R2 contains 03H. The statement "AND R1,R2" logically ANDs the source operand 03H with the destination operand value 12H, leaving the value 02H in register R1.

CALL — Call Procedure

CALL dst

Operation:

SP ¨ SP – 1
@SP ¨ PCL
SP ¨ SP –1
@SP ¨ PCH
PC ¨ dst

The current contents of the program counter are pushed onto the top of the stack. The program counter value used is the address of the first instruction following the CALL instruction. The specified destination address is then loaded into the program counter and points to the first instruction of a procedure. At the end of the procedure the return instruction (RET) can be used to return to the original program flow. RET pops the top of the stack back into the program counter.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	3	14	F6	DA
opc dst	2	12	F4	IRR

Examples: Given: R0 = 15H, R1 = 21H, PC = 1A47H, and SP = 0B2H:

CALL 1521H → SP = 0B0H
(Memory locations 00H = 1AH, 01H = 4AH, where 4AH is the address that follows the instruction.)

CALL @RR0 → SP = 0B0H (00H = 1AH, 01H = 49H)

In the first example, if the program counter value is 1A47H and the stack pointer contains the value 0B2H, the statement "CALL 1521H" pushes the current PC value onto the top of the stack. The stack pointer now points to memory location 00H. The PC is then loaded with the value 1521H, the address of the first instruction in the program sequence to be executed.

If the contents of the program counter and stack pointer are the same as in the first example, the statement "CALL @RR0" produces the same result except that the 49H is stored in stack location 01H (because the two-byte instruction format was used). The PC is then loaded with the value 1521H, the address of the first instruction in the program sequence to be executed.

CCF — Complement Carry Flag

CCF

Operation: C ¨ NOT C

The carry flag (C) is complemented. If C = "1", the value of the carry flag is changed to logic zero; if C = "0", the value of the carry flag is changed to logic one.

Flags: **C:** Complemented.

No other flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	EF

Example: Given: The carry flag = "0":

CCF

If the carry flag = "0", the CCF instruction complements it in the FLAGS register (0D5H), changing its value from logic zero to logic one.

CLR — Clear

CLR dst

Operation: dst ¨ "0"

The destination location is cleared to "0".

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc | dst	2	4	B0	R
		4	B1	IR

Examples: Given: Register 00H = 4FH, register 01H = 02H, and register 02H = 5EH:

```
CLR   00H    →   Register 00H = 00H
CLR   @01H   →   Register 01H = 02H, register 02H = 00H
```

In Register (R) addressing mode, the statement "CLR 00H" clears the destination register 00H value to 00H. In the second example, the statement "CLR @01H" uses Indirect Register (IR) addressing mode to clear the 02H register value to 00H.

COM — Complement

COM dst

Operation: dst ¨ NOT dst

The contents of the destination location are complemented (one's complement); all "1s" are changed to "0s", and vice-versa.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always reset to "0".
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc | dst	2	4	60	R
		4	61	IR

Examples: Given: R1 = 07H and register 07H = 0F1H:

```
COM     R1      →     R1 = 0F8H
COM     @R1     →     R1 = 07H, register 07H = 0EH
```

In the first example, destination working register R1 contains the value 07H (00000111B). The statement "COM R1" complements all the bits in R1: all logic ones are changed to logic zeros, and vice-versa, leaving the value 0F8H (11111000B).

In the second example, Indirect Register (IR) addressing mode is used to complement the value of destination register 07H (11110001B), leaving the new value 0EH (00001110B).

CP — Compare

CP dst,src

Operation: dst – src

The source operand is compared to (subtracted from) the destination operand, and the appropriate flags are set accordingly. The contents of both operands are unaffected by the comparison.

Flags:
C: Set if a "borrow" occurred (src > dst); cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the operands were of opposite signs and the sign of the result is of the same as the sign of the source operand; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	dst | src		2	4	A2	r	r
				6	A3	r	lr
opc	src	dst	3	6	A4	R	R
				6	A5	R	IR
opc	dst	src	3	6	A6	R	IM

Examples: 1. Given: R1 = 02H and R2 = 03H:

```
CP      R1,R2      →      Set the C and S flags
```

Destination working register R1 contains the value 02H and source register R2 contains the value 03H. The statement "CP R1,R2" subtracts the R2 value (source/subtrahend) from the R1 value (destination/minuend). Because a "borrow" occurs and the difference is negative, C and S are "1".

2. Given: R1 = 05H and R2 = 0AH:

```
        CP      R1,R2
        JP      UGE,SKIP
        INC     R1
SKIP    LD      R3,R1
```

In this example, destination working register R1 contains the value 05H which is less than the contents of the source working register R2 (0AH). The statement "CP R1,R2" generates C = "1" and the JP instruction does not jump to the SKIP location. After the statement "LD R3,R1" executes, the value 06H remains in working register R3.

DEC — Decrement

DEC dst

Operation: dst ¨ dst – 1

The contents of the destination operand are decremented by one.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, dst value is –128(80H) and result value is +127(7FH); cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	00	R
		4	01	IR

Examples: Given: R1 = 03H and register 03H = 10H:

```
DEC   R1    →   R1 = 02H
DEC   @R1   →   Register 03H = 0FH
```

In the first example, if working register R1 contains the value 03H, the statement "DEC R1" decrements the hexadecimal value by one, leaving the value 02H. In the second example, the statement "DEC @R1" decrements the value 10H contained in the destination register 03H by one, leaving the value 0FH.

DI — Disable Interrupts

DI

Operation: SYM (2) ¨ 0

Bit zero of the system mode register, SYM.2, is cleared to "0", globally disabling all interrupt processing. Interrupt requests will continue to set their respective interrupt pending bits, but the CPU will not service them while interrupt processing is disabled.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	8F

Example: Given: SYM = 04H:

DI

If the value of the SYM register is 04H, the statement "DI" leaves the new value 00H in the register and clears SYM.2 to "0", disabling interrupt processing.

EI — Enable Interrupts

EI

Operation: SYM (2) ¨ 1

An EI instruction sets bit 2 of the system mode register, SYM.2 to "1". This allows interrupts to be serviced as they occur. If an interrupt's pending bit was set while interrupt processing was disabled (by executing a DI instruction), it will be serviced when you execute the EI instruction.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	9F

Example: Given: SYM = 00H:

EI

If the SYM register contains the value 00H, that is, if interrupts are currently disabled, the statement "EI" sets the SYM register to 04H, enabling all interrupts (SYM.2 is the enable bit for global interrupt processing).

IDLE — Idle Operation

IDLE

Operation:

The IDLE instruction stops the CPU clock while allowing system clock oscillation to continue. Idle mode can be released by an interrupt request (IRQ) or an external reset operation.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	1	4	6F	–	–

Example: The instruction

IDLE

stops the CPU clock but not the system clock.

INC — Increment

INC dst

Operation: dst ¨ dst + 1

The contents of the destination operand are incremented by one.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is dst value is +127(7FH) and result is –128(80H); cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
dst | opc	1	4	rE r = 0 to F	r
opc | dst	2	4	20	R
		4	21	IR

Examples: Given: R0 = 1BH, register 00H = 0CH, and register 1BH = 0FH:

```
INC   R0    →   R0 = 1CH
INC   00H   →   Register 00H = 0DH
INC   @R0   →   R0 = 1BH, register 01H = 10H
```

In the first example, if destination working register R0 contains the value 1BH, the statement "INC R0" leaves the value 1CH in that same register.

The next example shows the effect an INC instruction has on register 00H, assuming that it contains the value 0CH.

In the third example, INC is used in Indirect Register (IR) addressing mode to increment the value of register 1BH from 0FH to 10H.

IRET — Interrupt Return

IRET IRET

Operation:
FLAGS ¨ @SP
SP ¨ SP + 1
PC ¨ @SP
SP ¨ SP + 2
SYM(2) ¨ 1

This instruction is used at the end of an interrupt service routine. It restores the flag register and the program counter. It also re-enables global interrupts.

Flags: All flags are restored to their original settings (that is, the settings before the interrupt occurred).

Format:

IRET (Normal)	Bytes	Cycles	Opcode (Hex)
opc	1	6	BF

JP — Jump

JP cc,dst (Conditional)

JP dst (Unconditional)

Operation: If cc is true, PC ¨ dst

The conditional JUMP instruction transfers program control to the destination address if the condition specified by the condition code (cc) is true; otherwise, the instruction following the JP instruction is executed. The unconditional JP simply replaces the contents of the PC with the contents of the specified register pair. Control then passes to the statement addressed by the PC.

Flags: No flags are affected.

Format: (1)

(2)	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
cc | opc / dst	3	8 (3)	ccD cc = 0 to F	DA
opc / dst	2	8	30	IRR

NOTES:

1. The 3-byte format is used for a conditional jump and the 2-byte format for an unconditional jump.
2. In the first byte of the three-byte instruction format (conditional jump), the condition code and the opcode are both four bits.

Examples: Given: The carry flag (C) = "1", register 00 = 01H, and register 01 = 20H:

```
JP   C,LABEL_W  →   LABEL_W = 1000H, PC = 1000H
JP   @00H       →   PC = 0120H
```

The first example shows a conditional JP. Assuming that the carry flag is set to "1", the statement "JP C,LABEL_W" replaces the contents of the PC with the value 1000H and transfers control to that location. Had the carry flag not been set, control would then have passed to the statement immediately following the JP instruction.

The second example shows an unconditional JP. The statement "JP @00" replaces the contents of the PC with the contents of the register pair 00H and 01H, leaving the value 0120H.

JR — Jump Relative

JR cc,dst

Operation: If cc is true, PC ¨ PC + dst

If the condition specified by the condition code (cc) is true, the relative address is added to the program counter and control passes to the statement whose address is now in the program counter; otherwise, the instruction following the JR instruction is executed (See list of condition codes).

The range of the relative address is +127, –128, and the original value of the program counter is taken to be the address of the first instruction byte following the JR statement.

Flags: No flags are affected.

Format:

(1)

cc | opc	dst

Bytes	Cycles	Opcode (Hex)	Addr Mode dst
2	6 (2)	ccB	RA
		cc = 0 to F	

NOTE: In the first byte of the two-byte instruction format, the condition code and the opcode are each four bits.

Example: Given: The carry flag = "1" and LABEL_X = 1FF7H:

JR C,LABEL_X → PC = 1FF7H

If the carry flag is set (that is, if the condition code is true), the statement "JR C,LABEL_X" will pass control to the statement whose address is now in the PC. Otherwise, the program instruction following the JR would be executed.

LD — Load

LD dst,src

Operation: dst ¨ src

The contents of the source are loaded into the destination. The source's contents are unaffected.

Flags: No flags are affected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
dst | opc	src		2	4	rC	r	IM
				4	r8	r	R
src | opc	dst		2	4	r9	R	r
					r = 0 to F		
opc	dst | src		2	4	C7	r	lr
				4	D7	lr	r
opc	src	dst	3	6	E4	R	R
				6	E5	R	IR
opc	dst	src	3	6	E6	R	IM
				6	D6	IR	IM
opc	src	dst	3	6	F5	IR	R
opc	dst | src	x	3	6	87	r	x [r]
opc	src | dst	x	3	6	97	x [r]	r

LD — Load

LD (Continued)

Examples: Given: R0 = 01H, R1 = 0AH, register 00H = 01H, register 01H = 20H, register 02H = 02H, LOOP = 30H, and register 3AH = 0FFH:

LD	R0,#10H	→	R0 = 10H
LD	R0,01H	→	R0 = 20H, register 01H = 20H
LD	01H,R0	→	Register 01H = 01H, R0 = 01H
LD	R1,@R0	→	R1 = 20H, R0 = 01H
LD	@R0,R1	→	R0 = 01H, R1 = 0AH, register 01H = 0AH
LD	00H,01H	→	Register 00H = 20H, register 01H = 20H
LD	02H,@00H	→	Register 02H = 20H, register 00H = 01H
LD	00H,#0AH	→	Register 00H = 0AH
LD	@00H,#10H	→	Register 00H = 01H, register 01H = 10H
LD	@00H,02H	→	Register 00H = 01H, register 01H = 02, register 02H = 02H
LD	R0,#LOOP[R1]	→	R0 = 0FFH, R1 = 0AH
LD	#LOOP[R0],R1	→	Register 31H = 0AH, R0 = 01H, R1 = 0AH

LDC/LDE — Load Memory

LDC/LDE dst,src

Operation: dst ¨ src

This instruction loads a byte from program or data memory into a working register or vice-versa. The source values are unaffected. LDC refers to program memory and LDE to data memory. The assembler makes 'Irr' or 'rr' values an even number for program memory and an odd number for data memory.

Flags: No flags are affected.

Format:

					Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
1.	opc	dst | src			2	10	C3	r	Irr
2.	opc	src | dst			2	10	D3	Irr	r
3.	opc	dst | src	XS		3	12	E7	r	XS [rr]
4.	opc	src | dst	XS		3	12	F7	XS [rr]	r
5.	opc	dst | src	XL_L	XL_H	4	14	A7	r	XL [rr]
6.	opc	src | dst	XL_L	XL_H	4	14	B7	XL [rr]	r
7.	opc	dst | 0000	DA_L	DA_H	4	14	A7	r	DA
8.	opc	src | 0000	DA_L	DA_H	4	14	B7	DA	r
9.	opc	dst | 0001	DA_L	DA_H	4	14	A7	r	DA
10.	opc	src | 0001	DA_L	DA_H	4	14	B7	DA	r

NOTES:

1. The source (src) or working register pair [rr] for formats 5 and 6 cannot use register pair 0–1.
2. For formats 3 and 4, the destination address 'XS [rr]' and the source address 'XS [rr]' are each one byte.
3. For formats 5 and 6, the destination address 'XL [rr] and the source address 'XL [rr]' are each two bytes.
4. The DA and r source values for formats 7 and 8 are used to address program memory; the second set of values, used in formats 9 and 10, are used to address data memory.

LDC/LDE — Load Memory

LDC/LDE (Continued)

Examples: Given: R0 = 11H, R1 = 34H, R2 = 01H, R3 = 04H, R4 = 00H, R5 = 60H; Program memory locations 0061 = AAH, 0103H = 4FH, 0104H = 1A, 0105H = 6DH, and 1104H = 88H. External data memory locations 0061H = BBH, 0103H = 5FH, 0104H = 2AH, 0105H = 7DH, and 1104H = 98H:

```
LDC     R0,@RR2          ; R0 ¨ contents of program memory location 0104H
                         ; R0 = 1AH, R2 = 01H, R3 = 04H
LDE     R0,@RR2          ; R0 ¨ contents of external data memory location 0104H
                         ; R0 = 2AH, R2 = 01H, R3 = 04H
LDC *   @RR2,R0          ; 11H (contents of R0) is loaded into program memory
                         ; location 0104H (RR2),
                         ; working registers R0, R2, R3 Æ no change
LDE     @RR2,R0          ; 11H (contents of R0) is loaded into external data memory
                         ; location 0104H (RR2),
                         ; working registers R0, R2, R3 Æ no change
LDC     R0,#01H[RR4]     ; R0 ¨ contents of program memory location 0061H
                         ; (01H + RR4),
                         ; R0 = AAH, R2 = 00H, R3 = 60H
LDE     R0,#01H[RR4]     ; R0 ¨ contents of external data memory location 0061H
                         ; (01H + RR4), R0 = BBH, R4 = 00H, R5 = 60H
LDC (note)  #01H[RR4],R0 ; 11H (contents of R0) is loaded into program memory
                         ; location 0061H (01H + 0060H)
LDE     #01H[RR4],R0     ; 11H (contents of R0) is loaded into external data memory
                         ; location 0061H (01H + 0060H)
LDC     R0,#1000H[RR2]   ; R0 ¨ contents of program memory location 1104H
                         ; (1000H + 0104H), R0 = 88H, R2 = 01H, R3 = 04H
LDE     R0,#1000H[RR2]   ; R0 ¨ contents of external data memory location 1104H
                         ; (1000H + 0104H), R0 = 98H, R2 = 01H, R3 = 04H
LDC     R0,1104H         ; R0 ¨ contents of program memory location 1104H,
                         ; R0 = 88H
LDE     R0,1104H         ; R0 ¨ contents of external data memory location 1104H,
                         ; R0 = 98H
LDC (note)  1105H,R0     ; 11H (contents of R0) is loaded into program memory
                         ; location 1105H, (1105H) ¨ 11H
LDE     1105H,R0         ; 11H (contents of R0) is loaded into external data memory
                         ; location 1105H, (1105H) ¨ 11H
```

NOTE: These instructions are not supported by masked ROM type devices.

LDCD/LDED — Load Memory and Decrement

LDCD/LDED dst,src

Operation: dst ¨ src

rr ¨ rr – 1

These instructions are used for user stacks or block transfers of data from program or data memory to the register file. The address of the memory location is specified by a working register pair. The contents of the source location are loaded into the destination location. The memory address is then decremented. The contents of the source are unaffected.

LDCD references program memory and LDED references external data memory. The assembler makes 'Irr' an even number for program memory and an odd number for data memory.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc | dst | src	2	10	E2	r	Irr

Examples: Given: R6 = 10H, R7 = 33H, R8 = 12H, program memory location 1033H = 0CDH, and external data memory location 1033H = 0DDH:

```
LDCD    R8,@RR6     ; 0CDH (contents of program memory location 1033H) is
                    ; loaded into R8 and RR6 is decremented by one
                    ; R8 = 0CDH, R6 = 10H, R7 = 32H (RR6 ← RR6 - 1)
LDED    R8,@RR6     ; 0DDH (contents of data memory location 1033H) is
                    ; loaded into R8 and RR6 is decremented by one
                    ; (RR6 ← RR6 - 1) R8 = 0DDH, R6 = 10H, R7 = 32H
```

LDCI/LDEI — Load Memory and Increment

LDCI/LDEI dst,src

Operation: dst ¨ src

rr ¨ rr + 1

These instructions are used for user stacks or block transfers of data from program or data memory to the register file. The address of the memory location is specified by a working register pair. The contents of the source location are loaded into the destination location. The memory address is then incremented automatically. The contents of the source are unaffected.

LDCI refers to program memory and LDEI refers to external data memory. The assembler makes 'Irr' even for program memory and odd for data memory.

Flags: No flags are affected.

Format:

		Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	dst | src	2	10	E3	r	Irr

Examples: Given: R6 = 10H, R7 = 33H, R8 = 12H, program memory locations 1033H = 0CDH and 1034H = 0C5H; external data memory locations 1033H = 0DDH and 1034H = 0D5H:

```
LDCI    R8,@RR6     ; 0CDH (contents of program memory location 1033H) is
                    ; loaded into R8 and RR6 is incremented by one
                    ; (RR6 ¨ RR6 + 1) R8 = 0CDH, R6 = 10H, R7 = 34H
LDEI    R8,@RR6     ; 0DDH (contents of data memory location 1033H) is
                    ; loaded into R8 and RR6 is incremented by one
                    ; (RR6 ¨ RR6 + 1) R8 = 0DDH, R6 = 10H, R7 = 34H
```

NOP — No Operation

NOP

Operation: No action is performed when the CPU executes this instruction. Typically, one or more NOPs are executed in sequence in order to effect a timing delay of variable duration.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	FF

Example: When the instruction

NOP

is encountered in a program, no operation occurs. Instead, there is a delay in instruction execution time.

OR — Logical OR

OR dst,src

Operation: dst ¨ dst OR src

The source operand is logically ORed with the destination operand and the result is stored in the destination. The contents of the source are unaffected. The OR operation results in a "1" being stored whenever either of the corresponding bits in the two operands is a "1"; otherwise a "0" is stored.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always cleared to "0".
D: Unaffected.
H: Unaffected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc	dst | src		2	4	42	r	r
				6	43	r	lr
opc	src	dst	3	6	44	R	R
				6	45	R	IR
opc	dst	src	3	6	46	R	IM

Examples: Given: R0 = 15H, R1 = 2AH, R2 = 01H, register 00H = 08H, register 01H = 37H, and register 08H = 8AH:

OR	R0,R1	→	R0 = 3FH, R1 = 2AH
OR	R0,@R2	→	R0 = 37H, R2 = 01H, register 01H = 37H
OR	00H,01H	→	Register 00H = 3FH, register 01H = 37H
OR	01H,@00H	→	Register 00H = 08H, register 01H = 0BFH
OR	00H,#02H	→	Register 00H = 0AH

In the first example, if working register R0 contains the value 15H and register R1 the value 2AH, the statement "OR R0,R1" logical-ORs the R0 and R1 register contents and stores the result (3FH) in destination register R0.

The other examples show the use of the logical OR instruction with the various addressing modes and formats.

POP — Pop From Stack

POP dst

Operation: dst ¨ @SP

SP ¨ SP + 1

The contents of the location addressed by the stack pointer are loaded into the destination. The stack pointer is then incremented by one.

Flags: No flags affected.

Format:

		Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc	dst	2	8	50	R
			8	51	IR

Examples: Given: Register 00H = 01H, register 01H = 1BH, SP (0D9H) = 0BBH, and stack register 0BBH = 55H:

```
POP   00H    →   Register 00H = 55H, SP = 0BCH
POP   @00H   →   Register 00H = 01H, register 01H = 55H, SP = 0BCH
```

In the first example, general register 00H contains the value 01H. The statement "POP 00H" loads the contents of location 0BBH (55H) into destination register 00H and then increments the stack pointer by one. Register 00H then contains the value 55H and the SP points to location 0BCH.

PUSH — Push To Stack

PUSH src

Operation: SP ¨ SP – 1

@SP ¨ src

A PUSH instruction decrements the stack pointer value and loads the contents of the source (src) into the location addressed by the decremented stack pointer. The operation then adds the new value to the top of the stack.

Flags: No flags are affected.

Format:

		Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc	src	2	8	70	R
			8	71	IR

Examples: Given: Register 40H = 4FH, register 4FH = 0AAH, SP = 0C0H:

PUSH 40H → Register 40H = 4FH, stack register 0BFH = 4FH, SP = 0BFH

PUSH @40H → Register 40H = 4FH, register 4FH = 0AAH, stack register 0BFH = 0AAH, SP = 0BFH

In the first example, if the stack pointer contains the value 0C0H, and general register 40H the value 4FH, the statement "PUSH 40H" decrements the stack pointer from 0C0 to 0BFH. It then loads the contents of register 40H into location 0BFH. Register 0BFH then contains the value 4FH and SP points to location 0BFH.

RCF — Reset Carry Flag

RCF RCF

Operation: C ¨ 0

The carry flag is cleared to logic zero, regardless of its previous value.

Flags: **C:** Cleared to "0".

No other flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	CF

Example: Given: C = "1" or "0":

The instruction RCF clears the carry flag (C) to logic zero.

RET — Return

RET

Operation: PC ¨ @SP

SP ¨ SP + 2

The RET instruction is normally used to return to the previously executing procedure at the end of a procedure entered by a CALL instruction. The contents of the location addressed by the stack pointer are popped into the program counter. The next statement that is executed is the one that is addressed by the new program counter value.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	8	AF

Example: Given: SP = 0BCH, (SP) = 101AH, and PC = 1234:

RET → PC = 101AH, SP = 0BEH

The statement "RET" pops the contents of stack pointer location 0BCH (10H) into the high byte of the program counter. The stack pointer then pops the value in location 0BDH (1AH) into the PC's low byte and the instruction at location 101AH is executed. The stack pointer now points to memory location 0BEH.

RL — Rotate Left

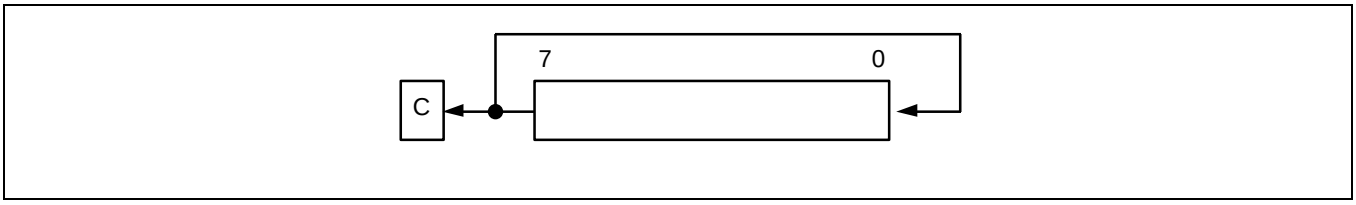
RL dst

Operation: C ¨ dst (7)

dst (0) ¨ dst (7)

dst (n + 1) ¨ dst (n), n = 0–6

The contents of the destination operand are rotated left one bit position. The initial value of bit 7 is moved to the bit zero (LSB) position and also replaces the carry flag.



Flags:
C: Set if the bit rotated from the most significant bit position (bit 7) was "1".
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the sign of the destination changed during rotation; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	90	R
		4	91	IR

Examples: Given: Register 00H = 0AAH, register 01H = 02H and register 02H = 17H:

```
RL    00H    →    Register 00H = 55H, C = "1"
RL    @01H   →    Register 01H = 02H, register 02H = 2EH, C = "0"
```

In the first example, if general register 00H contains the value 0AAH (10101010B), the statement "RL 00H" rotates the 0AAH value left one bit position, leaving the new value 55H (01010101B) and setting the carry and overflow flags.

RLC — Rotate Left Through Carry

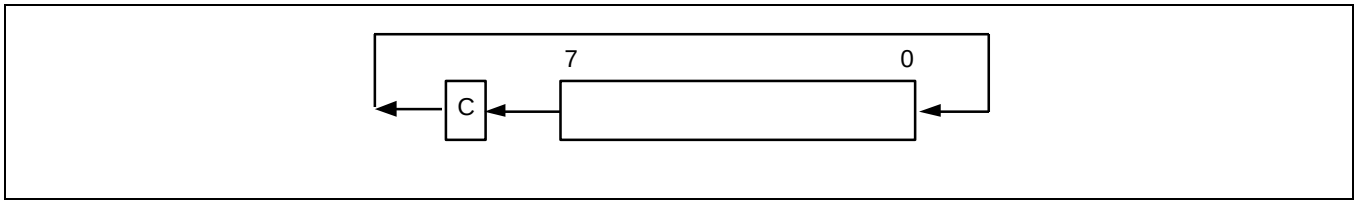
RLC dst

Operation: dst (0) ¨ C

C ¨ dst (7)

dst (n + 1) ¨ dst (n), n = 0–6

The contents of the destination operand with the carry flag are rotated left one bit position. The initial value of bit 7 replaces the carry flag (C); the initial value of the carry flag replaces bit zero.



Flags:
C: Set if the bit rotated from the most significant bit position (bit 7) was "1".
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the sign of the destination changed during rotation; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	10	R
		4	11	IR

Examples: Given: Register 00H = 0AAH, register 01H = 02H, and register 02H = 17H, C = "0":

```
RLC   00H    →   Register 00H = 54H, C = "1"
RLC   @01H   →   Register 01H = 02H, register 02H = 2EH, C = "0"
```

In the first example, if general register 00H has the value 0AAH (10101010B), the statement "RLC 00H" rotates 0AAH one bit position to the left. The initial value of bit 7 sets the carry flag and the initial value of the C flag replaces bit zero of register 00H, leaving the value 55H (01010101B). The MSB of register 00H resets the carry flag to "1" and sets the overflow flag.

RR — Rotate Right

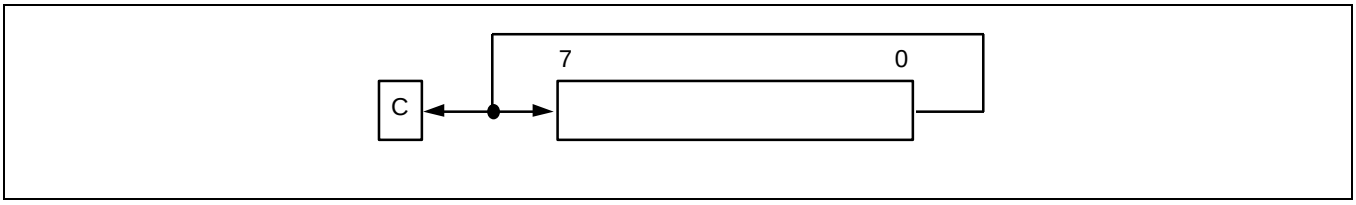
RR dst

Operation: C ¨ dst (0)

dst (7) ¨ dst (0)

dst (n) ¨ dst (n + 1), n = 0–6

The contents of the destination operand are rotated right one bit position. The initial value of bit zero (LSB) is moved to bit 7 (MSB) and also replaces the carry flag (C).



Flags:

C: Set if the bit rotated from the least significant bit position (bit zero) was "1".
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the sign of the destination changed during rotation; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	E0	R
		4	E1	IR

Examples: Given: Register 00H = 31H, register 01H = 02H, and register 02H = 17H:

```
RR    00H    →    Register 00H = 98H, C = "1"
RR    @01H   →    Register 01H = 02H, register 02H = 8BH, C = "1"
```

In the first example, if general register 00H contains the value 31H (00110001B), the statement "RR 00H" rotates this value one bit position to the right. The initial value of bit zero is moved to bit 7, leaving the new value 98H (10011000B) in the destination register. The initial bit zero also resets the C flag to "1" and the sign flag and overflow flag are also set to "1".

RRC — Rotate Right Through Carry

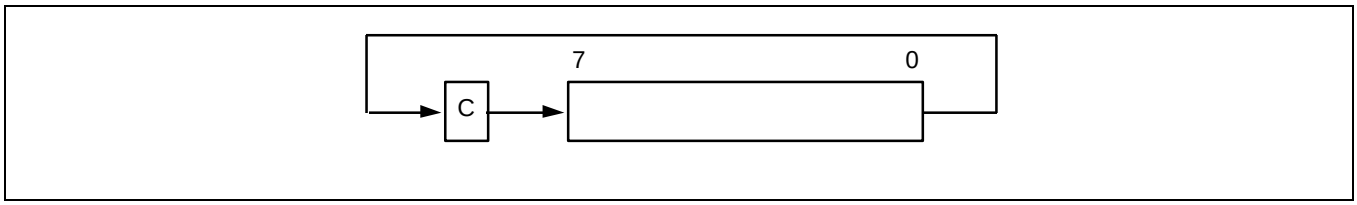
RRC dst

Operation: dst (7) ¨ C

C ¨ dst (0)

dst (n) ¨ dst (n + 1), n = 0–6

The contents of the destination operand and the carry flag are rotated right one bit position. The initial value of bit zero (LSB) replaces the carry flag; the initial value of the carry flag replaces bit 7 (MSB).



Flags:
C: Set if the bit rotated from the least significant bit position (bit zero) was "1".
Z: Set if the result is "0" cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the sign of the destination changed during rotation; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	C0	R
		4	C1	IR

Examples: Given: Register 00H = 55H, register 01H = 02H, register 02H = 17H, and C = "0":

```
RRC   00H    →   Register 00H = 2AH, C = "1"
RRC   @01H   →   Register 01H = 02H, register 02H = 0BH, C = "1"
```

In the first example, if general register 00H contains the value 55H (01010101B), the statement "RRC 00H" rotates this value one bit position to the right. The initial value of bit zero ("1") replaces the carry flag and the initial value of the C flag ("1") replaces bit 7. This leaves the new value 2AH (00101010B) in destination register 00H. The sign flag and overflow flag are both cleared to "0".

SBC — Subtract With Carry

SBC dst,src

Operation: dst ¨ dst – src – c

The source operand, along with the current value of the carry flag, is subtracted from the destination operand and the result is stored in the destination. The contents of the source are unaffected. Subtraction is performed by adding the two's-complement of the source operand to the destination operand. In multiple precision arithmetic, this instruction permits the carry ("borrow") from the subtraction of the low-order operands to be subtracted from the subtraction of high-order operands.

Flags:

C: Set if a borrow occurred (src > dst); cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the operands were of opposite sign and the sign

f the result is the same as the sign of the source; cleared otherwise.
D: Always set to "1".
H: Cleared if there is a carry from the most significant bit of the low-order four bits of the result; set otherwise, indicating a "borrow".

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc	dst | src		2	4	32	r	r
				6	33	r	lr
opc	src	dst	3	6	34	R	R
				6	35	R	IR
opc	dst	src	3	6	36	R	IM

Examples: Given: R1 = 10H, R2 = 03H, C = "1", register 01H = 20H, register 02H = 03H, and register 03H = 0AH:

SBC	R1,R2	→	R1 = 0CH, R2 = 03H
SBC	R1,@R2	→	R1 = 05H, R2 = 03H, register 03H = 0AH
SBC	01H,02H	→	Register 01H = 1CH, register 02H = 03H
SBC	01H,@02H	→	Register 01H = 15H,register 02H = 03H, register 03H = 0AH
SBC	01H,#8AH	→	Register 01H = 95H; C, S, and V = "1"

In the first example, if working register R1 contains the value 10H and register R2 the value 03H, the statement "SBC R1,R2" subtracts the source value (03H) and the C flag value ("1") from the destination (10H) and then stores the result (0CH) in register R1.

SCF — Set Carry Flag

SCF

Operation: C ¨ 1

The carry flag (C) is set to logic one, regardless of its previous value.

Flags: **C:** Set to "1".

No other flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	DF

Example: The statement

SCF

sets the carry flag to logic one.

SRA — Shift Right Arithmetic

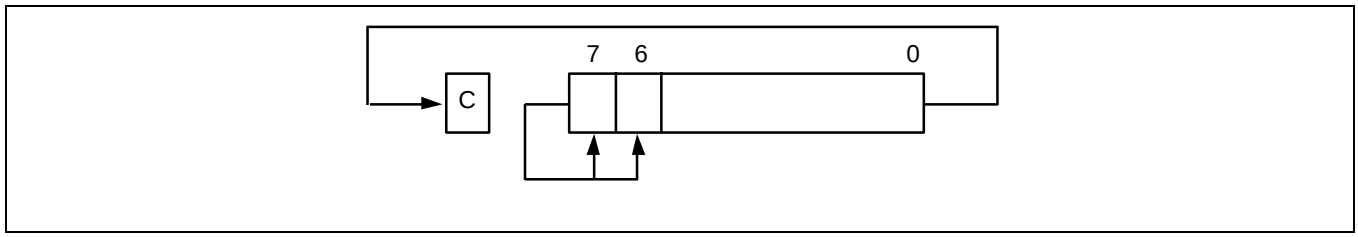
SRA dst

Operation: dst (7) ¨ dst (7)

C ¨ dst (0)

dst (n) ¨ dst (n + 1), n = 0–6

An arithmetic shift-right of one bit position is performed on the destination operand. Bit zero (the LSB) replaces the carry flag. The value of bit 7 (the sign bit) is unchanged and is shifted into bit position 6.



Flags:
C: Set if the bit shifted from the LSB position (bit zero) was "1".
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Always cleared to "0".
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	D0	R
		4	D1	IR

Examples: Given: Register 00H = 9AH, register 02H = 03H, register 03H = 0BCH, and C = "1":

```
SRA   00H    →   Register 00H = 0CD, C = "0"
SRA   @02H   →   Register 02H = 03H, register 03H = 0DEH, C = "0"
```

In the first example, if general register 00H contains the value 9AH (10011010B), the statement "SRA 00H" shifts the bit values in register 00H right one bit position. Bit zero ("0") clears the C flag and bit 7 ("1") is then shifted into the bit 6 position (bit 7 remains unchanged). This leaves the value 0CDH (11001101B) in destination register 00H.

STOP — Stop Operation

STOP

Operation:

The STOP instruction stops both the CPU clock and system clock and causes the microcontroller to enter Stop mode. During Stop mode, the contents of on-chip CPU registers, peripheral registers, and I/O port control and data registers are retained. Stop mode can be released by an external reset operation or External interrupt input. For the reset operation, the RESET pin must be held to Low level until the required oscillation stabilization interval has elapsed.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	1	4	7F	–	–

Example: The statement

STOP

halts all microcontroller operations.

SUB — Subtract

SUB dst,src

Operation: dst ¨ dst – src

The source operand is subtracted from the destination operand and the result is stored in the destination. The contents of the source are unaffected. Subtraction is performed by adding the two's complement of the source operand to the destination operand.

Flags:
C: Set if a "borrow" occurred; cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the operands were of opposite signs and the sign of the result is of the same as the sign of the source operand; cleared otherwise.
D: Always set to "1".
H: Cleared if there is a carry from the most significant bit of the low-order four bits of the result; set otherwise indicating a "borrow".

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	dst | src		2	4	22	r	r
				6	23	r	lr
opc	src	dst	3	6	24	R	R
				6	25	R	IR
opc	dst	src	3	6	26	R	IM

Examples: Given: R1 = 12H, R2 = 03H, register 01H = 21H, register 02H = 03H, register 03H = 0AH:

SUB	R1,R2	→	R1 = 0FH, R2 = 03H
SUB	R1,@R2	→	R1 = 08H, R2 = 03H
SUB	01H,02H	→	Register 01H = 1EH, register 02H = 03H
SUB	01H,@02H	→	Register 01H = 17H, register 02H = 03H
SUB	01H,#90H	→	Register 01H = 91H; C, S, and V = "1"
SUB	01H,#65H	→	Register 01H = 0BCH; C and S = "1", V = "0"

In the first example, if working register R1 contains the value 12H and if register R2 contains the value 03H, the statement "SUB R1,R2" subtracts the source value (03H) from the destination value (12H) and stores the result (0FH) in destination register R1.

TCM — Test Complement Under Mask

TCM dst,src

Operation: (NOT dst) AND src

This instruction tests selected bits in the destination operand for a logic one value. The bits to be tested are specified by setting a "1" bit in the corresponding position of the source operand (mask). The TCM statement complements the destination operand, which is then ANDed with the source mask. The zero (Z) flag can then be checked to determine the result. The destination and source operands are unaffected.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always cleared to "0".
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc | dst | src	2	4	62	r	r
		6	63	r	lr
opc | src | dst	3	6	64	R	R
		6	65	R	IR
opc | dst | src	3	6	66	R	IM

Examples: Given: R0 = 0C7H, R1 = 02H, R2 = 12H, register 00H = 2BH, register 01H = 02H, and register 02H = 23H:

TCM	R0,R1	→	R0 = 0C7H, R1 = 02H, Z = "1"
TCM	R0,@R1	→	R0 = 0C7H, R1 = 02H, register 02H = 23H, Z = "0"
TCM	00H,01H	→	Register 00H = 2BH, register 01H = 02H, Z = "1"
TCM	00H,@01H	→	Register 00H = 2BH, register 01H = 02H, register 02H = 23H, Z = "1"
TCM	00H,#34	→	Register 00H = 2BH, Z = "0"

In the first example, if working register R0 contains the value 0C7H (11000111B) and register R1 the value 02H (00000010B), the statement "TCM R0,R1" tests bit one in the destination register for a "1" value. Because the mask value corresponds to the test bit, the Z flag is set to logic one and can be tested to determine the result of the TCM operation.

TM — Test Under Mask

TM dst,src

Operation: dst AND src

This instruction tests selected bits in the destination operand for a logic zero value. The bits to be tested are specified by setting a "1" bit in the corresponding position of the source operand (mask), which is ANDed with the destination operand. The zero (Z) flag can then be checked to determine the result. The destination and source operands are unaffected.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always reset to "0".
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc | dst | src	2	4	72	r	r
		6	73	r	lr
opc | src | dst	3	6	74	R	R
		6	75	R	IR
opc | dst | src	3	6	76	R	IM

Examples: Given: R0 = 0C7H, R1 = 02H, R2 = 18H, register 00H = 2BH, register 01H = 02H, and register 02H = 23H:

```
TM    R0,R1      →   R0 = 0C7H, R1 = 02H, Z = "0"
TM    R0,@R1     →   R0 = 0C7H, R1 = 02H, register 02H = 23H, Z = "0"
TM    00H,01H    →   Register 00H = 2BH, register 01H = 02H, Z = "0"
TM    00H,@01H   →   Register 00H = 2BH, register 01H = 02H,
                     register 02H = 23H, Z = "0"
TM    00H,#54H   →   Register 00H = 2BH, Z = "1"
```

In the first example, if working register R0 contains the value 0C7H (11000111B) and register R1 the value 02H (00000010B), the statement "TM R0,R1" tests bit one in the destination register for a "0" value. Because the mask value does not match the test bit, the Z flag is cleared to logic zero and can be tested to determine the result of the TM operation.

XOR — Logical Exclusive OR

XOR dst,src

Operation: dst ¨ dst XOR src

The source operand is logically exclusive-ORed with the destination operand and the result is stored in the destination. The exclusive-OR operation results in a "1" bit being stored whenever the corresponding bits in the operands are different; otherwise, a "0" bit is stored.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always reset to "0".
D: Unaffected.
H: Unaffected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc	dst | src		2	4	B2	r	r
				6	B3	r	lr
opc	src	dst	3	6	B4	R	R
				6	B5	R	IR
opc	dst	src	3	6	B6	R	IM

Examples: Given: R0 = 0C7H, R1 = 02H, R2 = 18H, register 00H = 2BH, register 01H = 02H, and register 02H = 23H:

XOR	R0,R1	→	R0 = 0C5H, R1 = 02H
XOR	R0,@R1	→	R0 = 0E4H, R1 = 02H, register 02H = 23H
XOR	00H,01H	→	Register 00H = 29H, register 01H = 02H
XOR	00H,@01H	→	Register 00H = 08H, register 01H = 02H, register 02H = 23H
XOR	00H,#54H	→	Register 00H = 7FH

In the first example, if working register R0 contains the value 0C7H and if register R1 contains the value 02H, the statement "XOR R0,R1" logically exclusive-ORs the R1 value with the R0 value and stores the result (0C5H) in the destination register R0.

7 CLOCK CIRCUITS

OVERVIEW

The S3C9228 microcontroller has two oscillator circuits: a main clock, and a sub clock circuit. The CPU and peripheral hardware operate on the system clock frequency supplied through these circuits. The maximum CPU clock frequency, is determined by CLKCON register settings.

SYSTEM CLOCK CIRCUIT

The system clock circuit has the following components:

— Crystal, ceramic resonator, RC oscillation source (main clock only), or an external clock
— Oscillator stop and wake-up functions
— Programmable frequency divider for the CPU clock (fxx divided by 1, 2, 8, or 16)
— Clock circuit control register, CLKCON
— Oscillator control register, OSCCON

CPU CLOCK NOTATION

In this document, the following notation is used for descriptions of the CPU clock:

fx main clock
fxt sub clock
fxx selected system clock

MAIN OSCILLATOR CIRCUITS

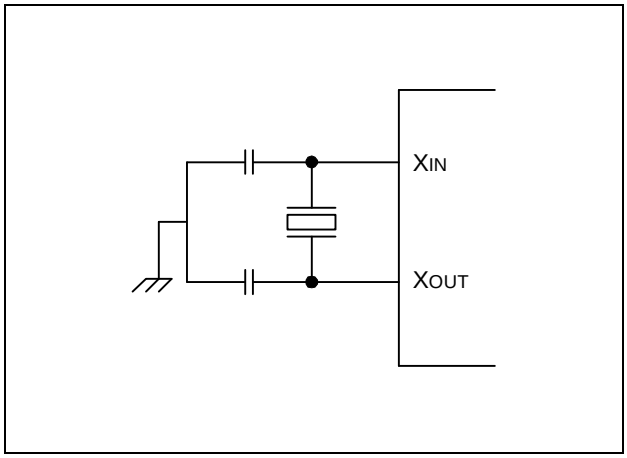


Figure 7-1. Crystal/Ceramic Oscillator

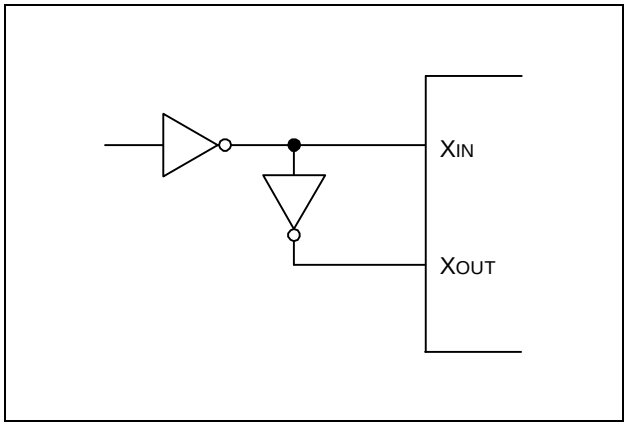


Figure 7-2. External Oscillator

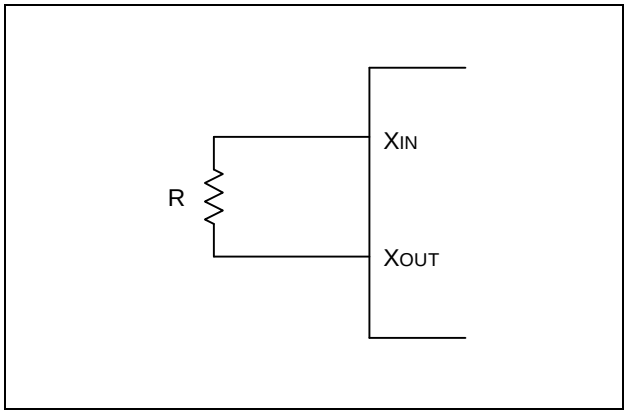


Figure 7-3. RC Oscillator

SUB OSCILLATOR CIRCUITS

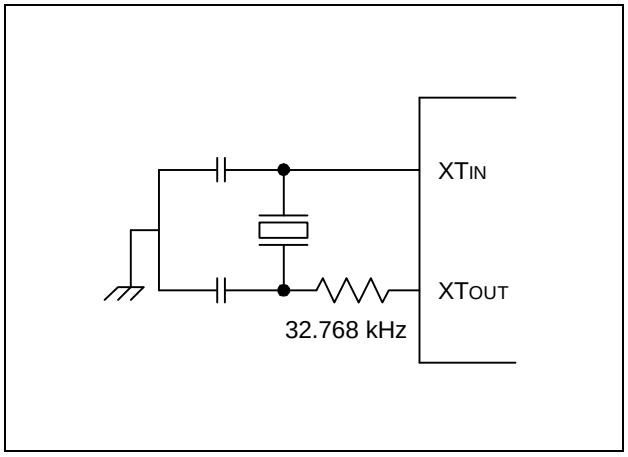


Figure 7-4. Crystal/Ceramic Oscillator

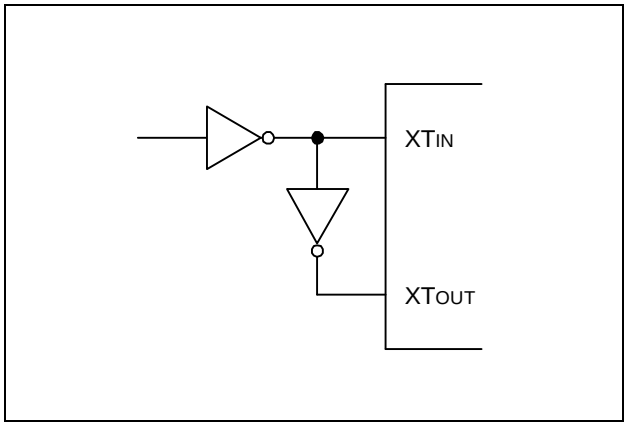


Figure 7-5. External Oscillator

CLOCK STATUS DURING POWER-DOWN MODES

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

— In Stop mode, the main oscillator is halted. Stop mode is released, and the oscillator started, by a reset operation, by an external interrupt, or by an internal interrupt if sub clock is selected as the clock source (When the fx is selected as system clock).

— In Idle mode, the internal clock signal is gated away from the CPU, but continues to be supplied to the interrupt structure, timer A/B, and watch timer. Idle mode is released by a reset or by an external or internal interrupts.

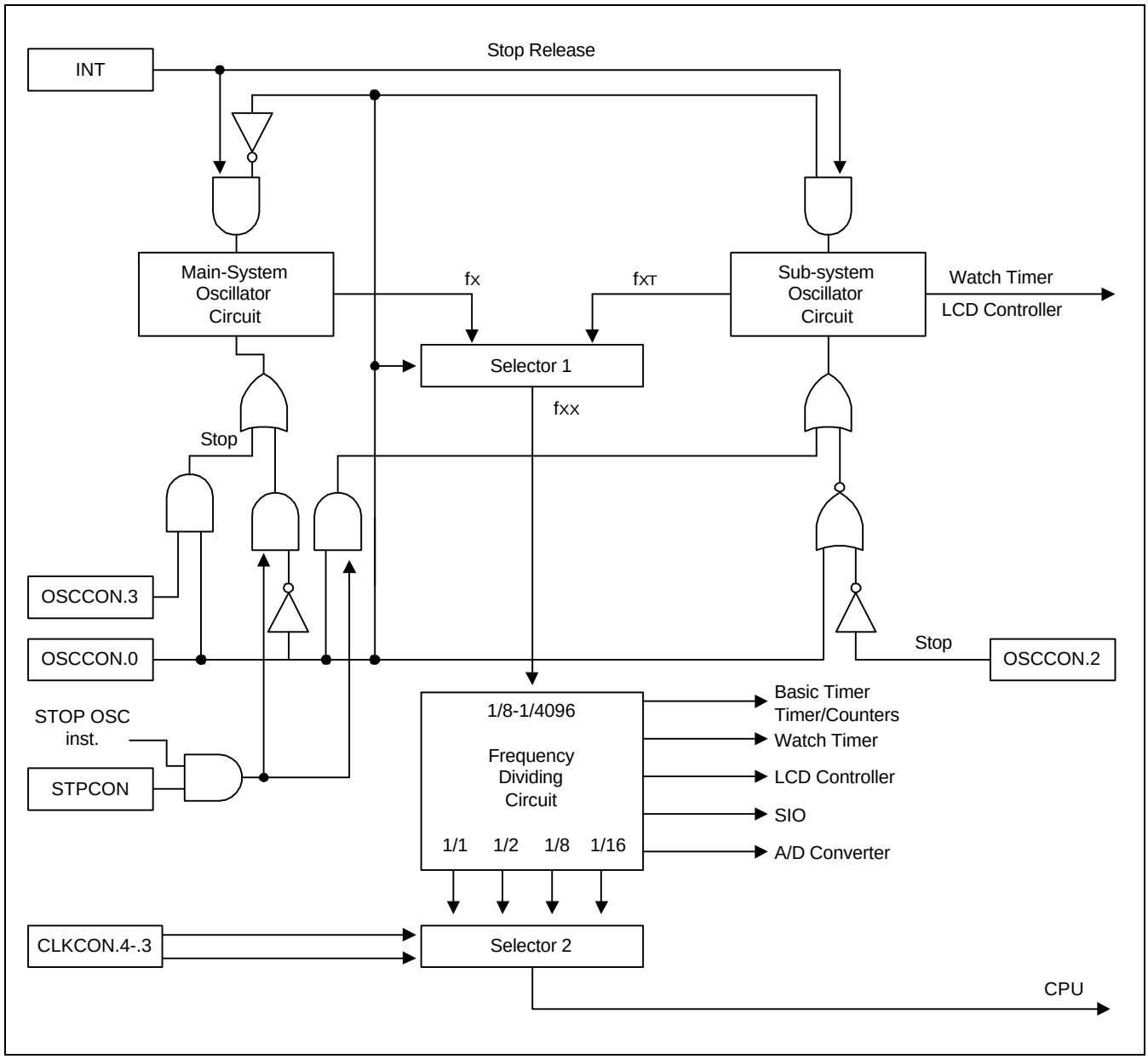


Figure 7-6. System Clock Circuit Diagram

SYSTEM CLOCK CONTROL REGISTER (CLKCON)

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

— Oscillator IRQ wake-up function enable/disable

— Oscillator frequency divide-by value

CLKCON register settings control 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 fx/16 (the slowest clock speed) is selected as the CPU clock. If necessary, you can then increase the CPU clock speed to fx, fx/2, or fx/8 by setting the CLKCON, and you can change system clock from main clock to sub clock by setting the OSCCON.

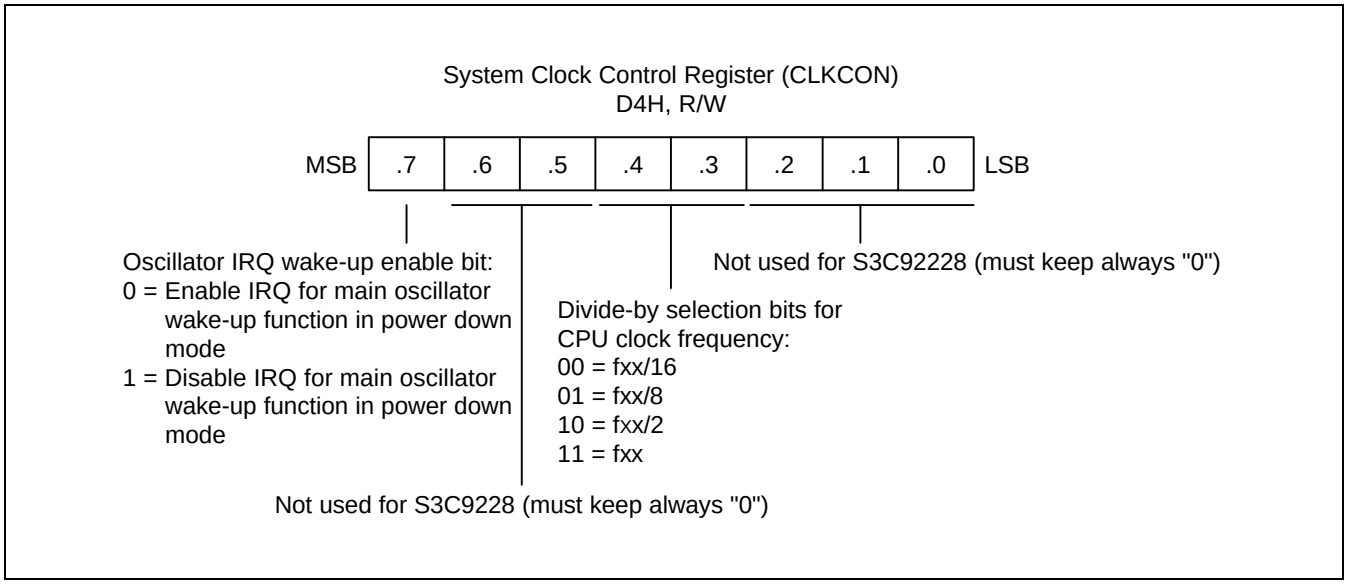


Figure 7-7. System Clock Control Register (CLKCON)

OSCILLATOR CONTROL REGISTER (OSCCON)

The oscillator control register, OSCCON, is located in address D3H. It is read/write addressable and has the following functions:

— System clock selection
— Main oscillator control
— Sub oscillator control

OSCCON.0 register settings select Main clock or Sub clock as system clock.
After a reset, Main clock is selected for system clock because the reset value of OSCCON.0 is "0".

The main oscillator can be stopped or run by setting OSCCON.3.

The sub oscillator can be stopped or run by setting OSCCON.2.

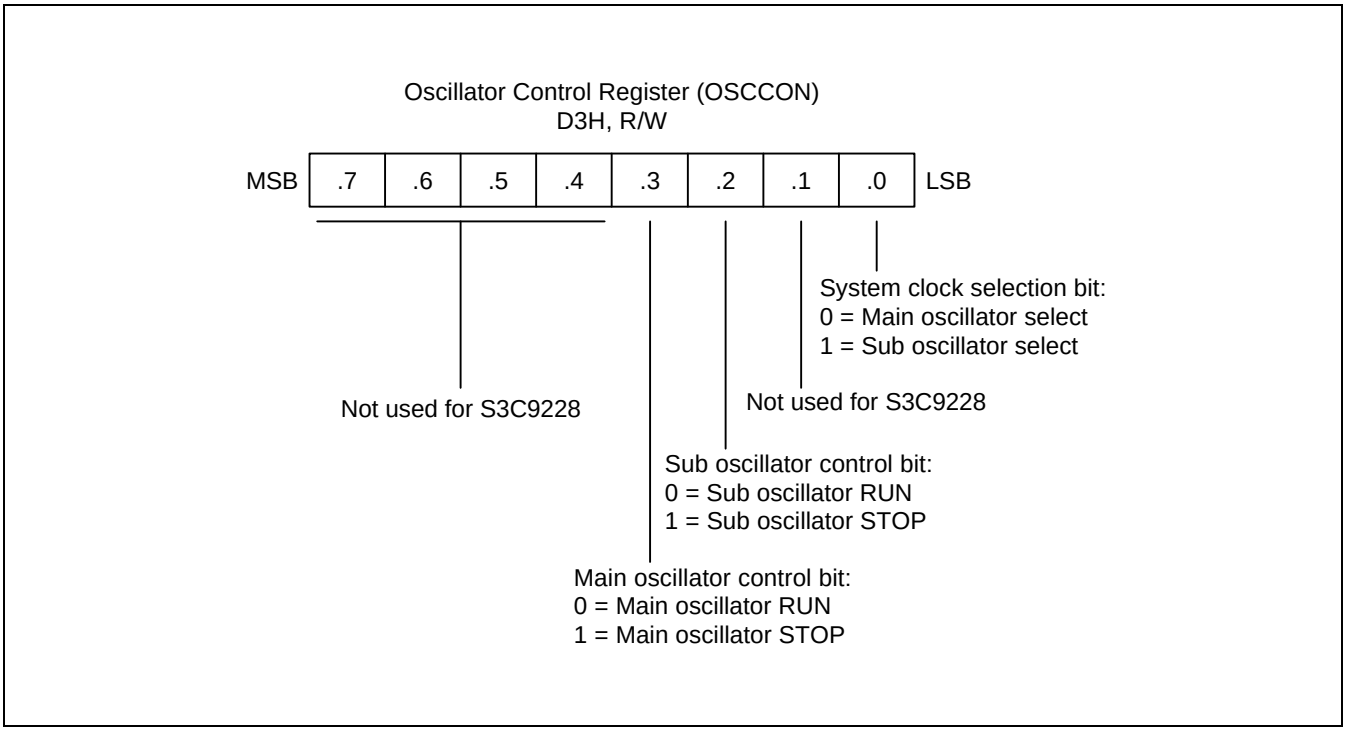


Figure 7-8. Oscillator Control Register (OSCCON)

SWITCHING THE CPU CLOCK

Data loadings in the oscillator control register, OSCCON, determine whether a main or a sub clock is selected as the CPU clock, and also how this frequency is to be divided by setting CLKCON. This makes it possible to switch dynamically between main and sub clocks and to modify operating frequencies.

OSCCON.0 select the main clock (fx) or the sub clock (fxt) for the system clock. OSCCON .3 start or stop main clock oscillation, and OSCCON.2 start or stop sub clock oscillation. CLKCON.4–.3 control the frequency divider circuit, and divide the selected fxx clock by 1, 2, 8, or 16.

For example, you are using the default system clock (normal operating mode and a main clock of fx/16) and you want to switch from the fx clock to a sub clock and to stop the main clock. To do this, you need to set OSCCON.0 to "1", take a delay, and OSCCON.3 to "1" sequently. This switches the clock from fx to fxt and stops main clock oscillation.

The following steps must be taken to switch from a sub clock to the main clock: first, set OSCCON.3 to "0" to enable main system clock oscillation. Then, after a certain number of machine cycles has elapsed, select the main clock by setting OSCCON.0 to "0".

☞ PROGRAMMING TIP — Switching the CPU clock

1. This example shows how to change from the main clock to the sub clock:

```
MA2SUB  OR      OSCCON,#01H     ;  Switches to the sub clock
        CALL    DLY16           ;  Delay 16ms
        OR      OSCCON,#08H     ;  Stop the main clock oscillation
        RET
```

2. This example shows how to change from sub clock to main clock:

```
SUB2MA  AND     OSCCON,#0F7H    ;  Start the main clock oscillation
        CALL    DLY16           ;  Delay 16 ms
        AND     OSCCON,#0FEH    ;  Switch to the main clock
        RET
DLY16   LD      R0,#20H
DEL     NOP
        DEC     R0
        JR      NZ,DEL
        RET
```

STOP CONTROL REGISTER (STPCON)

The STOP control register, STPCON, is located in address E0H. It is read/write addressable and has the following functions:

— Enable/Disable STOP instruction

After a reset, the STOP instruction is disabled, because the value of STPCON is "other values".
If necessary, you can use the STOP instruction by setting the value of STPCON to "10100101B".

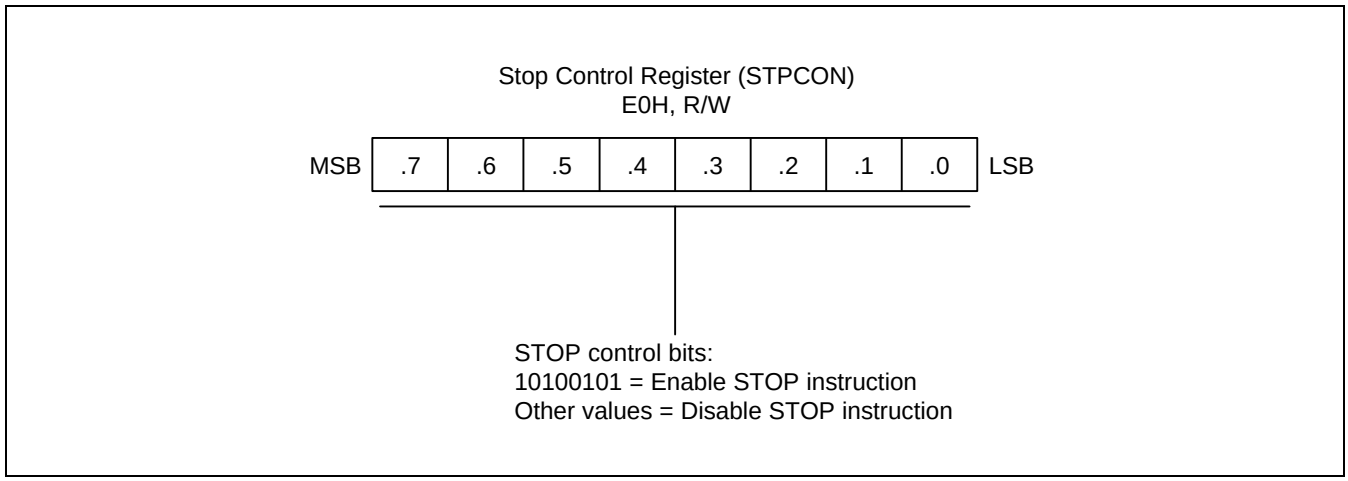


Figure 7-9. STOP Control Register (STPCON)

☞ PROGRAMMING TIP — How to Use Stop Instruction

This example shows how to go STOP mode when a main clock is selected as the system clock.

```
LD      STOPCON,#1010010B   ;  Enable STOP instruction
STOP                        ;  Enter STOP mode
NOP
NOP
NOP                         ;  Release STOP mode
LD      STOPCON,#00000000B  ;  Disable STOP instruction
```

NOTES

8 RESET and POWER-DOWN

SYSTEM RESET

OVERVIEW

During a power-on reset, the voltage at V_{DD} goes to High level and the RESET pin is forced to Low level. The RESET signal is input through a schmitt trigger circuit where it is then synchronized with the CPU clock. This procedure brings S3C9228/P9228 into a known operating status.

To allow time for internal CPU clock oscillation to stabilize, the RESET pin must be held to Low level for a minimum time interval after the power supply comes within tolerance. The minimum required oscillation stabilization time for a reset operation is 1 millisecond.

Whenever a reset occurs during normal operation (that is, when both V_{DD} and RESET are High level), the RESET pin is forced Low and the reset operation starts. All system and peripheral control registers are then reset to their default hardware values (see Table 8-1).

In summary, the following sequence of events occurs during a reset operation:

— All interrupts are disabled.

— The watchdog function (basic timer) is enabled.

— The P0.0–P0.3, P1, and P2.2–P2.3 are set to schmitt trigger input mode and all pull-up resistors are disabled for the I/O port pin circuits.

— Peripheral control and data registers are disabled and reset to their default hardware values.

— The program counter (PC) is loaded with the program reset address in the ROM, 0100H.

— When the programmed oscillation stabilization time interval has elapsed, the instruction stored in ROM location 0100H (and 0101H) is fetched and executed.

NOTE

To program the duration of the oscillation stabilization interval, you 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. In Stop mode, the operation of the CPU and main oscillator is halted. All peripherals which the main oscillator is selected as a clock source stop also because main oscillator stops. But the watch timer and LCD controller will not halted in stop mode if the sub clock is selected as watch timer clock source. The data stored in the internal register file are retained in stop mode. Stop mode can be released in one of three ways: by a system reset, by an internal watch timer interrupt (when sub clock is selected as clock source of watch timer), or by an external interrupt.

Example:

```
LD      STOPCON,#10100101B
STOP
NOP
NOP
NOP
LD      STOPCON,#00000000B
```

NOTES

1. Do not use stop mode if you are using an external clock source because X_{IN} input must be restricted internally to V_{SS} to reduce current leakage.
2. In application programs, a STOP instruction must be immediately followed by at least three NOP instructions. This ensures an adequate time interval for the clock to stabilize before the next instruction is executed. If three or more NOP instructions are not used after STOP instruction, leakage current could be flown because of the floating state in the internal bus.
3. To enable/disable STOP instruction, the STOPCON register should be written with 10100101B/other values before/after stop instruction.

Using RESET to Release Stop Mode

Stop mode is released when the RESET signal goes active (Low level): all system and peripheral control registers are reset to their default hardware values and the contents of all data registers are retained. When the programmed oscillation stabilization interval has elapsed, the CPU starts the system initialization routine by fetching the program instruction stored in ROM location 0100H.

Using an External Interrupt to Release Stop Mode

External interrupts can be used to release stop mode. For the S3C9228 microcontroller, we recommend using the INT interrupt, P0, P1, and P3.

Using an Internal Interrupt to Release Stop Mode

An internal interrupt, watch timer, can be used to release stop mode because the watch timer operates in stop mode if the clock source of watch timer is sub clock. If system clock is sub clock, you can't use any interrupts to release stop mode. That is, you had better use the idle instruction instead of stop one when sub clock is selected as the system clock.

Please note the following conditions for Stop mode release:

— If you release stop mode using an internal or external interrupt, the current values in system and peripheral control registers are unchanged.

— If you use an internal or 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.

— If you use an interrupt to release stop mode, the bit-pair setting for CLKCON.4/CLKCON.3 remains unchanged and the currently selected clock value is used.

— The internal or 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.

IDLE MODE

Idle mode is invoked by the instruction IDLE (opcode 6FH). In Idle mode, CPU operations are halted while some peripherals remain active. During Idle mode, the internal clock signal is gated away from the CPU and from all but the following peripherals, which remain active:

— Interrupt logic

— Basic timer

— Timer 1 (Timer A and B)

— Watch timer

— LCD controller

I/O port pins retain the mode (input or output) they had at the time Idle mode was entered.

Idle Mode Release

You can release Idle mode in one of two ways:

1. Execute a 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 the *slowest clock (1/16)* because of the hardware reset value for the CLKCON register. If all external interrupts are masked in the IMR register, a reset is the only way you can release Idle mode.
2. Activate any enabled interrupt — internal or external. When you use an interrupt to release Idle mode, the 2-bit CLKCON.4/CLKCON.3 value remains unchanged, and the *currently selected clock* value is used. The interrupt is then serviced. When the return-from-interrupt condition (IRET) occurs, the instruction immediately following the one which initiated Idle mode is executed.

HARDWARE RESET VALUES

Table 8-1 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 table 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 Values after RESET

Register Name	Mnemonic	Address		Bit Values after RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
Locations B8H–B9H are not mapped.											
Timer B Control Register	TBCON	202	BAH	–	0	0	0	0	0	0	–
Timer 1/A Control Register	TACON	203	BBH	0	0	0	0	0	0	0	–
Timer B Data Register	TBDATA	204	BCH	1	1	1	1	1	1	1	1
Timer A Data Register	TADATA	205	BDH	1	1	1	1	1	1	1	1
Timer B Counter	TBCNT	206	BEH	0	0	0	0	0	0	0	0
Timer A Counter	TACNT	207	BFH	0	0	0	0	0	0	0	0
A/D Converter Control Register	ADCON	208	D0H	–	–	0	0	0	0	0	0
A/D Converter Data Register (high byte)	ADDATAH	209	D1H	X	X	X	X	X	X	X	X
A/D Converter Data Register (low byte)	ADDATAL	210	D2H	–	–	–	–	–	–	X	X
Oscillator Control Register	OSCCON	211	D3H	–	–	–	–	0	0	–	0
System Clock Control Register	CLKCON	212	D4H	0	0	0	0	0	0	0	0
System Flags Register	FLAGS	213	D5H	X	X	X	X	–	–	–	–
Interrupt Pending Register 1	INTPND1	214	D6H	0	0	0	0	0	0	0	0
Interrupt Pending Register 2	INTPND2	215	D7H	–	–	0	0	0	0	0	0
LCD Port Control Register	LOPT	216	D8H	–	0	0	0	0	0	0	0
Stack Pointer	SP	217	D9H	X	X	X	X	X	X	X	X
Watch Timer Control Register	WTCON	218	DAH	0	0	0	0	0	0	0	–
Locations DBH is not mapped.											
Basic Timer Control Register	BTCON	220	DCH	0	0	0	0	0	0	0	0
Basic Timer Counter	BTCNT	221	DDH	0	0	0	0	0	0	0	0
Locations DEH is not mapped.											

Table 8-1. Register Values after RESET (Continued)

Register Name	Mnemonic	Address		Bit Values after RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
System Mode Register	SYM	223	DFH	–	–	–	–	0	0	0	0
STOP Control Register	STPCON	224	E0H	0	0	0	0	0	0	0	0
SIO Control Register	SIOCON	225	E1H	0	0	0	0	0	0	0	–
SIO Data Register	SIODATA	226	E2H	0	0	0	0	0	0	0	0
SIO Prescaler Register	SIOPS	227	E3H	0	0	0	0	0	0	0	0
Port 0 Data Register	P0	228	E4H	0	0	0	0	0	0	0	0
Port 1 Data Register	P1	229	E5H	0	0	0	0	0	0	0	0
Port 2 Data Register	P2	230	E6H	0	0	0	0	0	0	0	0
Port 3 Data Register	P3	231	E7H	0	0	0	0	0	0	0	0
Port 4 Data Register	P4	232	E8H	0	0	0	0	0	0	0	0
Port 5 Data Register	P5	233	E9H	0	0	0	0	0	0	0	0
Port 6 Data Register	P6	234	EAH	0	0	0	0	0	0	0	0
Port 0 Control Register	P0CON	235	EBH	0	0	0	0	0	0	0	0
Port 0 Pull-up Resistors Enable Register	P0PUR	236	ECH	–	–	–	–	0	0	0	0
Port 0 Interrupt Control Register	P0INT	237	EDH	–	–	–	–	0	0	0	0
Port 0 Interrupt Edge Selection Register	P0EDGE	238	EEH	–	–	–	–	0	0	0	0
Port 1 Control Register	P1CON	239	EFH	0	0	0	0	0	0	0	0
Port 1 Pull-up Resistors Enable Register	P1PUR	240	F0H	–	–	–	–	0	0	0	0
Port 1 Interrupt Control Register	P1INT	241	F1H	–	–	–	–	0	0	0	0
Port 1 Interrupt Edge Selection Register	P1EDGE	242	F2H	–	–	–	–	0	0	0	0
Port 2 Control Register	P2CON	243	F3H	0	0	0	0	0	0	0	0
Port 2 Pull-up Resistors Enable Register	P2PUR	244	F4H	–	–	–	–	0	0	0	0
Port 3 Control Register	P3CON	245	F5H	–	–	–	–	0	0	0	0
Port 3 Pull-up Resistors Enable Register	P3PUR	246	F6H	–	–	–	–	–	–	0	0
Port 3 Interrupt Control Register	P3INT	247	F7H	–	–	–	–	–	–	0	0
Port 3 Interrupt Edge Selection Register	P3EDGE	248	F8H	–	–	–	–	–	–	0	0
Port 4 Control Register (High Byte)	P4CONH	249	F9H	0	0	0	0	0	0	0	0
Port 4 Control Register (High Byte)	P4CONL	250	FAH	0	0	0	0	0	0	0	0
Port 5 Control Register (High Byte)	P5CONH	251	FBH	0	0	0	0	0	0	0	0
Port 5 Control Register (High Byte)	P5CONL	252	FCH	0	0	0	0	0	0	0	0
Port 6 Control Register	P6CON	253	FDH	0	0	0	0	0	0	0	0
LCD Mode Register	LMOD	254	FEH	–	0	0	0	0	0	0	0
Location FFH is not mapped.											

NOTES

9 I/O PORTS

OVERVIEW

The S3C9228/P9228 microcontroller has seven bit-programmable I/O ports, P0-P6. Port 0 is 6-bit port, port 1, port 2, and port 6 are 4-bit ports, port 3 is 2-bit port, and port 4 and port 5 are 8-bit ports. This gives a total of 36 I/O pins. Each port can be flexibly configured to meet application design requirements.

The CPU accesses ports by directly writing or reading port registers. No special I/O instructions are required. All ports of the S3C9228/P9228 except P0.4 and P0.5 can be configured to input or output mode. All LCD signal pins are shared with normal I/O ports.

Table 9-1 gives you a general overview of S3C9228 I/O port functions.

Table 9-1. S3C9228 Port Configuration Overview

Port	Configuration Options
0	1-bit programmable I/O port except P0.4 and P0.5. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups. The P0.4 and P0.5 are only push-pull output ports. Alternatively P0.0-P0.3 can be used as input for external interrupts INT and can be used as TAOUT, T1CLK, and BUZ.
1	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups. Alternatively P1 can be used as input for external interrupts INT and can be used as AD0-AD3.
2	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups. Alternatively P2.0 and P2.1 can be used as outputs for LCD segment signals and P2.0-P2.2 can be used as SCK, SO, and SI.
3	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups. Alternatively P3 can be used as input for external interrupts INTP and can be used as outputs for LCD segment signals.
4	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups. Alternatively P4 can be used as outputs for LCD segment signals.
5	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups. Alternatively P5.0-P5.3 can be used as outputs for LCD segment signals and P5.4-P5.7 can be used as outputs for LCD common or segment signals.
6	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups. Alternatively P6 can be used as outputs for LCD common signals.

PORT DATA REGISTERS

Table 9-2 gives you an overview of the register locations of all seven S3C9228 I/O port data registers. Data registers for ports 1, 2, 3, 4, 5, and 6 have the general format shown in Figure 9-1.

Table 9-2. Port Data Register Summary

Register Name	Mnemonic	Decimal	Hex	R/W
Port 0 data register	P0	228	E4H	R/W
Port 1 data register	P1	229	E5H	R/W
Port 2 data register	P2	230	E6H	R/W
Port 3 data register	P3	231	E7H	R/W
Port 4 data register	P4	232	E8H	R/W
Port 5 data register	P5	233	E9H	R/W
Port 6 data register	P6	234	EAH	R/W

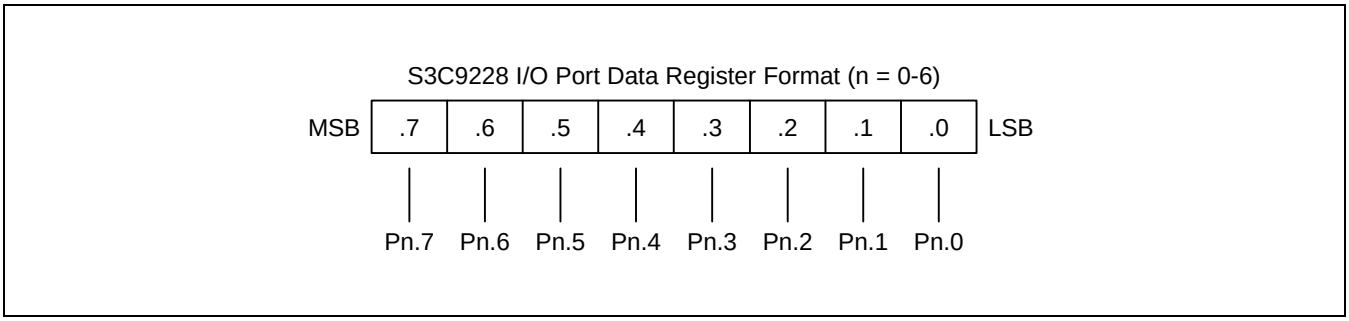


Figure 9-1. S3C9228 I/O Port Data Register Format

PORT 0

Port 0 is an 6-bit I/O port with individually configurable pins. Port 0 pins are accessed directly by writing or reading the port 0 data register, P0 at location E4H in page 0. P0.0-P0.3 can serve as inputs (with or without pull-up), as outputs (push-pull or open-drain) or you can be configured the following functions.

— Low-nibble pins (P0.0-P0.3): TAOUT,T1CLK, BUZ, INT

— High-nibble pins (P0.4-P0.5): push-pull output ports (only 44-QFP package)

Port 0 Control register (P0CON)

Port 0 has a 8-bit control register: P0CON for P0.0-P0.3. A reset clears the P0CON register to “00H”, configuring pins to input mode. You use control register setting to select input or output mode (push-pull or open-drain) and enable the alternative functions.

When programming this port, please remember that any alternative peripheral I/O function you configure using the port 0 control register must also be enabled in the associated peripheral module.

Port 0 Pull-up Resistor Control Register (P0PUR)

Using the port 0 pull-up resistor control register, P0PUR (ECH, page 0), you can configure pull-up resistors to individual port 0 pins.

Port 0 Interrupt Enable, Pending, and Edge Selection Registers (P0INT, INTPND1.3-.0, P0EDGE)

To process external interrupts at the port 0 pins, three additional control registers are provided: the port 0 interrupt enable register P0INT (EDH, page 0), the port 0 interrupt pending bits INTPND1.3-.0 (D6H, page 0), and the port 0 interrupt edge selection register P0EDGE (EEH, page 0).

The port 0 interrupt pending register bits lets you check for interrupt pending conditions and clear the pending condition when the interrupt service routine has been initiated. The application program detects interrupt requests by polling the INTPND1.3-.0 register at regular intervals.

When the interrupt enable bit of any port 0 pin is "1", a rising or falling edge at that pin will generate an interrupt request. The corresponding INTPND1 bit is then automatically set to "1" and the IRQ level goes low to signal the CPU that an interrupt request is waiting. When the CPU acknowledges the interrupt request, application software must the clear the pending condition by writing a "0" to the corresponding INTPND1 bit.

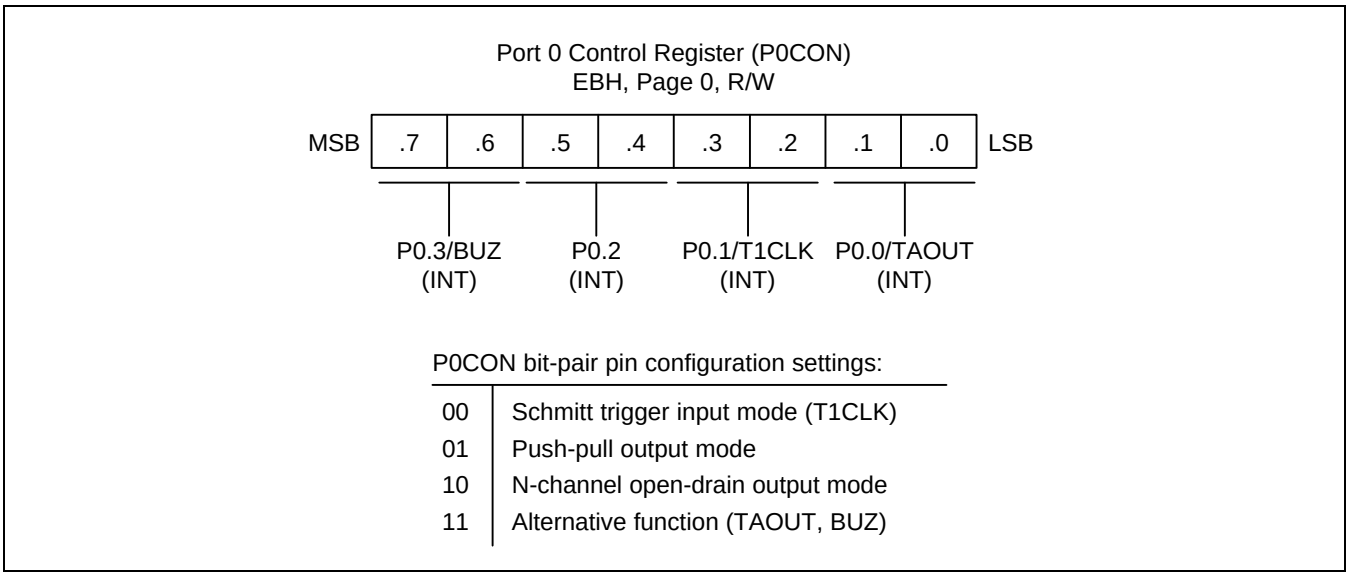


Figure 9-2. Port 0 Control Register (P0CON)

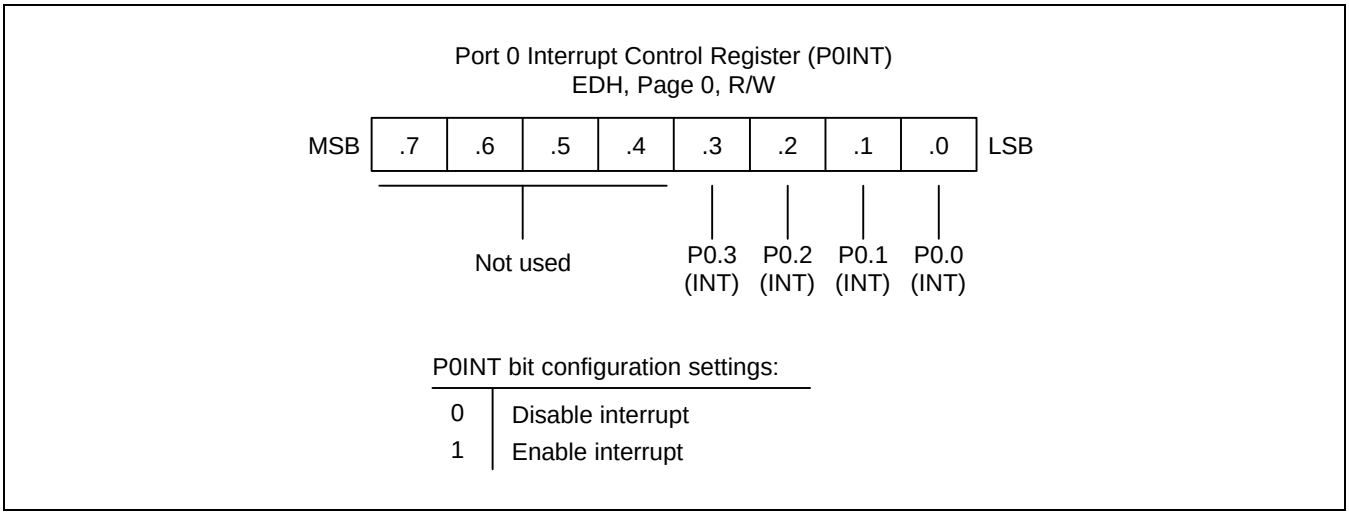


Figure 9-3. Port 0 Interrupt Control Register (P0INT)

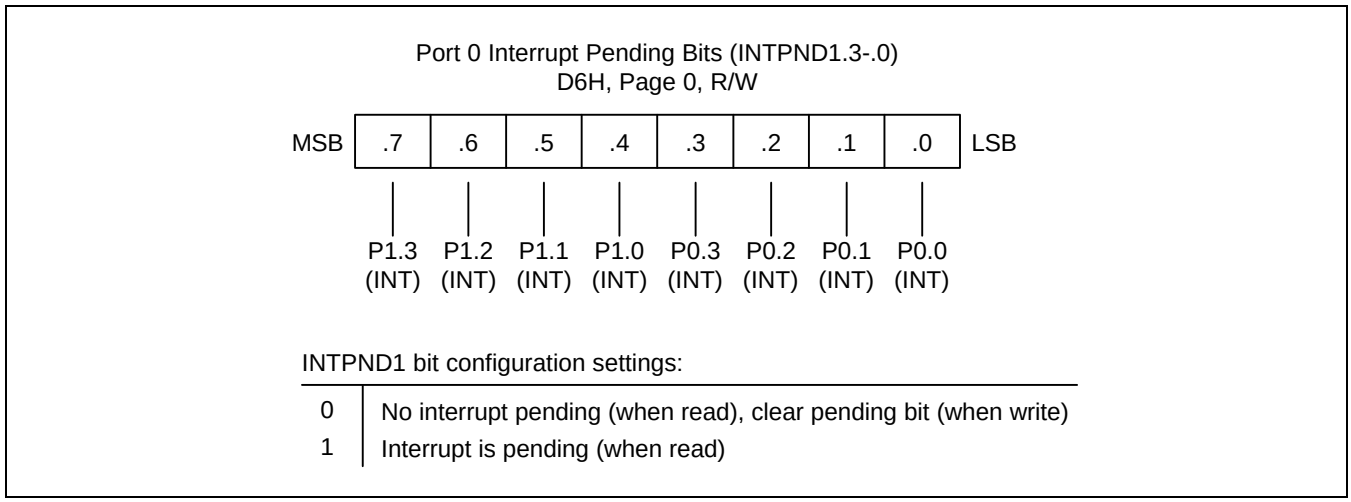


Figure 9-4. Port 0 Interrupt Pending Bits (INTPND1.3-.0)

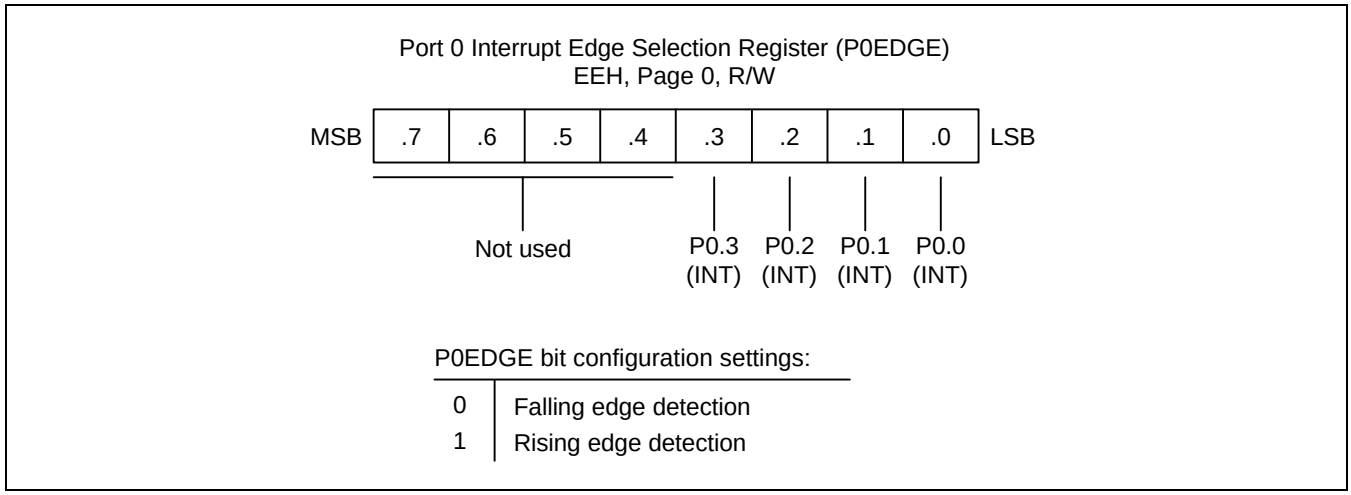


Figure 9-5. Port 0 Interrupt Edge Selection Register (P0EDGE)

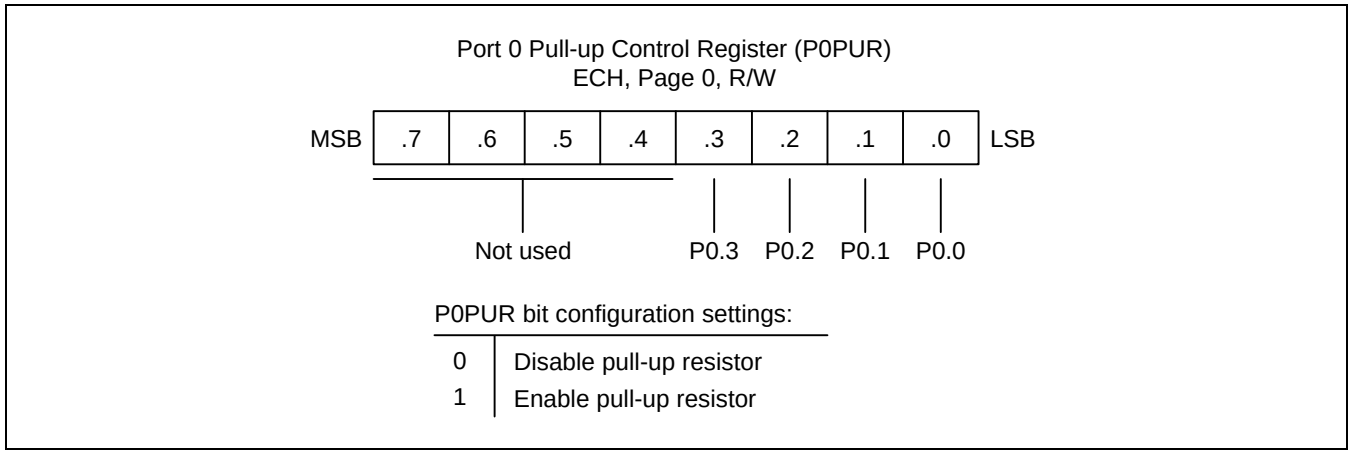


Figure 9-6. Port 0 Pull-up Control Register (P0PUR)

PORT 1

Port 1 is an 4-bit I/O port with individually configurable pins. Port 1 pins are accessed directly by writing or reading the port 1 data register, P1 at location E5H in page 0. P1.0-P1.3 can serve as inputs (with or without pull-up), as outputs (push-pull or open-drain) or you can be configured the following functions.

— Low-nibble pins (P1.0-P1.3): AD0-AD3, INT

Port 1 Control Register (P1CON)

Port 1 has a 8-bit control register: P1CON for P1.0-P1.3. A reset clears the P1CON register to "00H", configuring pins to input mode. You use control register setting to select input or output mode (push-pull or open-drain) and enable the alternative functions.

When programming this port, please remember that any alternative peripheral I/O function you configure using the port 1 control register must also be enabled in the associated peripheral module.

Port 1 Pull-up Resistor Control Register (P1PUR)

Using the port 1 pull-up resistor control register, P1PUR (F0H, page 0), you can configure pull-up resistors to individual port 1 pins.

Port 1 Interrupt Enable, Pending, and Edge Selection Registers (P1INT, INTPND1.7-.4, P1EDGE)

To process external interrupts at the port 1 pins, three additional control registers are provided: the port 1 interrupt enable register P1INT (F1H, page 0), the port 1 interrupt pending bits INTPND1.7-.4 (D6H, page 0), and the port 1 interrupt edge selection register P1EDGE (F2H, page 0).

The port 1 interrupt pending register bits lets you check for interrupt pending conditions and clear the pending condition when the interrupt service routine has been initiated. The application program detects interrupt requests by polling the INTPND1.7-.4 register at regular intervals.

When the interrupt enable bit of any port 1 pin is "1", a rising or falling edge at that pin will generate an interrupt request. The corresponding INTPND1 bit is then automatically set to "1" and the IRQ level goes low to signal the CPU that an interrupt request is waiting. When the CPU acknowledges the interrupt request, application software must the clear the pending condition by writing a "0" to the corresponding INTPND1 bit.

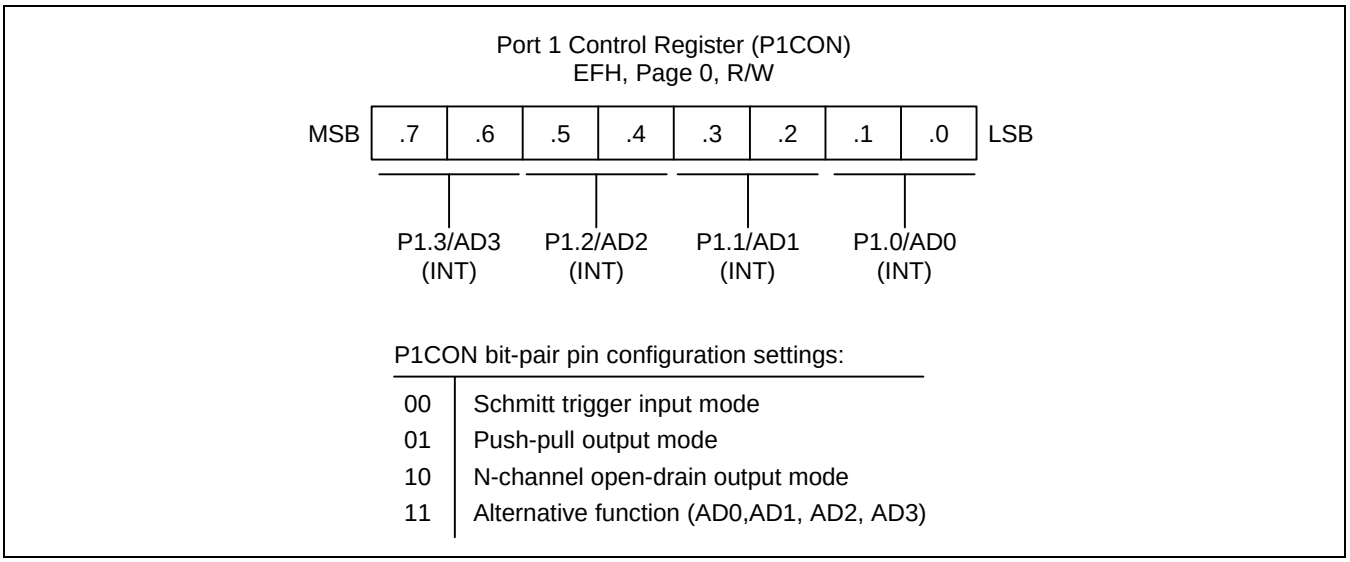


Figure 9-7. Port 1 Control Register (P1CON)

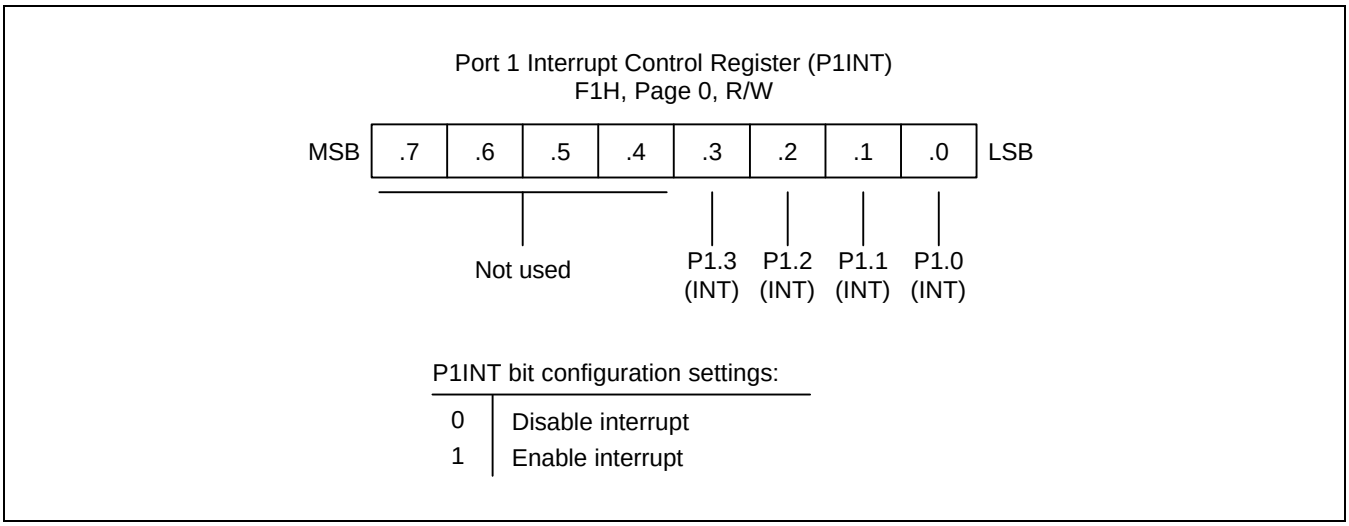


Figure 9-8. Port 1 Interrupt Control Register (P1INT)

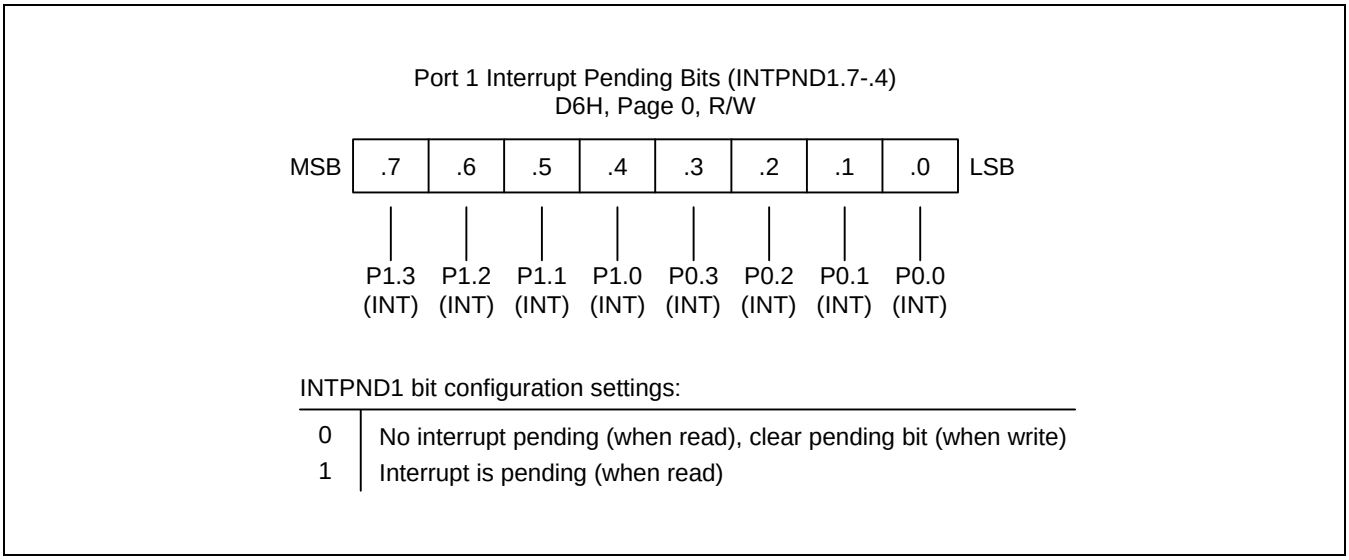


Figure 9-9. Port 1 Interrupt Pending Bits (INTPND1.7-.4)

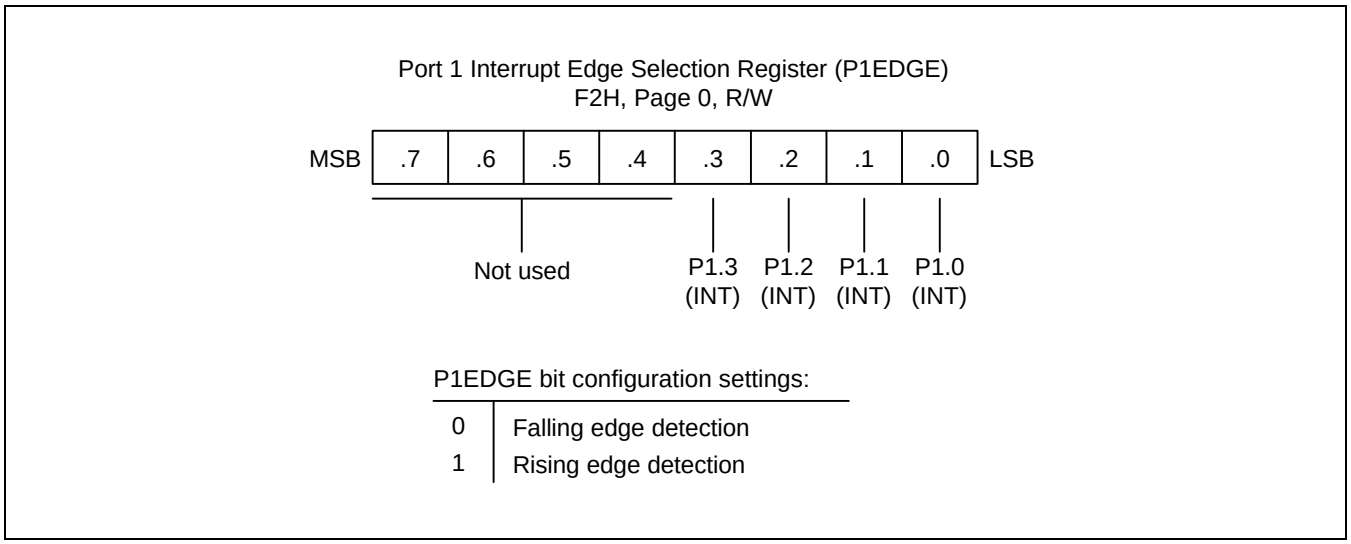


Figure 9-10. Port 1 Interrupt Edge Selection Register (P1EDGE)

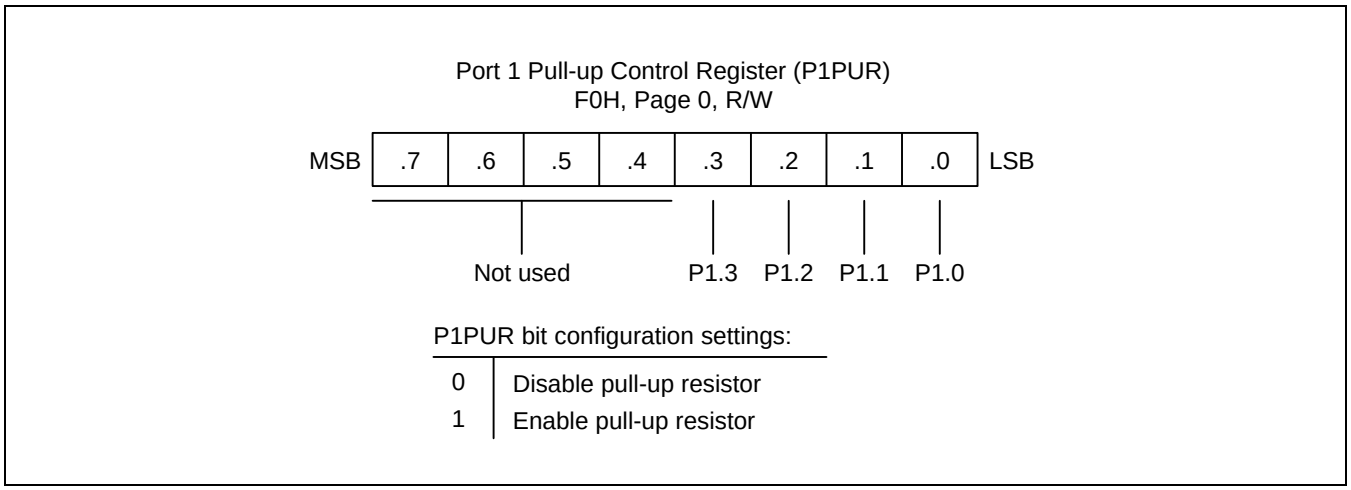


Figure 9-11. Port 1 Pull-up Control Register (P1PUR)

PORT 2

Port 2 is an 4-bit I/O port with individually configurable pins. Port 2 pins are accessed directly by writing or reading the port 2 data register, P2 at location E6H in page 0. P2.0-P2.3 can serve as inputs (with or without pull-up), as outputs (push-pull or open-drain) or you can be configured the following functions.

— Low-nibble pins (P2.0-P2.3): SCK, SO, SI, SEG0-SEG1

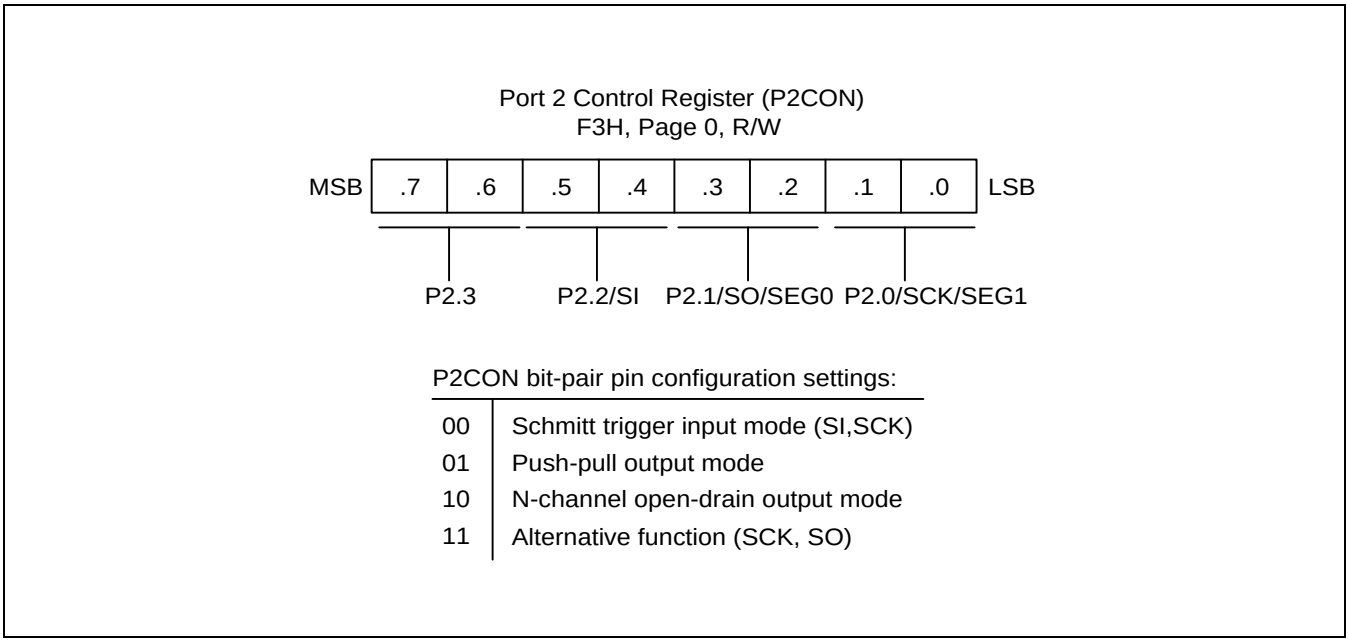
Port 2 Control Register (P2CON)

Port 2 has a 8-bit control register: P2CON for P2.0-P2.3. A reset clears the P2CON register to "00H", configuring pins to input mode. You use control register setting to select input or output mode (push-pull or open-drain) and enable the alternative functions.

When programming this port, please remember that any alternative peripheral I/O function you configure using the port 2 control register must also be enabled in the associated peripheral module.

Port 2 Pull-up Resistor Control Register (P2PUR)

Using the port 2 pull-up resistor control register, P2PUR (F4H, page 0), you can configure pull-up resistors to individual port 2 pins.



P2CON bit-pair pin configuration settings:

00	Schmitt trigger input mode (SI,SCK)
01	Push-pull output mode
10	N-channel open-drain output mode
11	Alternative function (SCK, SO)

Figure 9-12. Port 2 Control Register (P2CON)

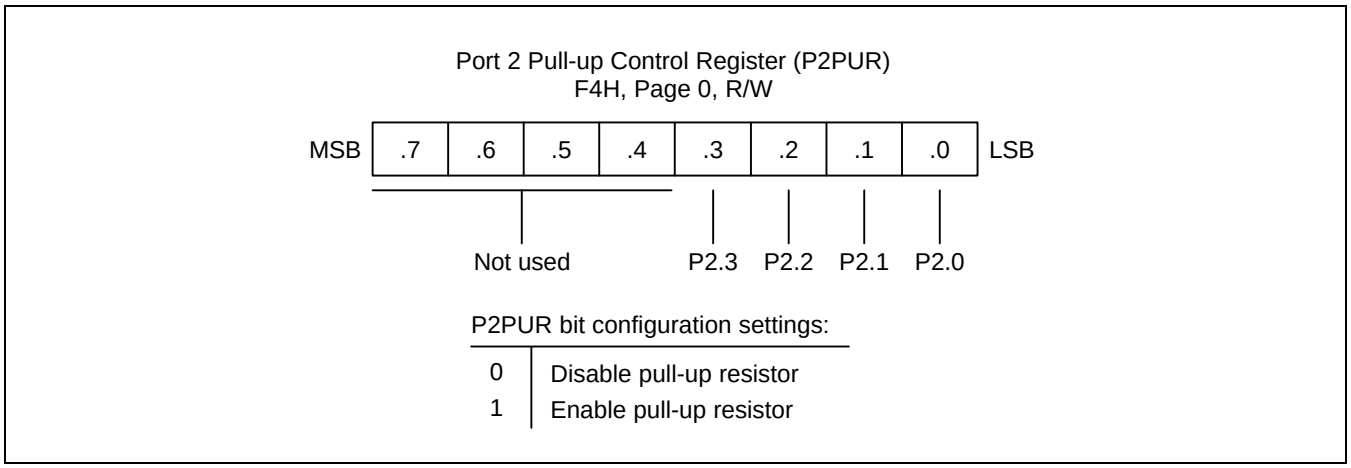


Figure 9-13. Port 2 Pull-up Control Register (P2PUR)

PORT 3

Port 3 is an 2-bit I/O port with individually configurable pins. Port 3 pins are accessed directly by writing or reading the port 3 data register, P3 at location E7H in page 0. P3.0-P3.1 can serve as inputs (with or without pull-up, and high impedance input), as outputs (push-pull or open-drain) or you can be configured the following functions.

— Low-nibble pins (P3.0-P3.1): SEG2-SEG3, INTP

Port 3 Control Register (P3CON)

Port 3 has a 8-bit control register: P3CON for P3.0-P3.1. A reset clears the P3CON register to "00H", configuring pins to input mode. You use control register setting to select input or output mode (push-pull or open-drain).

Port 3 Pull-up Resistor Control Register (P3PUR)

Using the port 3 pull-up resistor control register, P3PUR (F6H, page 0), you can configure pull-up resistors to individually port 3 pins.

Port 3 Interrupt Enable, Pending, and Edge Selection Registers(P3INT, INTPND2.5-.4, P3EDGE)

To process external interrupts at the port 3 pins, three additional control registers are provided: the port 3 interrupt enable register P3INT (F7H, page 0), the port 3 interrupt pending bits INTPND2.5-.4 (D7H, page 0), and the port 3 interrupt edge selection register P3EDGE (F8H, page 0).

The port 3 interrupt pending register bits lets you check for interrupt pending conditions and clear the pending condition when the interrupt service routine has been initiated. The application program detects interrupt requests by polling the INTPND2.5-.4 register at regular intervals.

When the interrupt enable bit of any port 3 pin is "1", a rising or falling edge at that pin will generate an interrupt request. The corresponding INTPND2 bit is then automatically set to "1" and the IRQ level goes low to signal the CPU that an interrupt request is waiting. When the CPU acknowledges the interrupt request, application software must the clear the pending condition by writing a "0" to the corresponding INTPND2 bit.

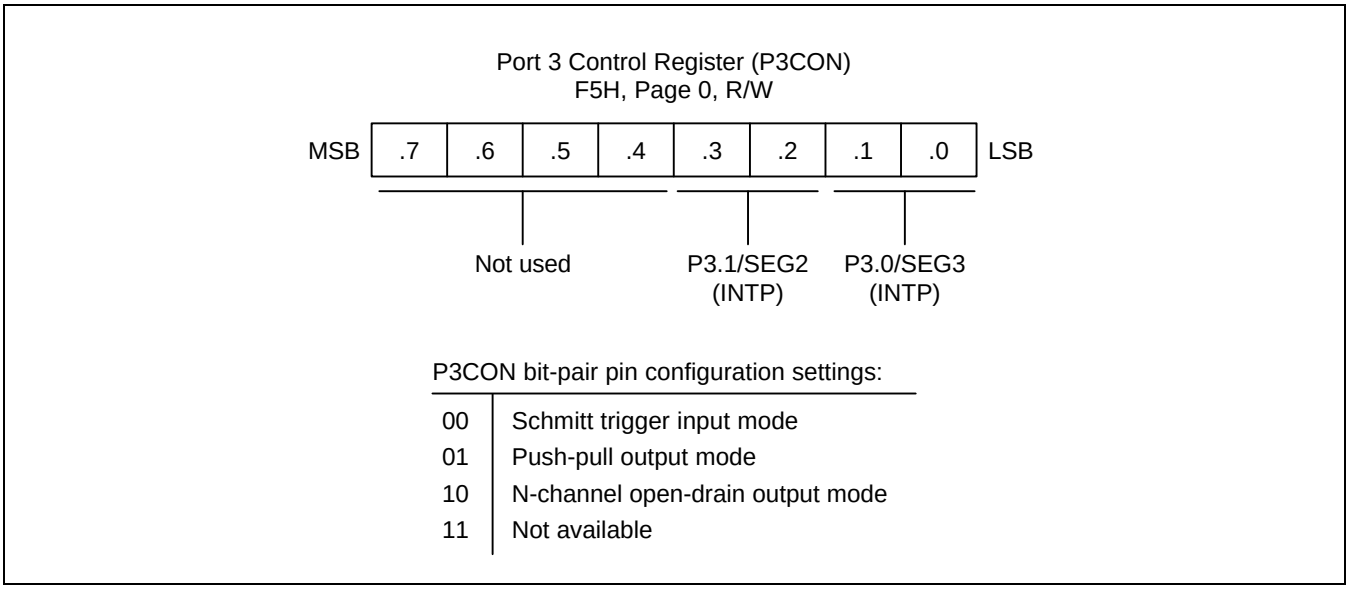


Figure 9-14. Port 3 Control Register (P3CON)

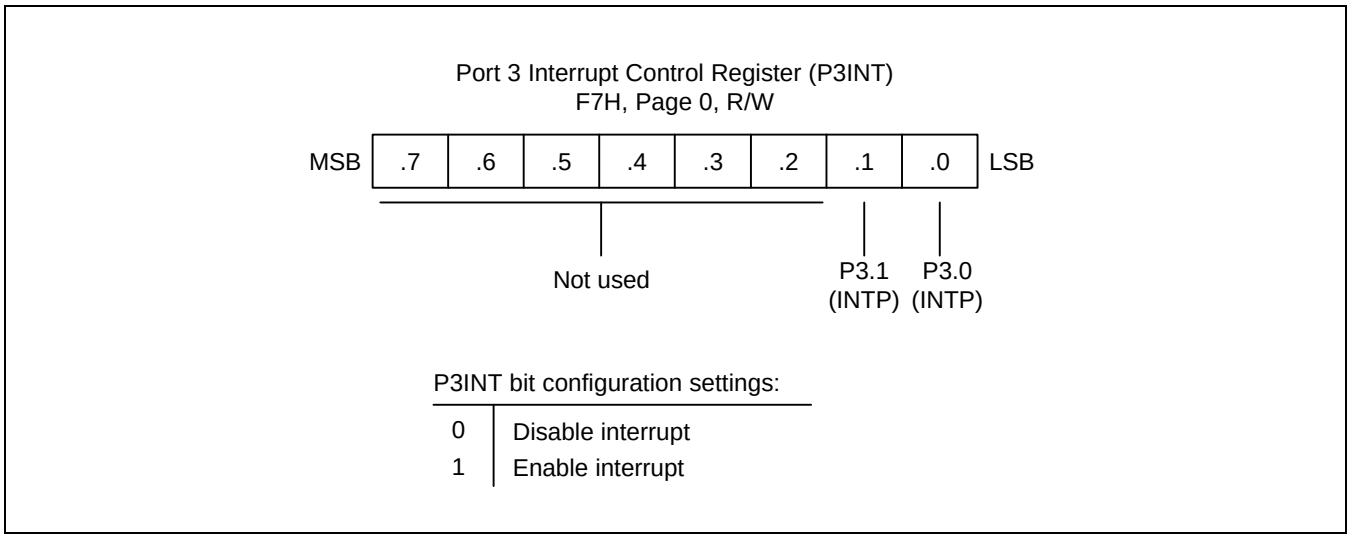


Figure 9-15. Port 3 Interrupt Control Register (P3INT)

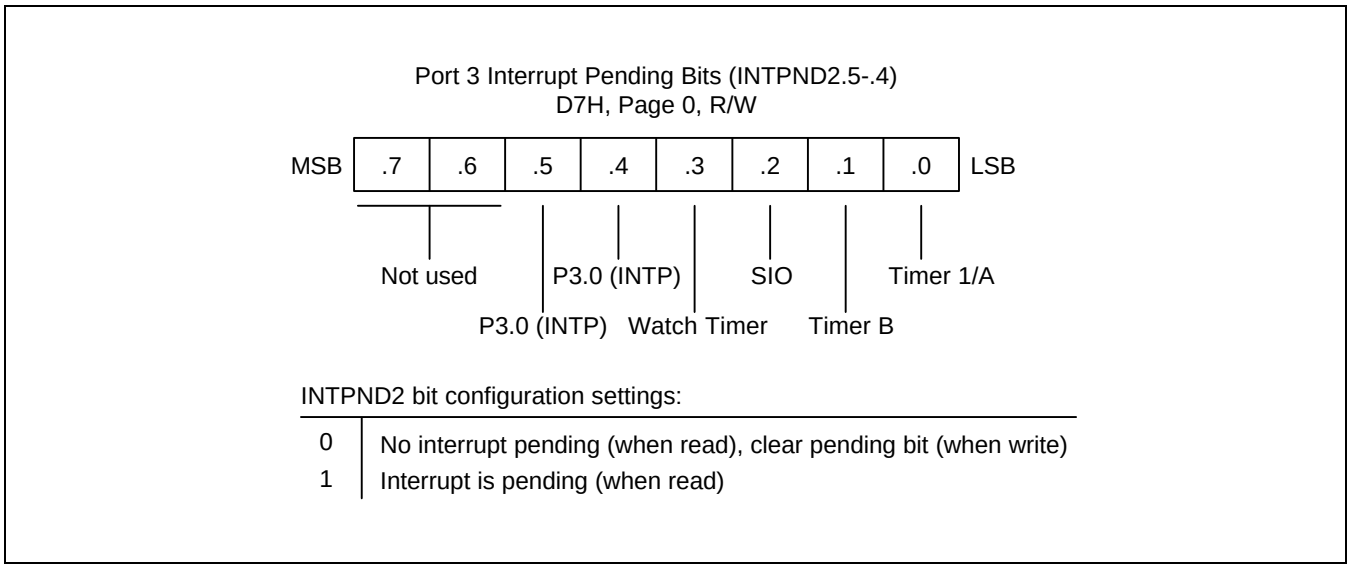


Figure 9-16. Port 3 Interrupt Pending Bits (INTPND2.5-.4)

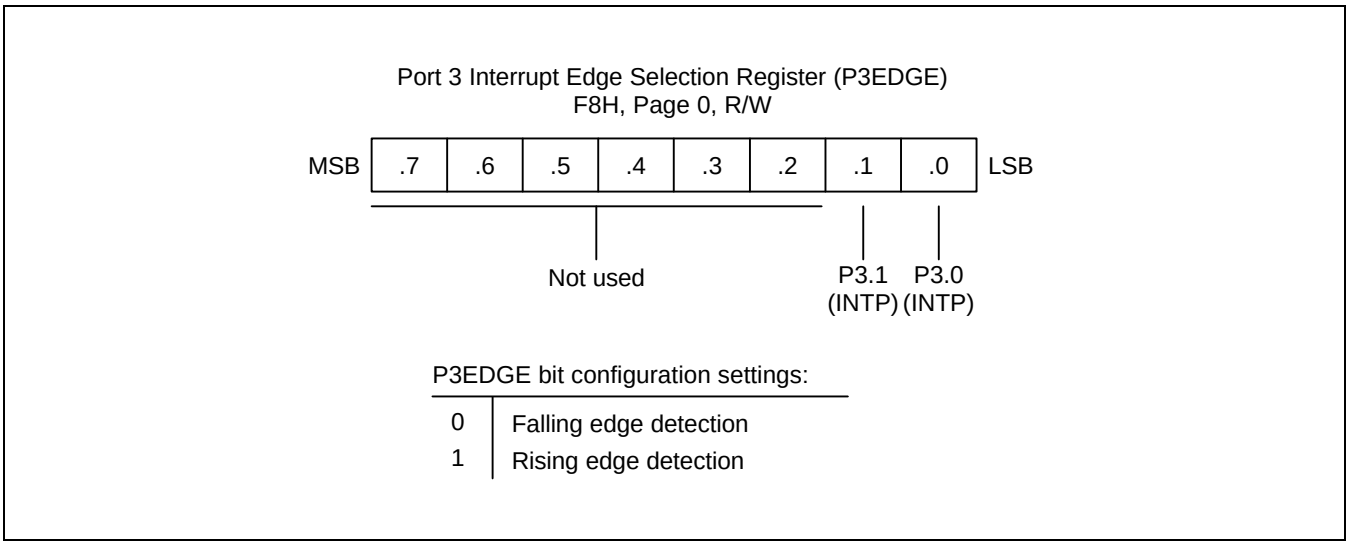


Figure 9-17. Port 3 Interrupt Edge Selection Register (P3EDGE)

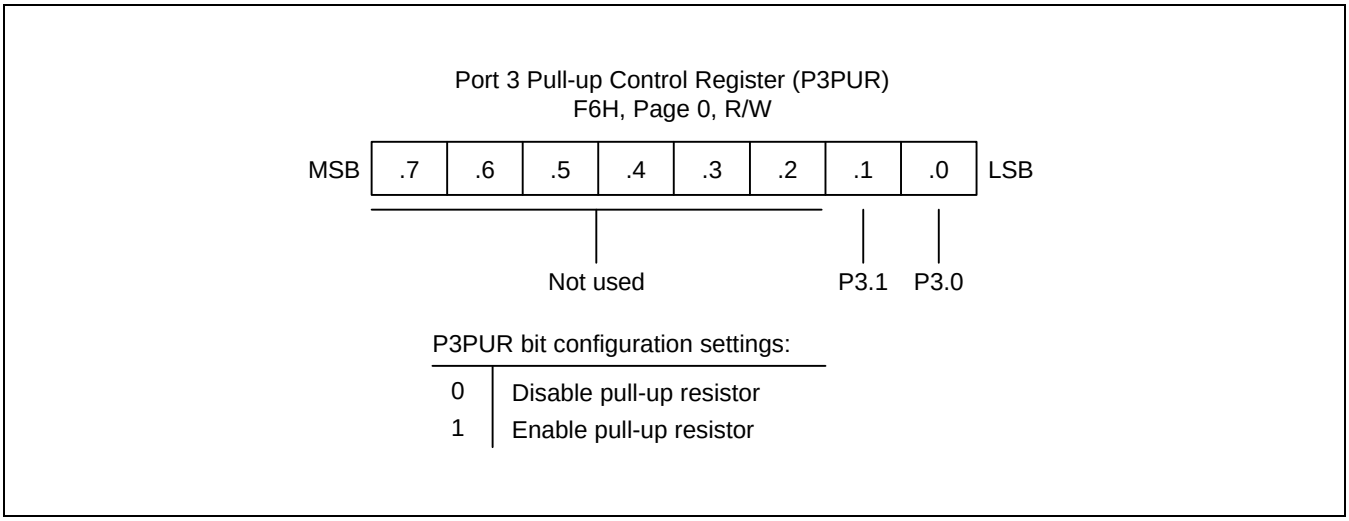


Figure 9-18. Port 3 Pull-up Control Register (P3PUR)

PORT 4

Port 4 is an 8-bit I/O port with individually configurable pins. Port 4 pins are accessed directly by writing or reading the port 4 data register, P4 at location E8H in page 0. P4.0-P4.7 can serve as inputs or as push-pull, open-drain outputs. You can configure the following alternative functions with LCD port control register, LPOT:

— Low-nibble pins (P4.0-P4.3): SEG4-SEG7

— High-nibble pins (P4.4-P4.7): SEG8-SEG11

Port 4 Control Registers (P4CONH, P4CONL)

Port 4 has two 8-bit control registers: P4CONH for P4.4-P4.7 and P4CONL for P4.0-P4.3. A reset clears the P4CONH and P4CONL registers to "00H", configuring all pins to input mode. You use control registers setting to select input or output mode.

Port 4 Control Register, High Byte (P4CONH)
F9H, Page 0, R/W

MSB	.7	.6	.5	.4	.3	.2	.1	.0	LSB

P4.7/SEG11 P4.6/SEG10 P4.5/SEG9 P4.4/SEG8

P4CONH bit-pair pin configuration settings:

00	Input mode
01	Push-pull output mode
10	N-channel open-drain output mode
11	Input mode with pull-up

Figure 9-19. Port 4 High-Byte Control Register (P4CONH)

Port 4 Control Register, Low Byte (P4CONL)
FAH, Page 0, R/W

MSB	.7	.6	.5	.4	.3	.2	.1	.0	LSB

P4.3/SEG7 P4.2/SEG6 P4.1/SEG5 P4.0/SEG4

P4CONL bit-pair pin configuration settings:

00	Input mode
01	Push-pull output mode
10	N-channel open-drain output mode
11	Input mode with pull-up

Figure 9-20. Port 4 Low-Byte Control Register (P4CONL)

PORT 5

Port 5 is an 8-bit I/O port with individually configurable pins. Port 5 pins are accessed directly by writing or reading the port 5 data register, P5 at location E9H in page 0. P5.0-P5.7 can serve as inputs or as push-pull, open-drain outputs. You can configure the following alternative functions with LCD port control register, LPOT:

— Low-nibble pins (P5.0-P5.3): SEG12-SEG15

— High-nibble pins (P5.4-P5.7): SEG16-SEG19, COM4-COM7

Port 5 Control Registers (P5CONH, P5CONL)

Port 5 has two 8-bit control registers: P5CONH for P5.4-P5.7 and P4CONL for P5.0-P5.3. A reset clears the P5CONH and P5CONL registers to "00H", configuring all pins to input mode. You use control registers setting to select input or output mode.

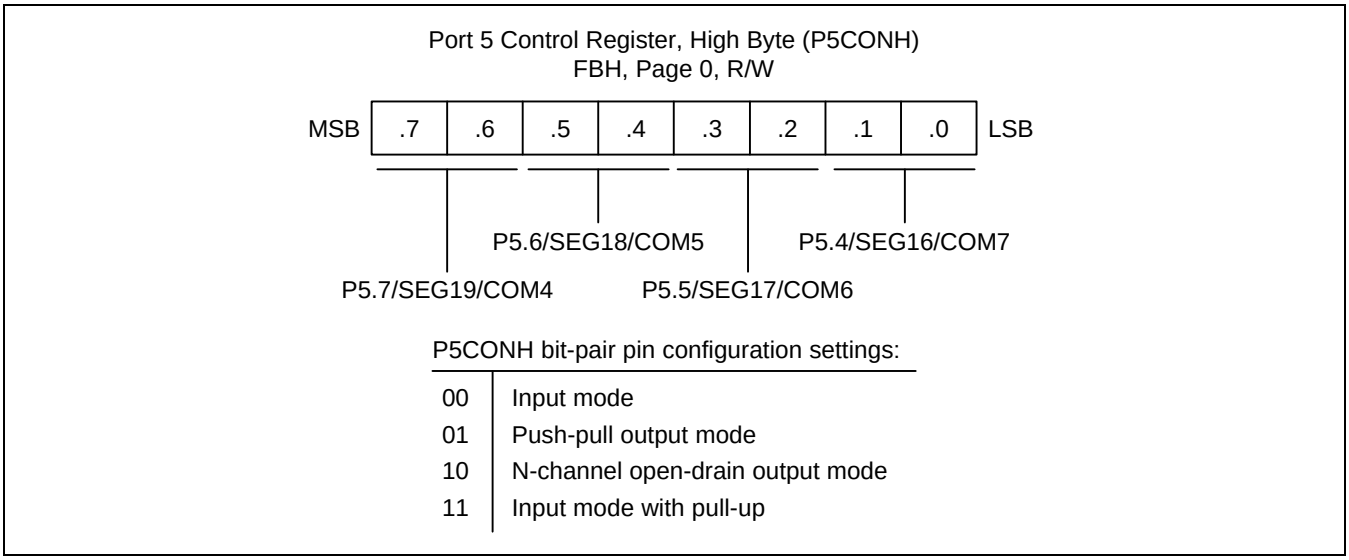


Figure 9-21. Port 5 High-Byte Control Register (P5CONH)

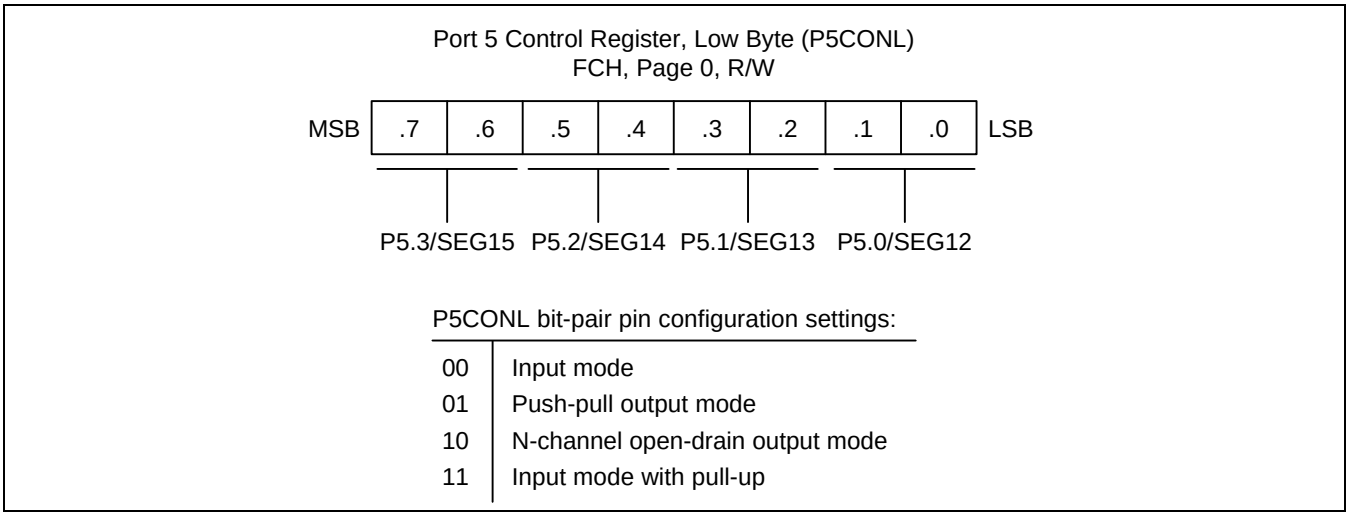


Figure 9-22. Port 5 Low-Byte Control Register (P5CONL)

PORT 6

Port 6 is an 4-bit I/O port with individually configurable pins. Port 6 pins are accessed directly by writing or reading the port 6 data register, P6 at location EAH in page 0. P6.0-P6.3 can serve as inputs or as push-pull, open-drain outputs. You can configure the following alternative functions with LCD port control register, LPOT:

— Low-nibble pins (P6.0-P6.3): COM0-COM3

Port 6 Control Register (P6CON)

Port 6 has a 8-bit control register: P6CONH for P6.0-P6.3. A reset clears the P6CON registers to "00H", configuring all pins to input mode. You use control registers setting to select input or output mode.

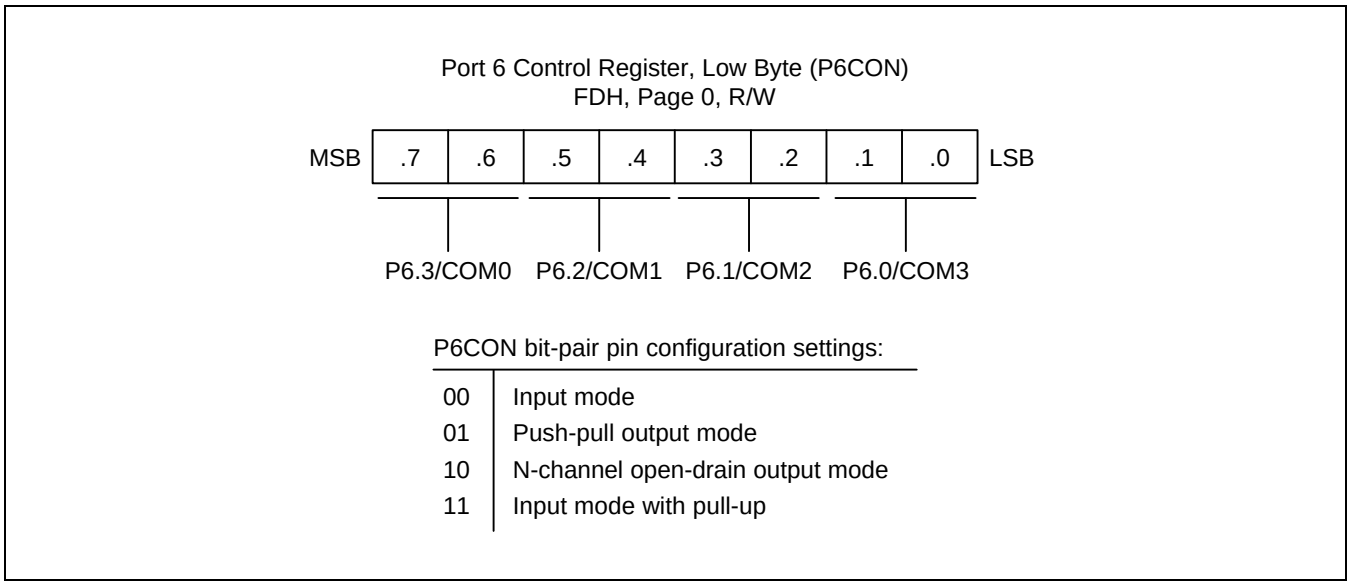


Figure 9-23. Port 6 Control Register (P6CON)

10 BASIC TIMER

OVERVIEW

Basic timer (BT) can be used 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 (fxx divided by 4096, 1024, 128, or 16) with multiplexer
— 8-bit basic timer counter, BTCNT (DDH, read-only)
— Basic timer control register, BTCON (DCH, read/write)

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 page 0, address DCH, 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 fxx/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 (page 0, DDH), can be cleared at any time during normal operation by writing a "1" to BTCON.1. To clear the frequency dividers for the basic timer input clock and timer counters, you write a "1" to BTCON.0.

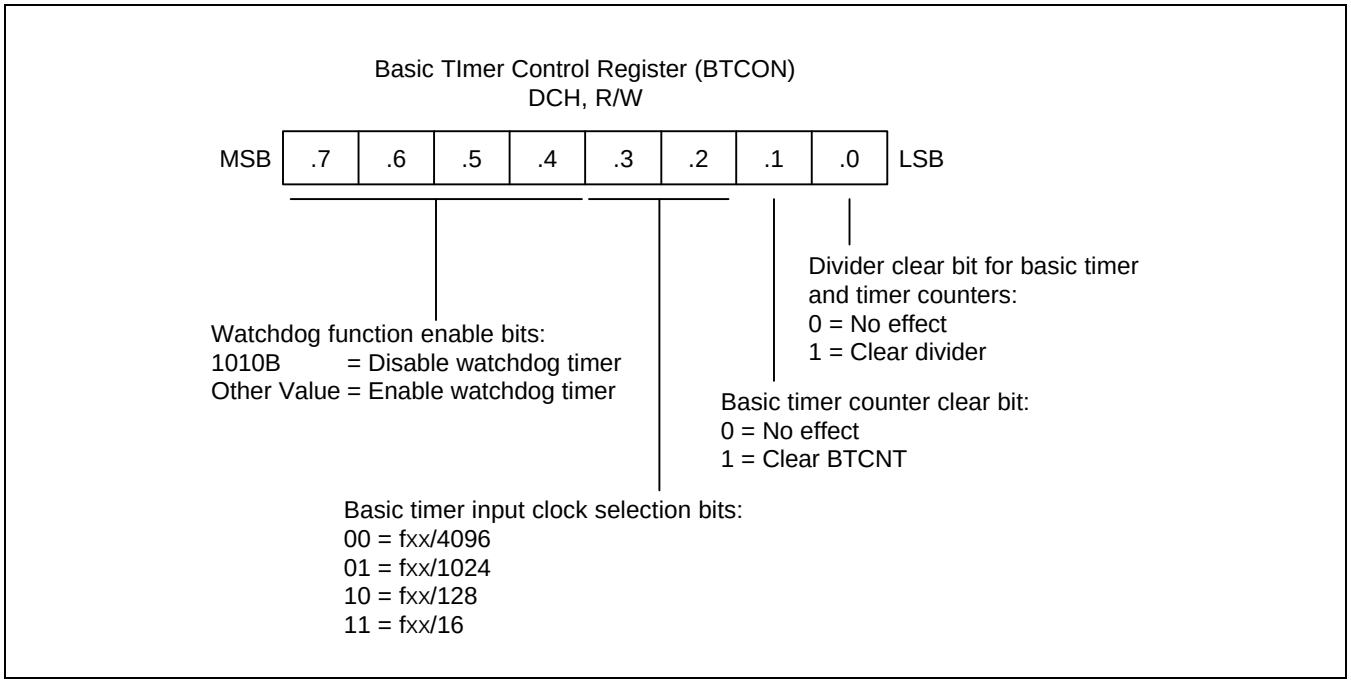


Figure 10-1. Basic Timer Control Register (BTCON)

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 BT clock.

A reset whenever a basic timer counter overflow occurs. During 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 BT counter clear operation will not be executed and a basic timer overflow will occur, initiating a reset. In other words, during 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.

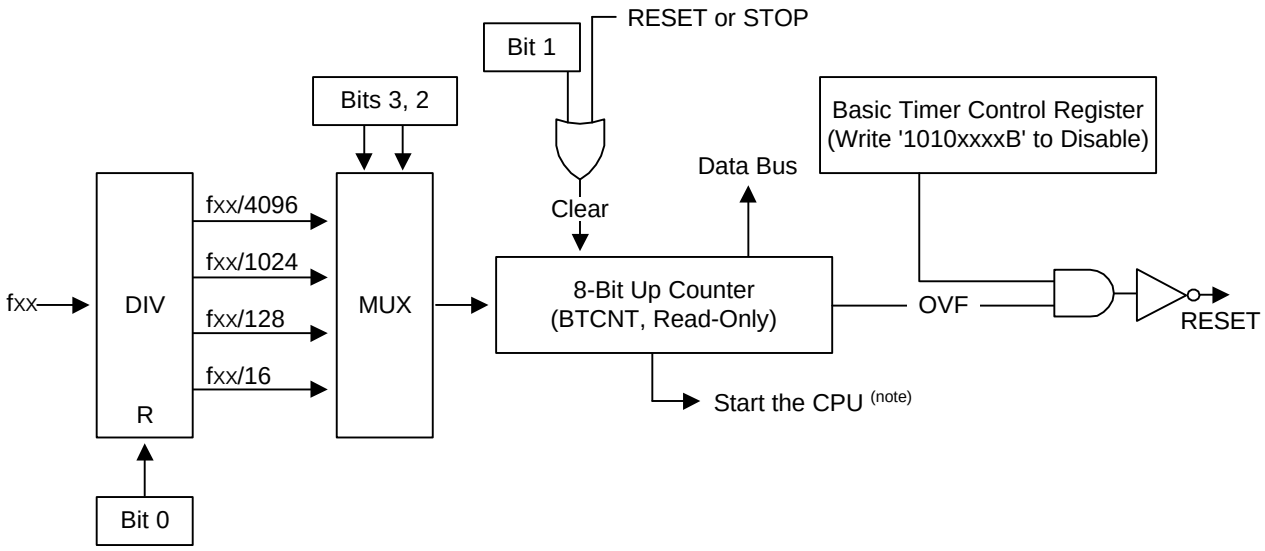
Oscillation Stabilization Interval Timer Function

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

In stop mode, whenever a reset or an internal and an external interrupt occurs, the oscillator starts. The BTCNT value then starts increasing at the rate of fxx/4096 (for reset), or at the rate of the preset clock source (for an internal and an external interrupt). When BTCNT.3 overflows, 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 internal and 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 fxx/4096. If an internal and 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 3 of the basic timer counter overflows.
4. When a BTCNT.3 overflow occurs, normal CPU operation resumes.



NOTE: During a power-on reset operation, the CPU is idle during the required oscillation stabilization interval (until bit 4 of the basic timer counter overflows).

Figure 10-2. Basic Timer Block Diagram

11 TIMER 1

ONE 16-BIT TIMER MODE (TIMER 1)

The 16-bit timer 1 is used in one 16-bit timer or two 8-bit timers mode. If TACON.7 is set to "1", timer 1 is used as a 16-bit timer. If TACON.7 is set to "0", timer 1 is used as two 8-bit timers.

— One 16-bit timer mode (Timer 1)

— Two 8-bit timers mode (Timer A and B)

OVERVIEW

The 16-bit timer 1 is an 16-bit general-purpose timer. Timer 1 has the interval timer mode by using the appropriate TACON setting.

Timer 1 has the following functional components:

— Clock frequency divider (fxx divided by 512, 256, 64, 8, or 1, fxt, and T1CLK: External clock) with multiplexer

— 16-bit counter (TACNT, TBCNT), 16-bit comparator, and 16-bit reference data register (TADATA, TBDATA)

— Timer 1 match interrupt generation

— Timer 1 control register, TACON (page 0, BBH, read/write)

FUNCTION DESCRIPTION

Interval Timer Function

The timer 1 module can generate an interrupt: the timer 1 match interrupt (T1INT).

The T1INT pending condition should be cleared by software when it has been serviced. Even though T1INT is disabled, the application's service routine can detect a pending condition of T1INT by the software and execute it's sub-routine. When this case is used, the T1INT pending bit must be cleared by the application sub-routine by writing a "0" to the INTPND2.0 pending bit.

In interval timer mode, a match signal is generated when the counter value is identical to the values written to the timer 1 reference data registers, TADATA and TBDATA. The match signal generates a timer 1 match interrupt and clears the counter.

If, for example, you write the value 32H and 10H to TADATA and TBDATA, respectively, and 8EH to TACON, the counter will increment until it reaches 3210H. At this point, the timer 1 interrupt request is generated, the counter value is reset, and counting resumes.

Timer 1 Control Register (TACON)

You use the timer 1 control register, TACON, to

— Enable the timer 1 operating (interval timer)

— Select the timer 1 input clock frequency

— Clear the timer 1 counter, TACNT and TBCNT

— Enable the timer 1 interrupt

TACON is located in page 0, at address BBH, and is read/write addressable using register addressing mode.

A reset clears TACON to "00H". This sets timer 1 to disable interval timer mode, selects an input clock frequency of fxx/512, and disables timer 1 interrupt. You can clear the timer 1 counter at any time during normal operation by writing a "1" to TACON.3.

To enable the timer 1 interrupt, you must write TACON.7, TACON.2, and TACON.1 to "1".
To generate the exact time interval, you should write TACON.3 and INTPND2.0, which cleared counter and interrupt pending bit. To detect an interrupt pending condition when T1INT is disabled, the application program polls pending bit, INTPND.2.0. When a "1" is detected, a timer 1 interrupt is pending. When the T1INT sub-routine has been serviced, the pending condition must be cleared by software by writing a "0" to the timer 1 interrupt pending bit, INTPND2.0.

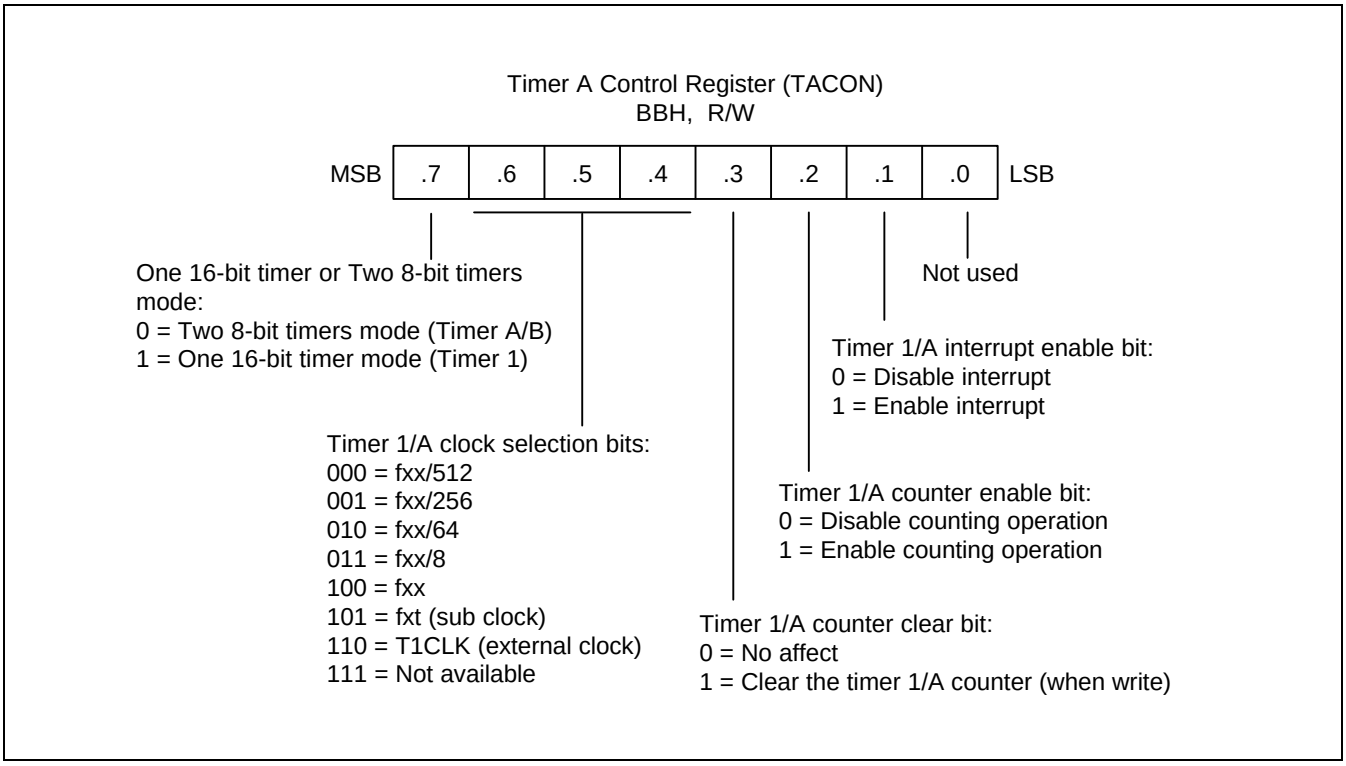


Figure 11-1. Timer 1 Control Register (TACON)

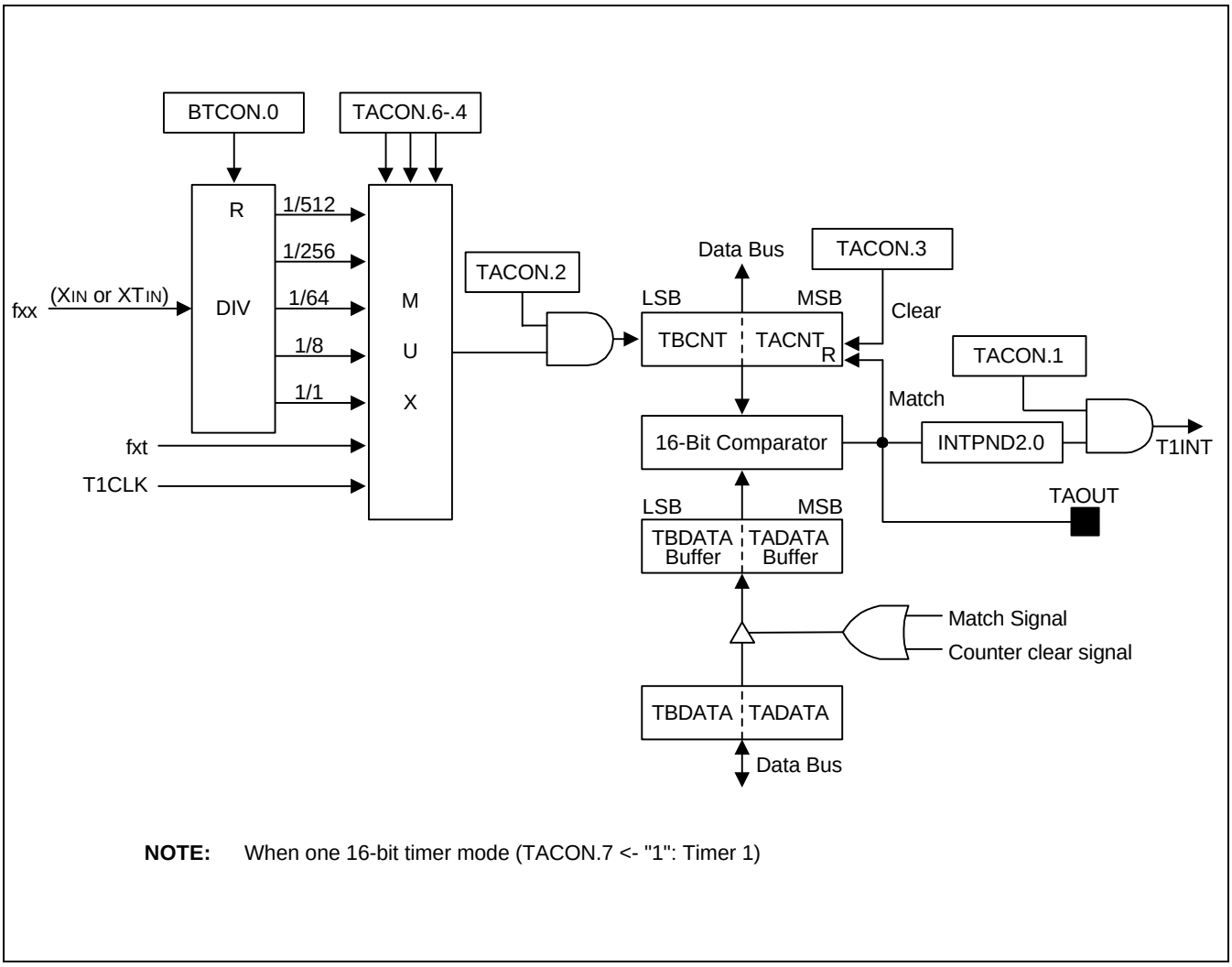


Figure 11-2. Timer 1 Block Diagram (One 16-bit Mode)

TWO 8-BIT TIMERS MODE (TIMER A and B)

OVERVIEW

The 8-bit timer A and B are the 8-bit general-purpose timers. Timer A and B have the interval timer mode by using the appropriate TACON and TBCON setting, respectively.

Timer A and B have the following functional components:

— Clock frequency divider with multiplexer
  – fxx divided by 512, 256, 64, 8 or 1, fxt, and T1CLK (External clock) for timer A
  – fxx divided by 512, 256, 64, 8 or 1, and fxt for timer B

— 8-bit counter (TACNT, TBCNT), 8-bit comparator, and 8-bit reference data register (TADATA, TBDATA)

— Timer A have I/O pin for match output (TAOUT)

— Timer A match interrupt generation

— Timer A control register, TACON (page 0, BBH, read/write)

— Timer B match interrupt generation

— Timer B control register, TBCON (page 0, BAH, read/write)

Timer A and B Control Register (TACON, TBCON)

You use the timer A and B control register, TACON and TBCON, to

— Enable the timer A (interval timer mode) and B operating (interval timer mode)

— Select the timer A and B input clock frequency

— Clear the timer A and B counter, TACNT and TBCNT

— Enable the timer A and B interrupt

TACON and TBCON are located in page 0, at address BBH and BAH, and is read/write addressable using register addressing mode.

A reset clears TACON to "00H". This sets timer A to disable interval timer mode, selects an input clock frequency of fxx/512, and disables timer A interrupt. You can clear the timer A counter at any time during normal operation by writing a "1" to TACON.3.

A reset clears TBCON to "00H". This sets timer B to disable interval timer mode, selects an input clock frequency of fxx/512, and disables timer A interrupt. You can clear the timer B counter at any time during normal operation by writing a "1" to TBCON.3.

To enable the timer A interrupt (TAINT) and timer B interrupt (TBINT), you must write TACON.7 to "0", TACON.2 (TBCON.2) and TACON.1 (TBCON.1) to "1". To generate the exact time interval, you should write TACON.3 (TBCON.3) and INTPND2.0 (INTPND2.1), which cleared counter and interrupt pending bit. To detect an interrupt pending condition when TAINT and TBINT is disabled, the application program polls pending bit, INTPND2.0 and INTPND2.1. When a "1" is detected, a timer A interrupt (TAINT) and timer B interrupt (TBINT) is pending. When the TAINT and TBINT sub-routine has been serviced, the pending condition must be cleared by software by writing a "0" to the timer A and B interrupt pending bit, INTPND2.0 and INTPND2.1.

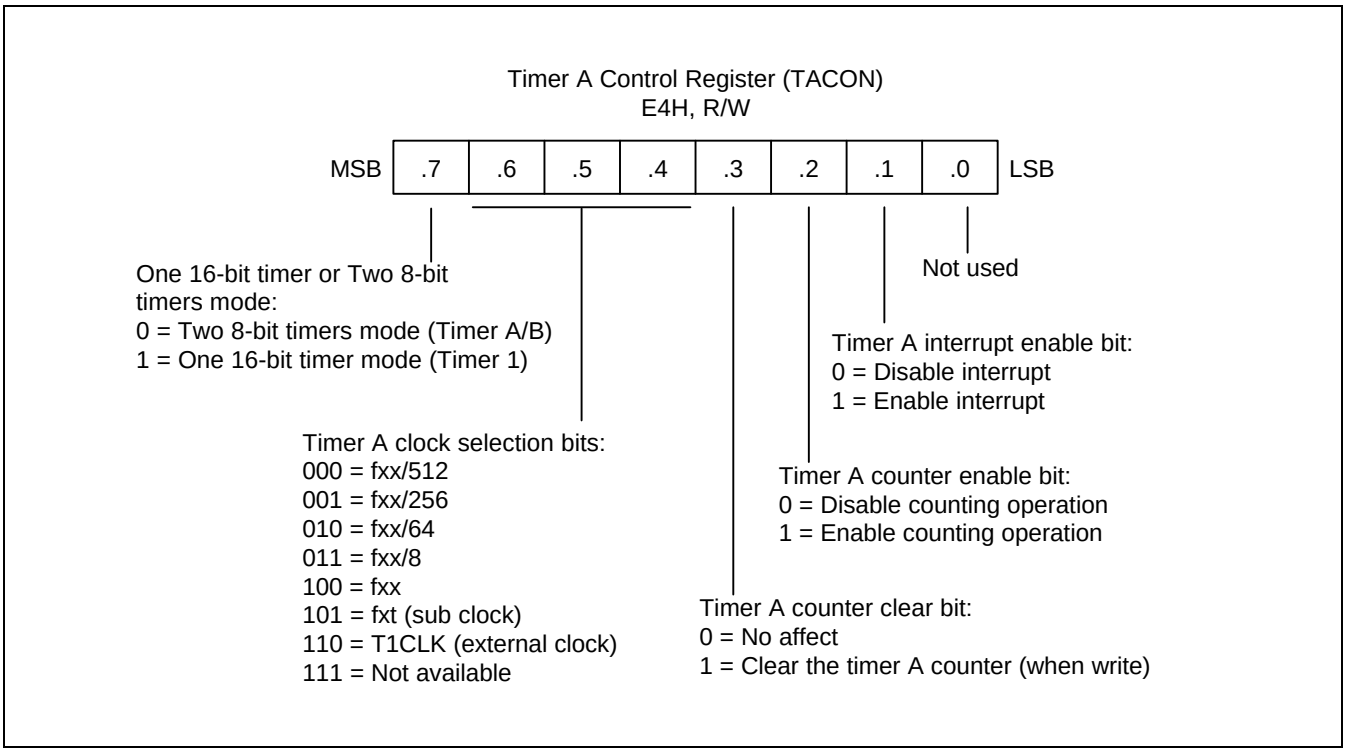


Figure 11-3. Timer A Control Register (TACON)

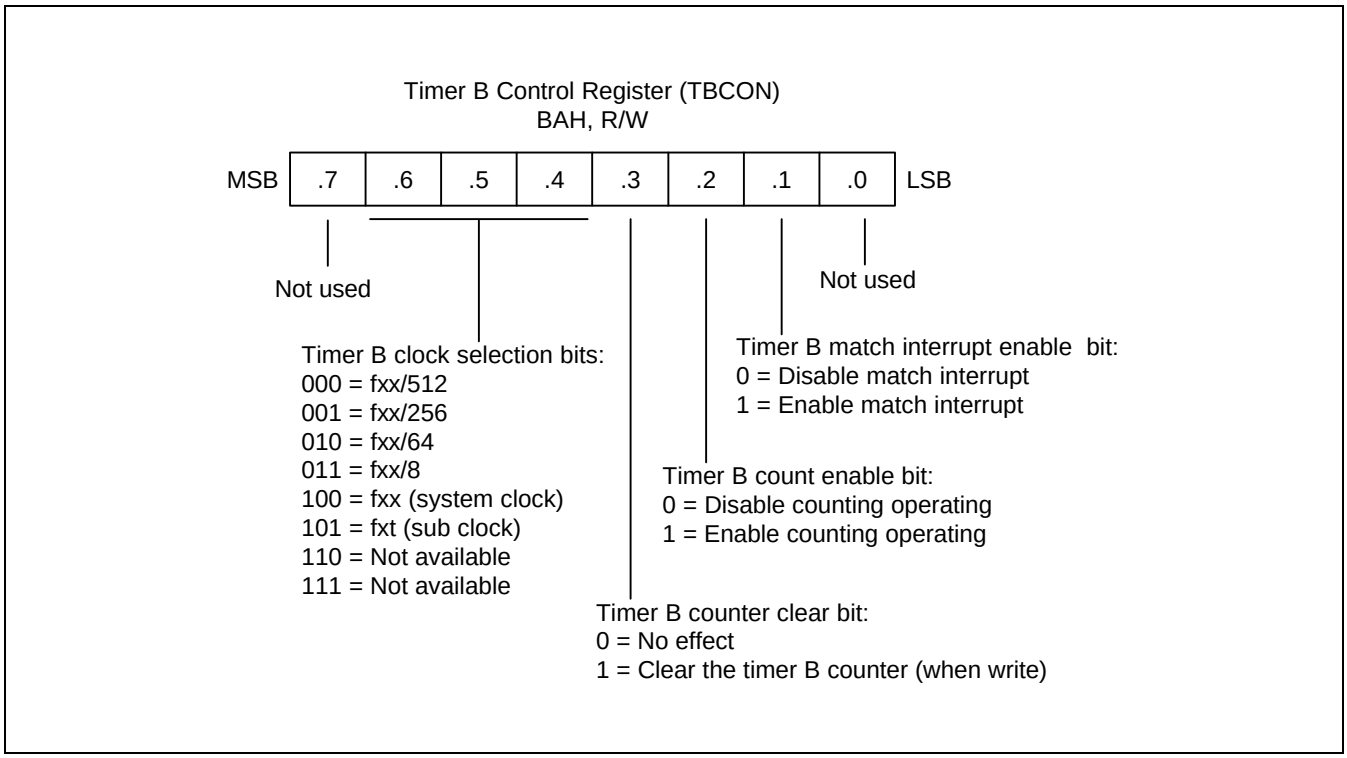


Figure 11-4. Timer B Control Register (TBCON)

FUNCTION DESCRIPTION

Interval Timer Function (Timer A and Timer B)

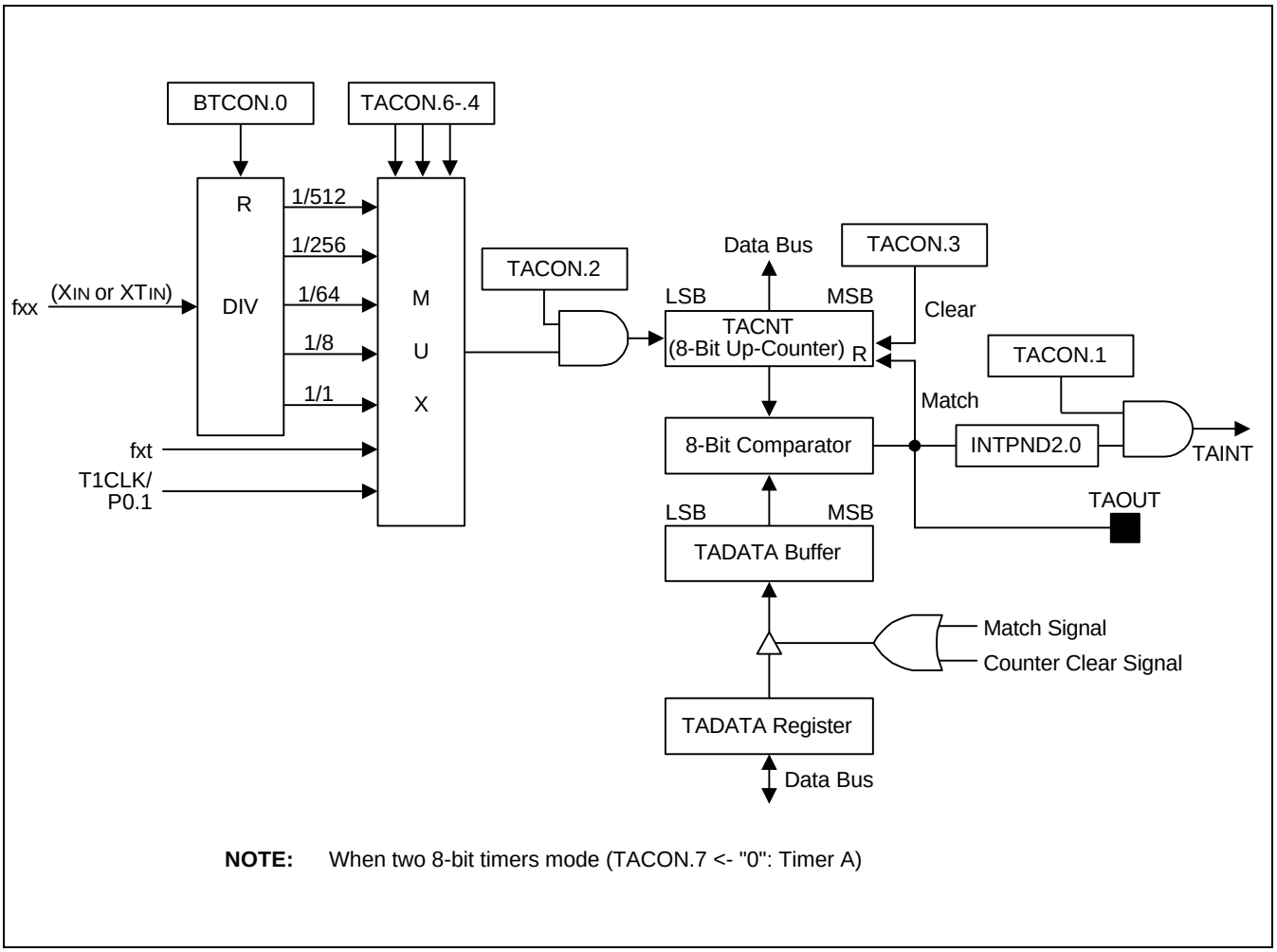
The timer A and B module can generate an interrupt: the timer A match interrupt (TAINT) and the timer B match interrupt (TBINT).

The timer A match interrupt pending condition (INTPND2.0) and the timer B match interrupt pending condition (INTPND2.1) must be cleared by software in the application's interrupt service by means of writing a "0" to the INTPND2.0 and INTPND2.1 interrupt pending bit.

Even though TAINT and TBINT are disabled, the application's service routine can detect a pending condition of TAINT and TBINT by the software and execute it's sub-routine. When this case is used, the TAINT and TBINT pending bit must be cleared by the application sub-routine by writing a "0" to the corresponding pending bit INTPND2.0 and INTPND2.1.

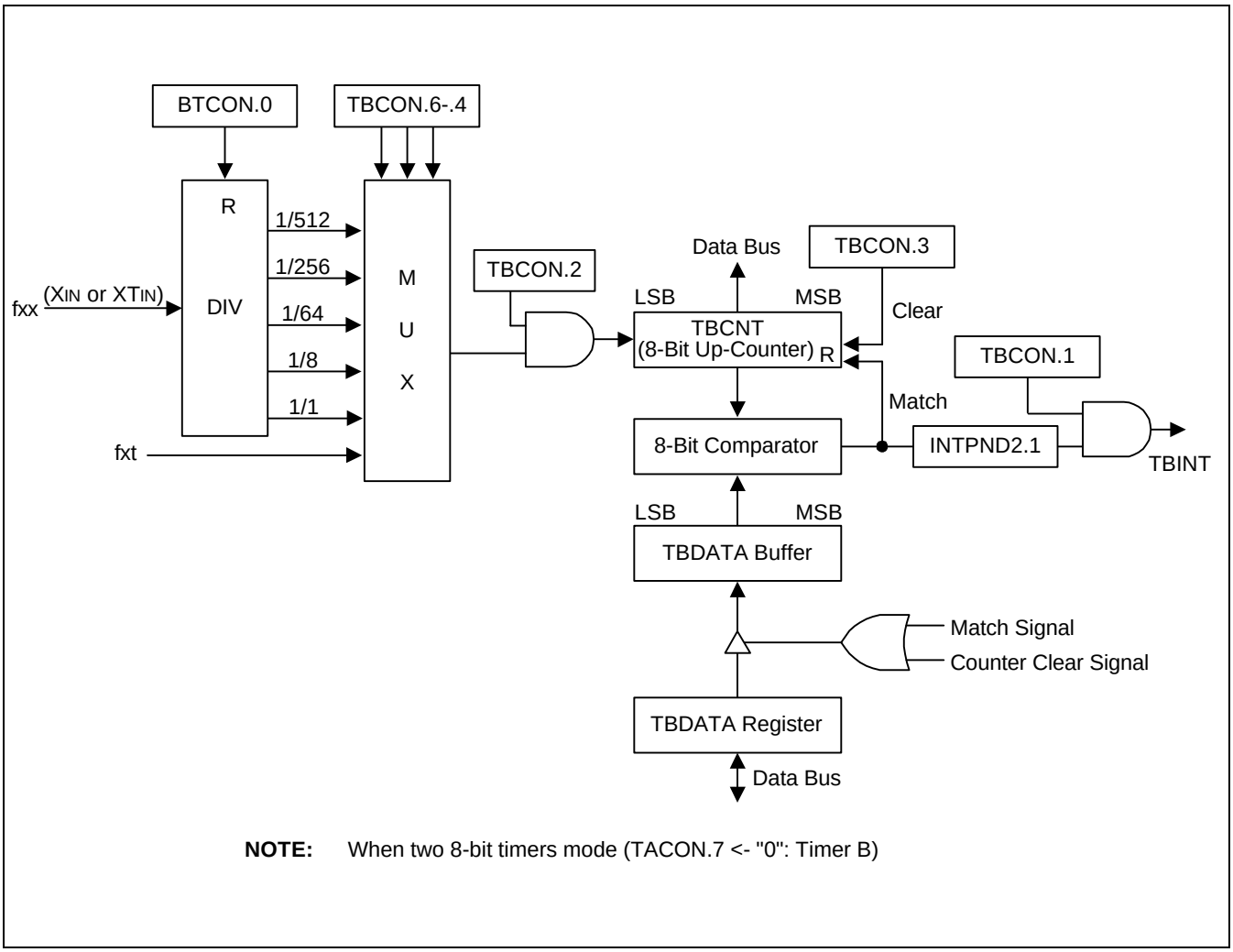
In interval timer mode, a match signal is generated when the counter value is identical to the values written to the timer A or timer B reference data registers, TADATA or TBDATA. The match signal generates corresponding match interrupt and clears the counter.

If, for example, you write the value 20H to TADATA and 0EH to TACON, the counter will increment until it reaches 20H. At this point, the timer A interrupt request is generated, the counter value is cleared, and counting resumes and you write the value 10H to TBDATA, "0" to TACON.7, and 0EH to TBCON, the counter will increment until it reaches 10H. At this point, TB interrupt request is generated, the counter value is cleared and counting resumes.



NOTE: When two 8-bit timers mode (TACON.7 <- "0": Timer A)

Figure 11-5. Timer A Block Diagram(Two 8-bit Timers Mode)



NOTE: When two 8-bit timers mode (TACON.7 <- "0": Timer B)

Figure 11-6. Timer B Block Diagram (Two 8-bit Timers Mode)

NOTES

12 WATCH TIMER

OVERVIEW

Watch timer functions include real-time and watch-time measurement and interval timing for the system clock. To start watch timer operation, set bit 1 of the watch timer control register, WTCON.1 to "1".
And if you want to service watch timer overflow interrupt, then set the WTCON.6 to “1”.
The watch timer overflow interrupt pending condition (INTPND2.3) must be cleared by software in the application's interrupt service routine by means of writing a "0" to the INTPND2.3 interrupt pending bit.
After the watch timer starts and elapses a time, the watch timer interrupt pending bit (INTPND2.3) is automatically set to "1", and interrupt requests commence in 3.91ms, 0.25, 0.5 and 1-second intervals by setting Watch timer speed selection bits (WTCON.3 – .2).

The watch timer can generate a steady 0.5 kHz, 1 kHz, 2 kHz, or 4 kHz signal to BUZ output pin for Buzzer. By setting WTCON.3 and WTCON.2 to "11b", the watch timer will function in high-speed mode, generating an interrupt every 3.91 ms. High-speed mode is useful for timing events for program debugging sequences.

Also, you can select watch timer clock source by setting the WTCON.7 appropriately value.

The watch timer supplies the clock frequency for the LCD controller (f_{LCD}). Therefore, if the watch timer is disabled, the LCD controller does not operate.

Watch timer has the following functional components:

— Real Time and Watch-Time Measurement
— Using a Main or Sub Clock Source (Main clock divided by 2^7(fx/128) or Sub clock(fxt))
— Clock Source Generation for LCD Controller (f_{LCD})
— I/O pin for Buzzer Output Frequency Generator (P0.3, BUZ)
— Timing Tests in High-Speed Mode
— Watch timer overflow interrupt generation
— Watch timer control register, WTCON (page 0, DAH, read/write)

WATCH TIMER CONTROL REGISTER (WTCON)

The watch timer control register, WTCON is used to select the input clock source, the watch timer interrupt time and Buzzer signal, to enable or disable the watch timer function. It is located in page 0 at address DAH, and is read/write addressable using register addressing mode.
A reset clears WTCON to "00H". This disable the watch timer and select fx/128 as the watch timer clock.
So, if you want to use the watch timer, you must write appropriate value to WTCON.

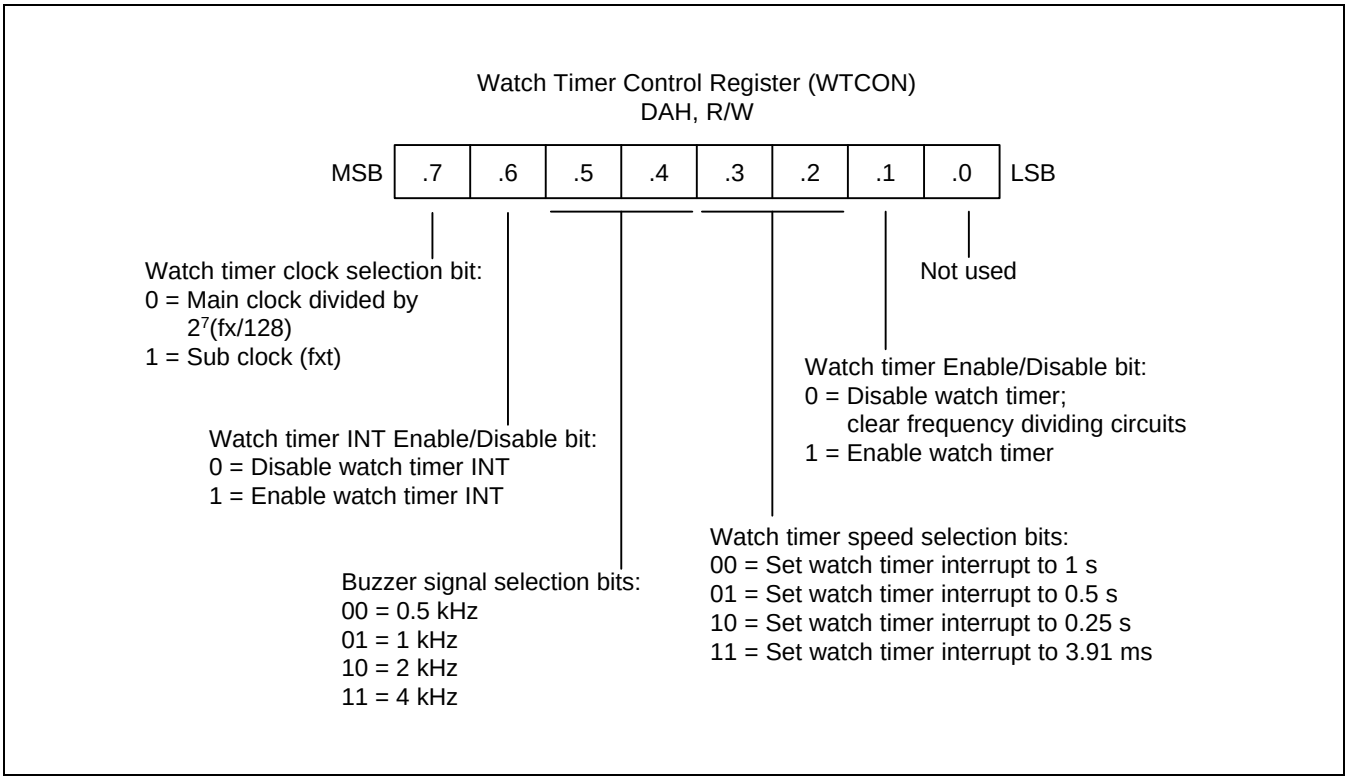


Figure 12-1. Watch Timer Control Register (WTCON)

WATCH TIMER CIRCUIT DIAGRAM

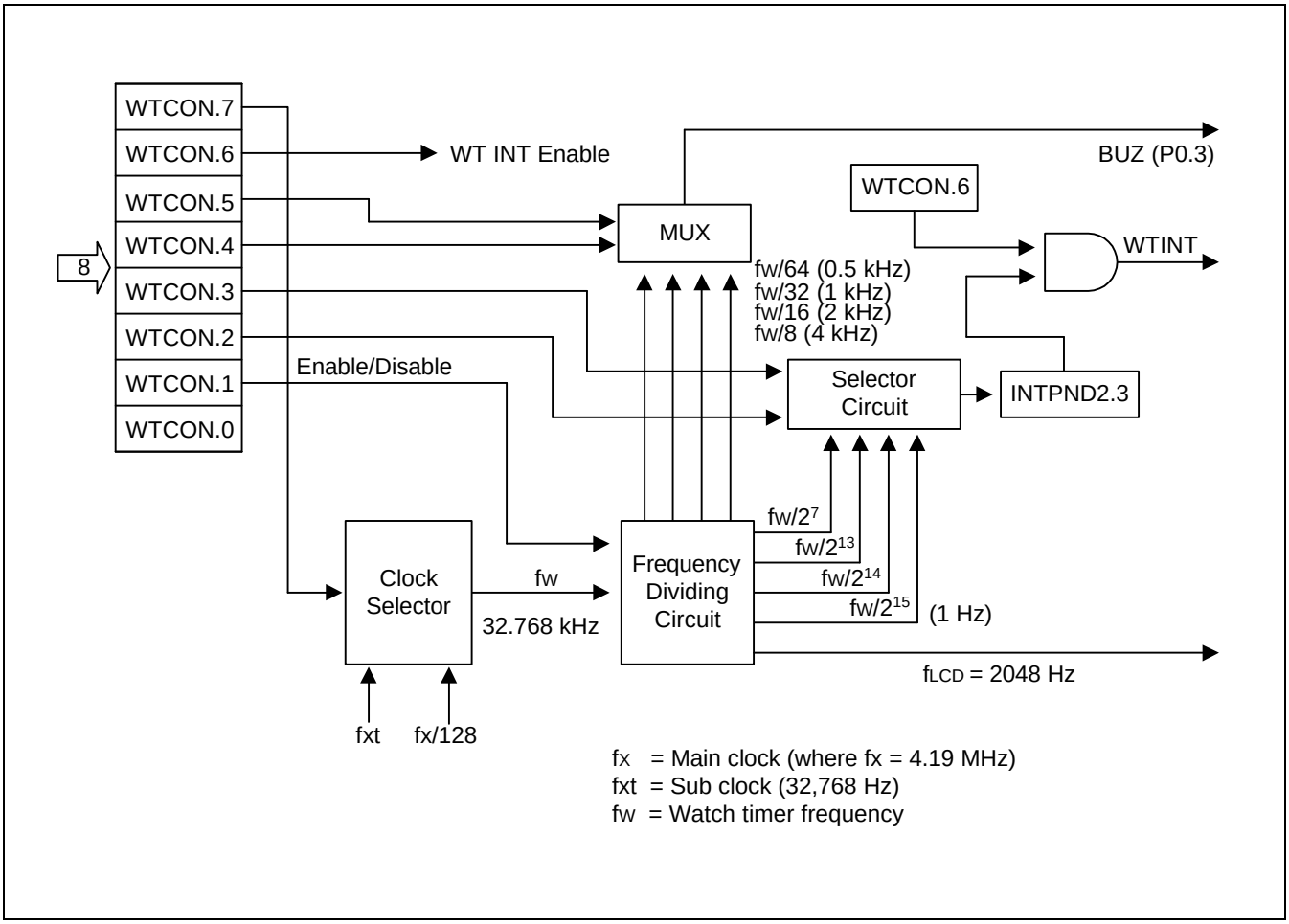


Figure 12-2. Watch Timer Circuit Diagram

NOTES

13 LCD CONTROLLER/DRIVER

OVERVIEW

The S3C9228/P9228 microcontroller can directly drive an up-to-128-dot (16segments x 8 commons) LCD panel. Its LCD block has the following components:

— LCD controller/driver
— Display RAM for storing display data
— 16 segment output pins (SEG0–SEG15)
— 8 common output pins (COM0–COM7)
— Internal resistor circuit for LCD bias

To use the LCD controller, bit 2 in the watch mode register WMOD must be set to 1 because LCDCK is supplied by the watch timer.

The LCD mode control register, LMOD, is used to turn the LCD display on or off, to select LCD clock frequency, to turn the COM signal output on or off, to select bias and duty, and to switch the port 3 high impedance or normal I/O port. Data written to the LCD display RAM can be transferred to the segment signal pins automatically without program control.

The LCD port control register, LPOT, is used to determine the LCD signal pins used for display output.

When a sub clock is selected as the LCD clock source, the LCD display is enabled even during main clock stop and idle modes.

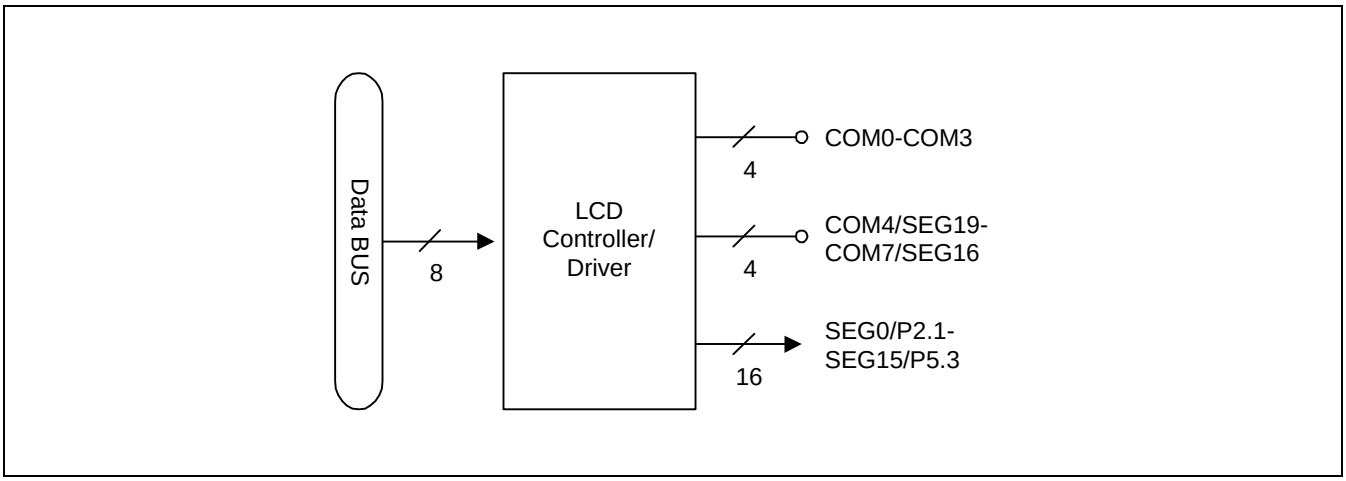


Figure 13-1. LCD Function Diagram

LCD CIRCUIT DIAGRAM

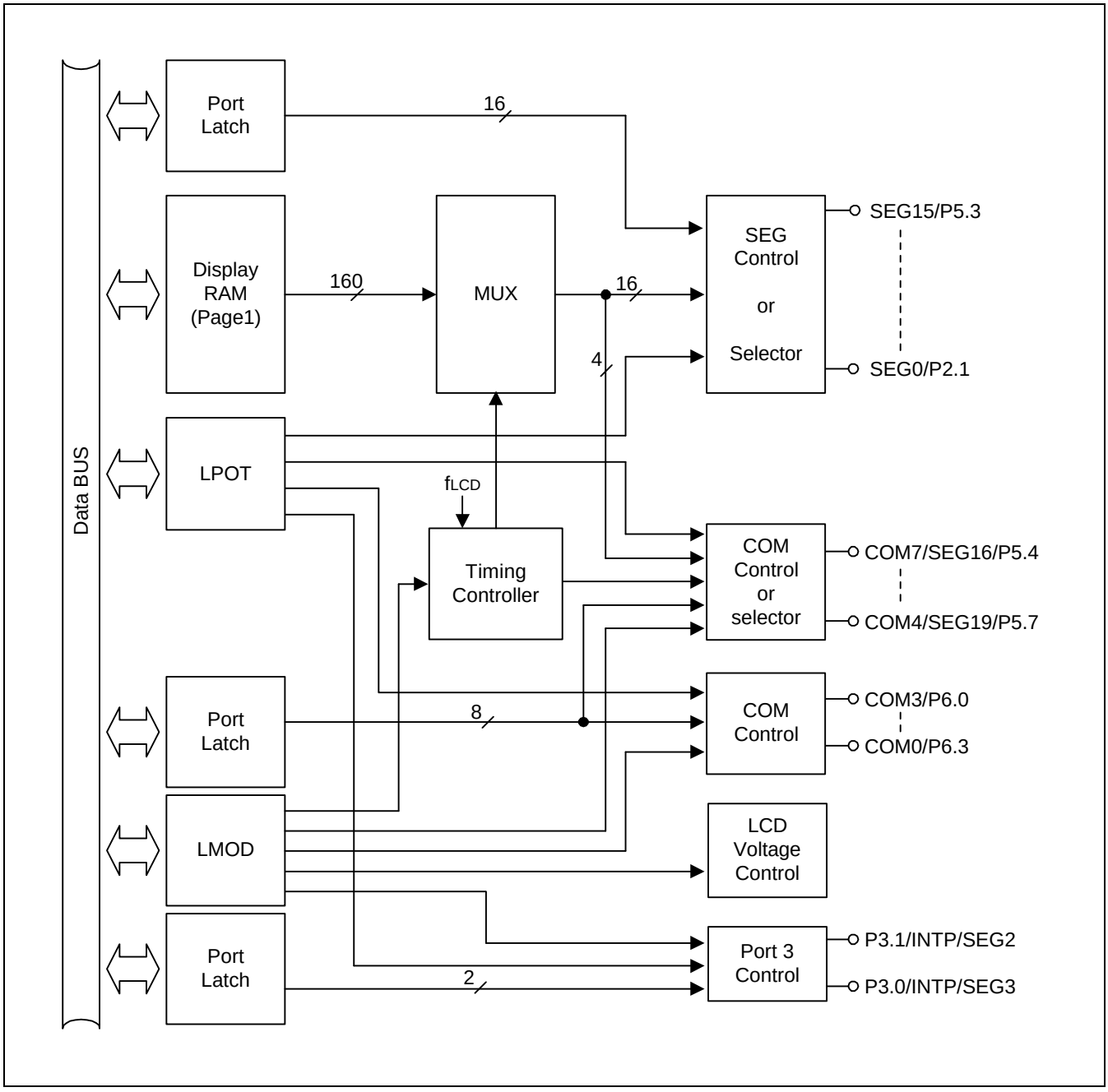


Figure 13-2. LCD Circuit Diagram

LCD RAM ADDRESS AREA

RAM addresses of page 1 are used as LCD data memory. When the bit value of a display segment is "1", the LCD display is turned on; when the bit value is "0", the display is turned off.

Display RAM data are sent out through segment pins SEG0–SEG19 using a direct memory access (DMA) method that is synchronized with the f_{LCD} signal. RAM addresses in this location that are not used for LCD display can be allocated to general-purpose use.

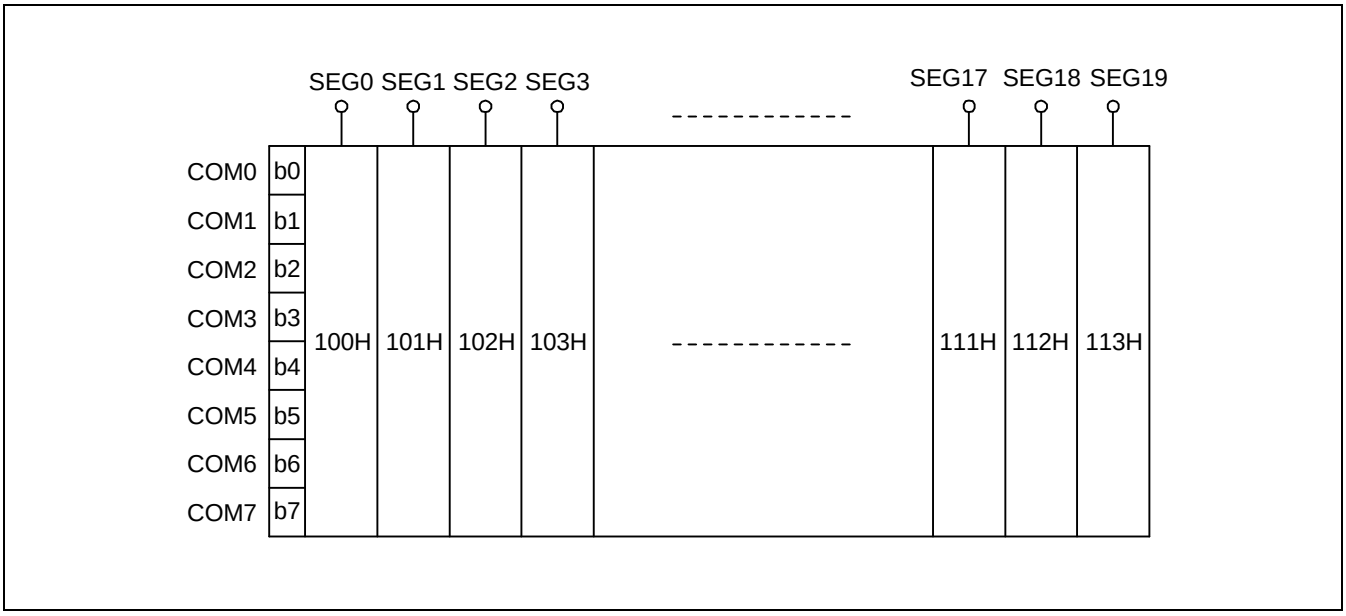


Figure 13-3. LCD Display Data RAM Organization

Table 13-1. Common and Segment Pins per Duty Cycle

Duty	Common Pins	Segment Pins	Dot Number
1/8	COM0–COM7	16 pins	128 dots
1/4	COM0–COM3	20 pins	80 dots
1/3	COM0–COM2	20 pins	60 dots

LCD MODE CONTROL REGISTER (LMOD)

A LMOD is located in page 0, at address FEH, and is read/write addressable using register addressing mode. It has the following control functions.

— LCD duty and bias selection
— LCD clock selection
— LCD display control
— COMs signal output control
— P3 high impedance control

The LMOD register is used to turn the LCD display on/off, to select duty and bias, to select LCD clock, to control port 3 high impedance/normal I/O port, and to turn the COM signal output on/off. Following a RESET, all LMOD values are cleared to "0". This turns off the LCD display, select 1/3 duty and 1/3 bias, and select 256Hz for LCD clock.

The LCD clock signal determines the frequency of COM signal scanning of each segment output. This is also referred as the LCD frame frequency. Since the LCD clock is generated by watch timer clock (fw). The watch timer should be enabled when the LCD display is turned on.

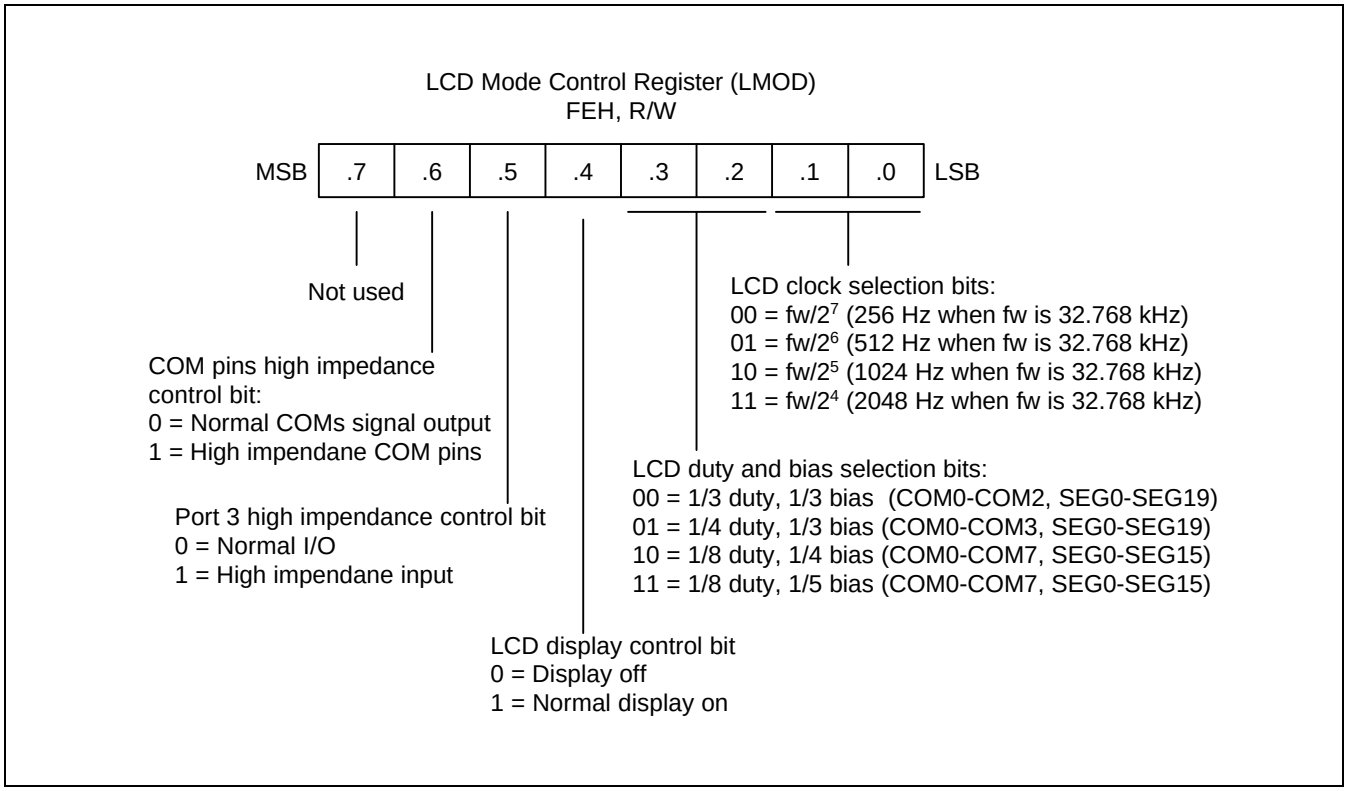


Figure 13-4. LCD Mode Control Register (LMOD)

LCD PORT CONTROL REGISTER

The LCD port control register LPOT is used to control LCD signal pins or normal I/O pins. Following a RESET, a LPOT values are cleared to "0".

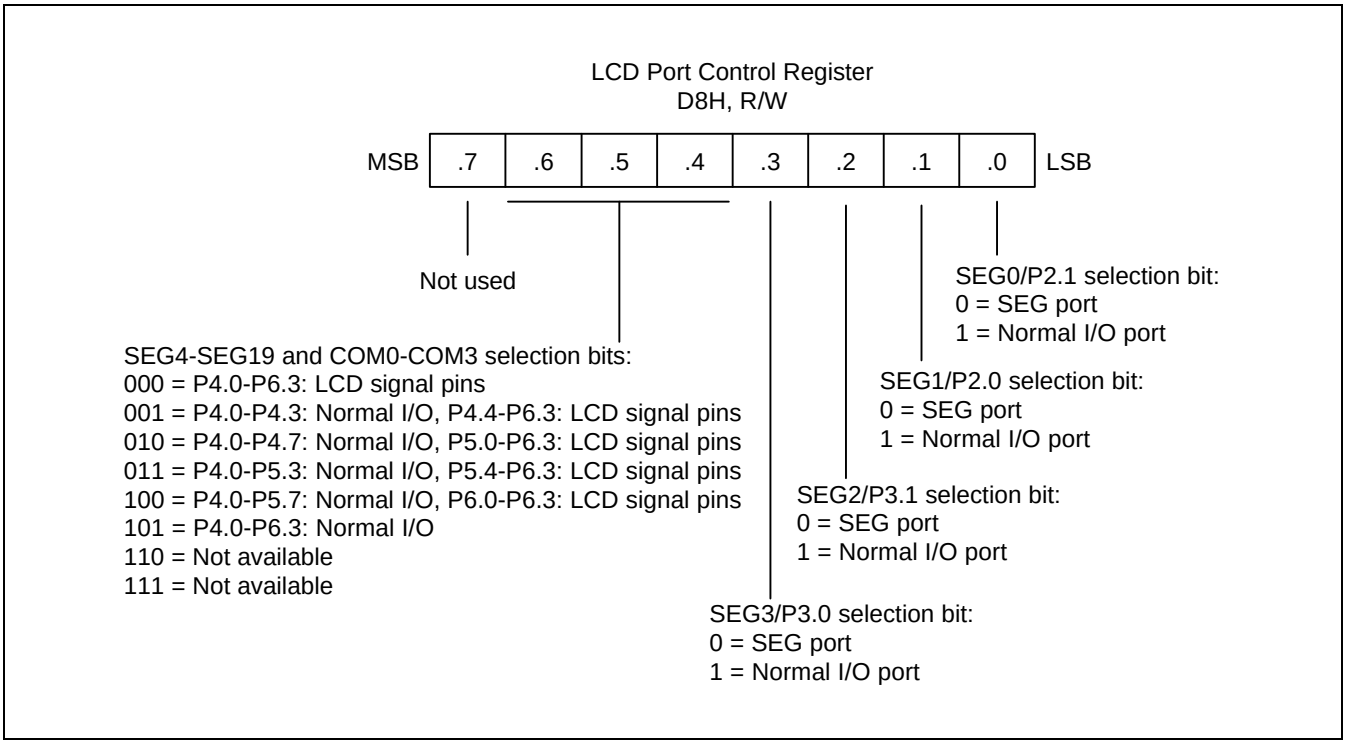


Figure 13-5. LCD Port Control Register

LCD VOLTAGE DIVIDING RESISTORS

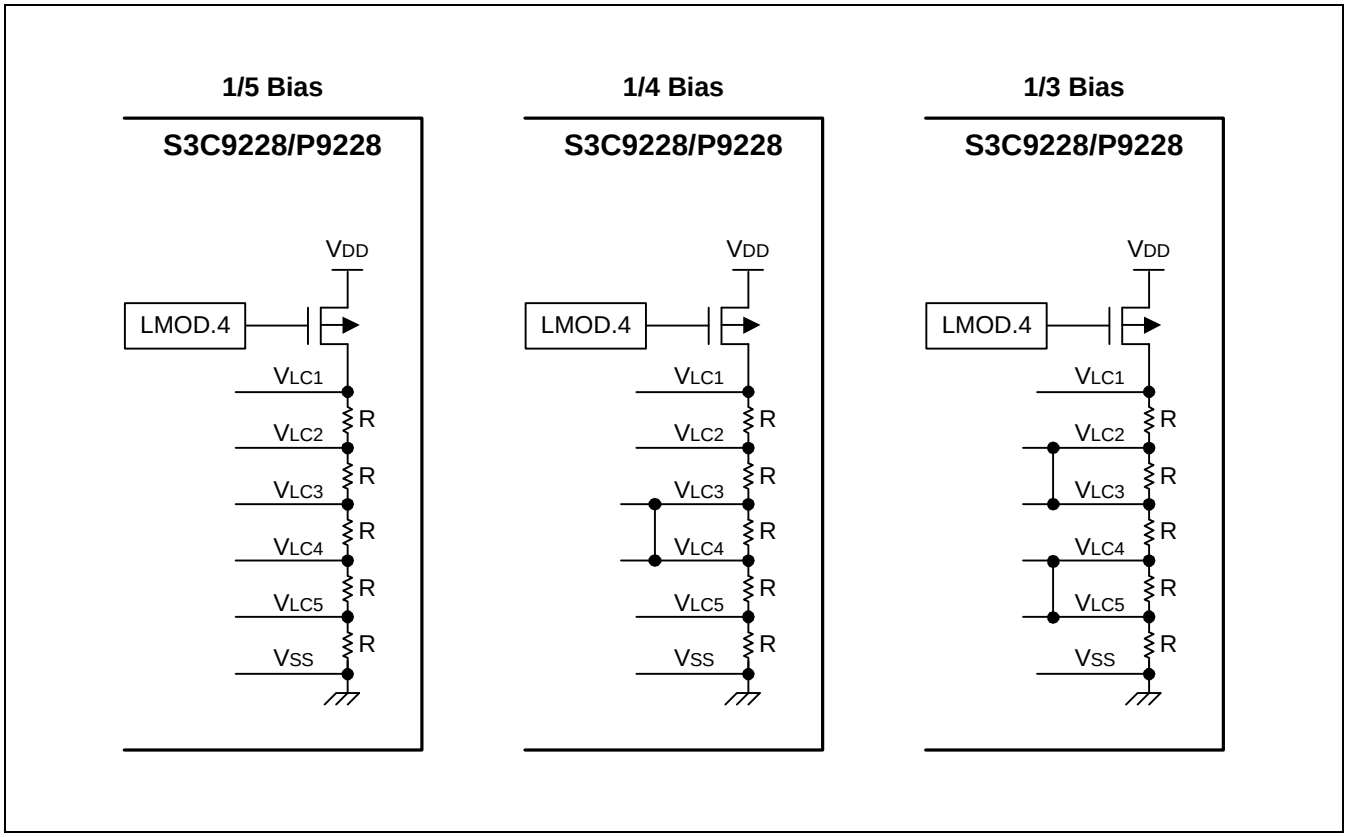


Figure 13-6. Internal Voltage Dividing Resistor Connection

COMMON (COM) SIGNALS

The common signal output pin selection (COM pin selection) varies according to the selected duty cycle.

— In 1/3 duty mode, COM0-COM2 pins are selected
— In 1/4 duty mode, COM0-COM3 pins are selected
— In 1/8 duty mode, COM0-COM7 pins are selected

SEGMENT (SEG) SIGNALS

The 19 LCD segment signal pins are connected to corresponding display RAM locations at page 1. Bits of the display RAM are synchronized with the common signal output pins.

When the bit value of a display RAM location is "1", a select signal is sent to the corresponding segment pin. When the display bit is "0", a 'no-select' signal to the corresponding segment pin.

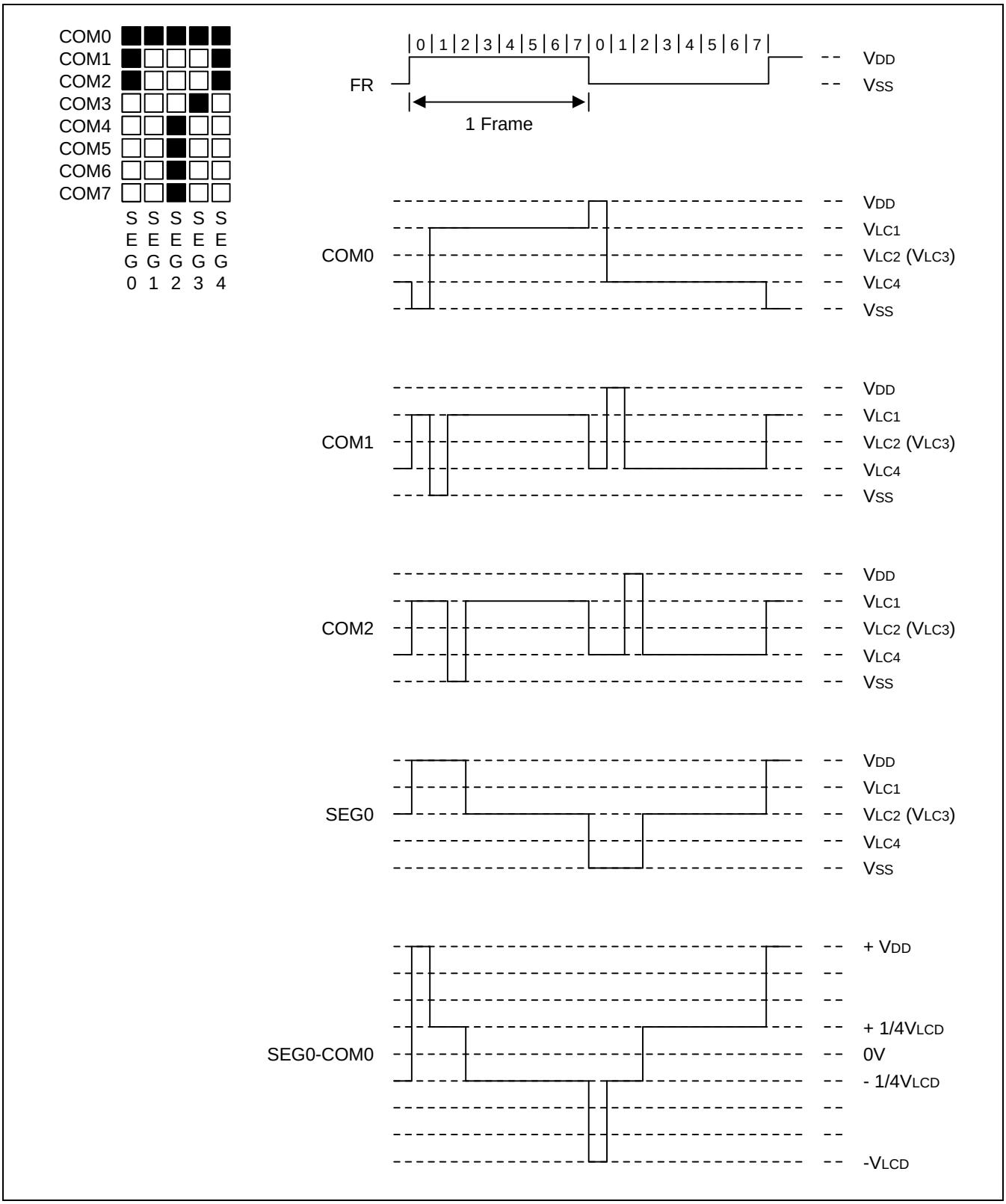


Figure 13-7. LCD Signal Waveforms (1/8 Duty, 1/4 Bias)

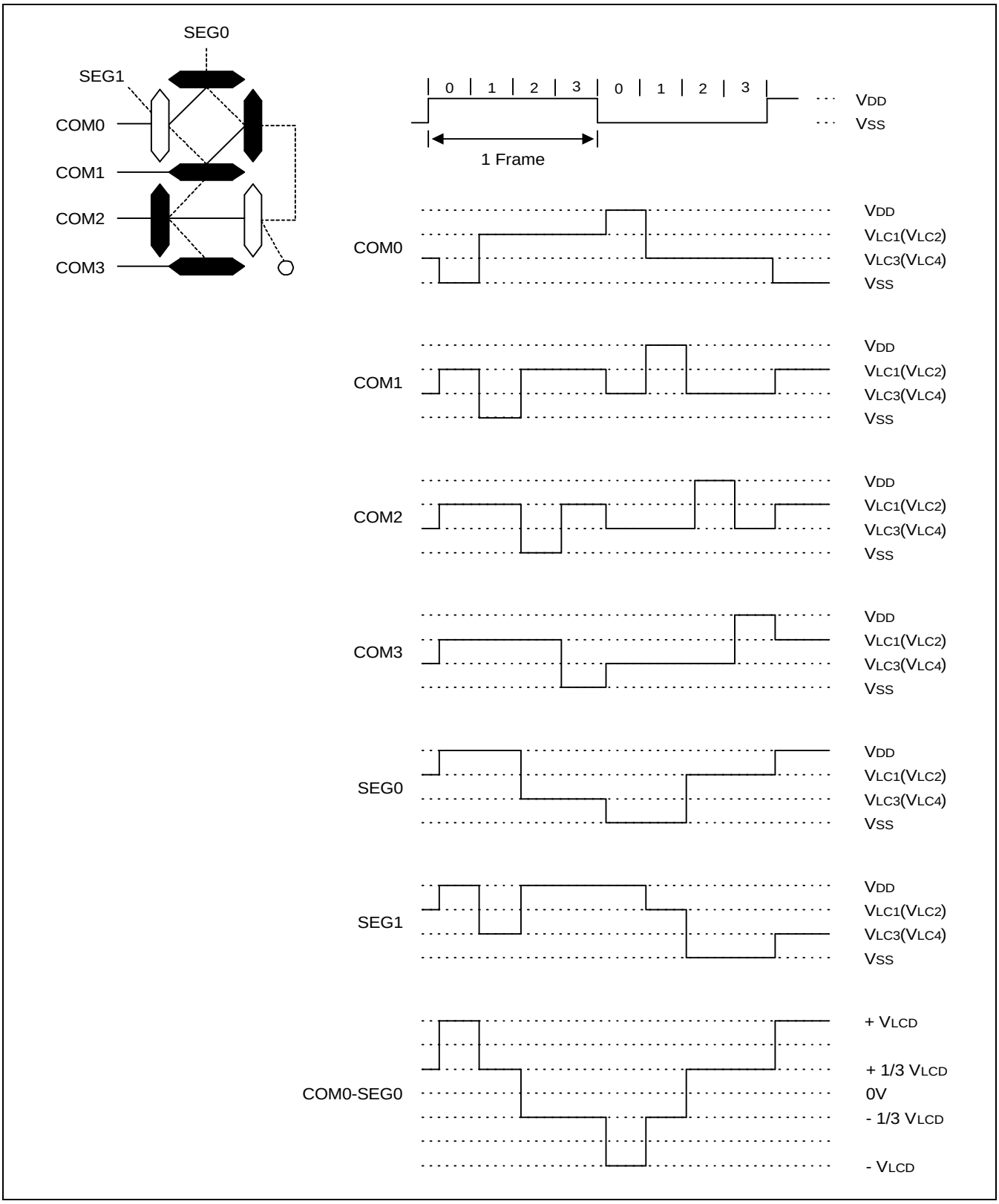


Figure 13-8. LCD Signal Waveforms (1/4 Duty, 1/3 Bias)

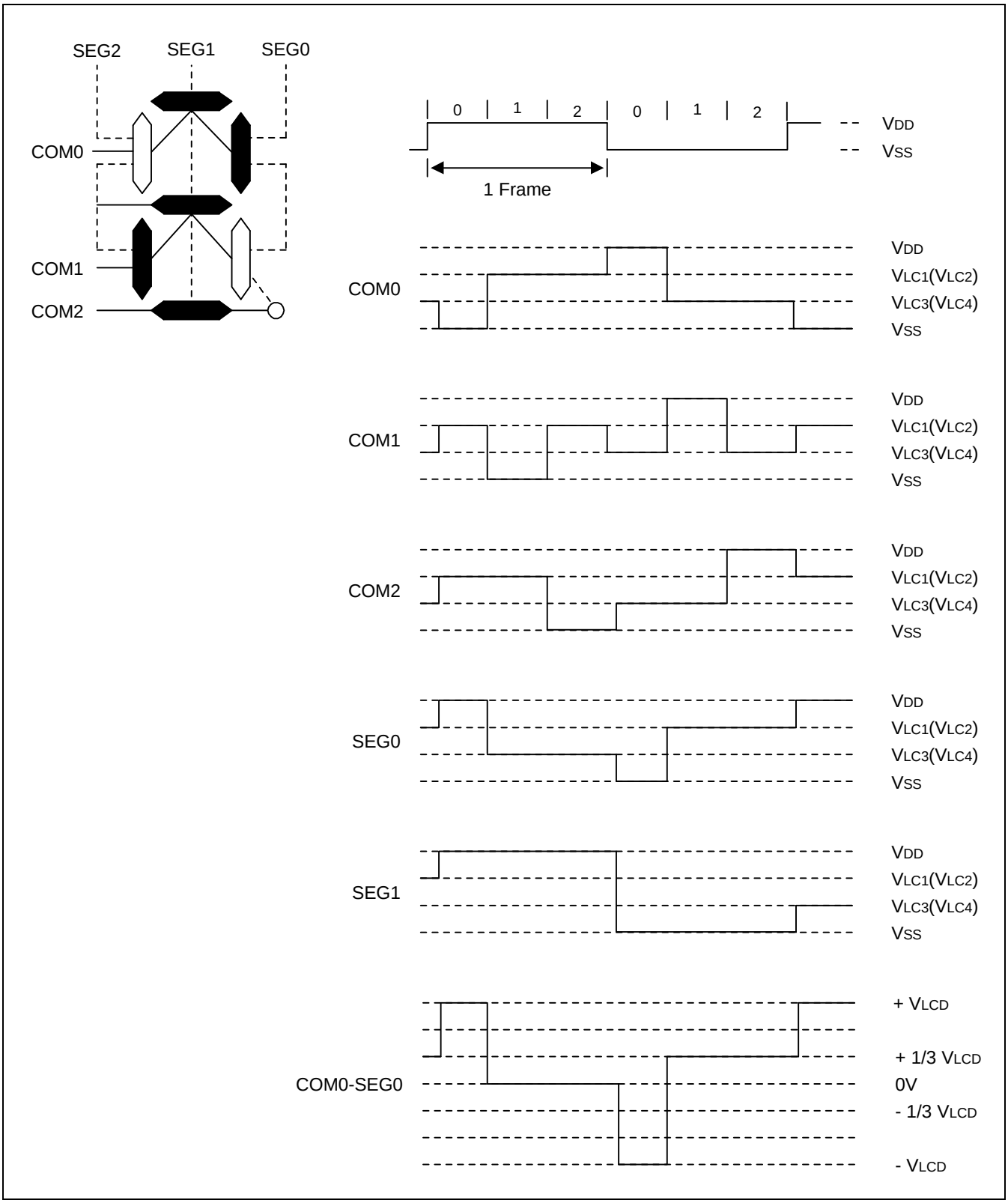


Figure 13-9. LCD Signal Waveforms (1/3 Duty, 1/3 Bias)

NOTES

14 10-BIT ANALOG-TO-DIGITAL CONVERTER

OVERVIEW

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

— Analog comparator with successive approximation logic

— D/A converter logic (resistor string type)

— ADC control register (ADCON)

— Four multiplexed analog data input pins (AD0–AD3)

— 10-bit A/D conversion data output register (ADDATAH/ADDATAL)

— 4-bit digital input port (Alternately, I/O port)

FUNCTION DESCRIPTION

To initiate an analog-to-digital conversion procedure, at first you must set with alternative function for ADC input enable at port 1, the pin set with alternative function can be used for ADC analog input. And you write the channel selection data in the A/D converter control register ADCON.4–.5 to select one of the four analog input pins (AD0–3) and set the conversion start or enable bit, ADCON.0. The read-write ADCON register is located in page 0, at address D0H. The pins which are not used for ADC can be used for normal I/O.

During a normal conversion, ADC logic initially sets the successive approximation register to 800H (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.5–4) 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 ADDATAH/ADDATAL register where it can be read. The A/D converter then enters an idle state. Remember to read the contents of ADDATAH/ADDATAL before another conversion starts. Otherwise, the previous result will be overwritten by the next conversion result.

NOTE

Because the A/D converter has no sample-and-hold circuitry, it is very important that fluctuation in the analog level at the AD0–AD3 input pins during a conversion procedure be kept to an absolute minimum. Any change in the input level, perhaps due to noise, will invalidate the result. If the chip enters to STOP or IDLE mode in conversion process, there will be a leakage current path in A/D block. You must use STOP or IDLE mode after ADC operation is finished.

CONVERSION TIMING

The A/D conversion process requires 4 steps (4 clock edges) to convert each bit and 10 clocks to set-up A/D conversion. Therefore, total of 50 clocks are required to complete an 10-bit conversion: When fxx/8 is selected for conversion clock with an 4.5 MHz fxx clock frequency, one clock cycle is 1.78 us. Each bit conversion requires 4 clocks, the conversion rate is calculated as follows:

4 clocks/bit × 10-bit + set-up time = 50 clocks, 50 clock × 1.78 us = 89 us at 0.56 MHz (4.5 MHz/8)

Note that A/D converter needs at least 25μs for conversion time.

A/D CONVERTER CONTROL REGISTER (ADCON)

The A/D converter control register, ADCON, is located at address D0H in page 0. It has three functions:

— Analog input pin selection (bits 4 and 5)

— End-of-conversion status detection (bit 3)

— ADC clock selection (bits 2 and 1)

— A/D operation start or enable (bit 0)

After a reset, the start bit is turned off. You can select only one analog input channel at a time. Other analog input pins (AD0–AD3) can be selected dynamically by manipulating the ADCON.4–5 bits. And the pins not used for analog input can be used for normal I/O function.

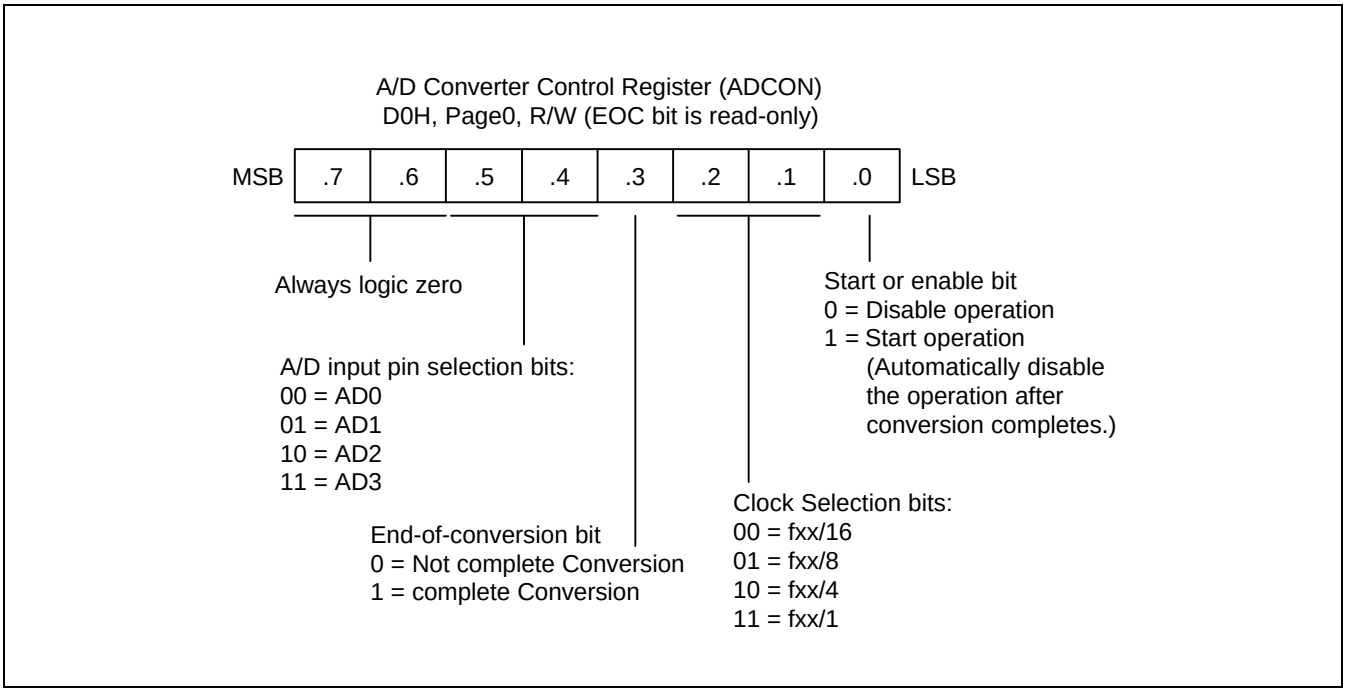


Figure 14-1. A/D Converter Control Register (ADCON)

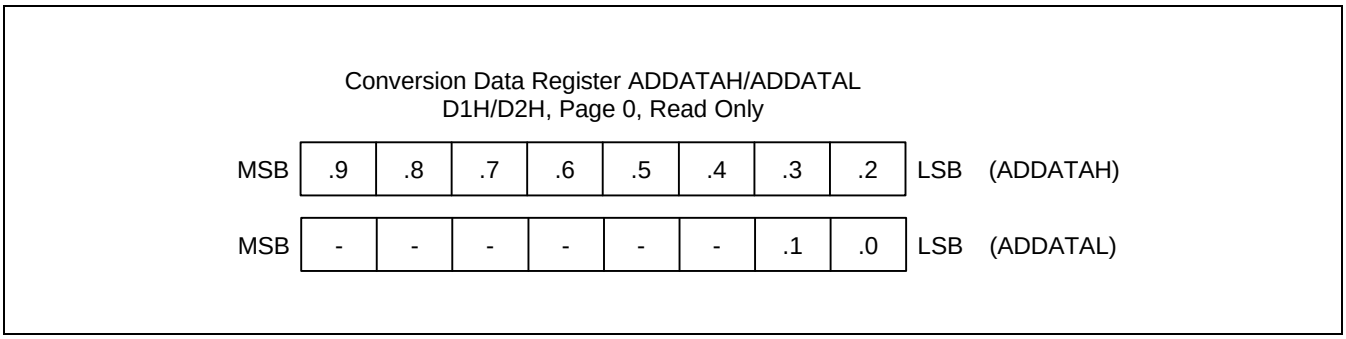


Figure 14-2. A/D Converter Data Register (ADDATAH/ADDATAL)

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 V_{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 conversion bit is always 1/2 V_{DD}.

BLOCK DIAGRAM

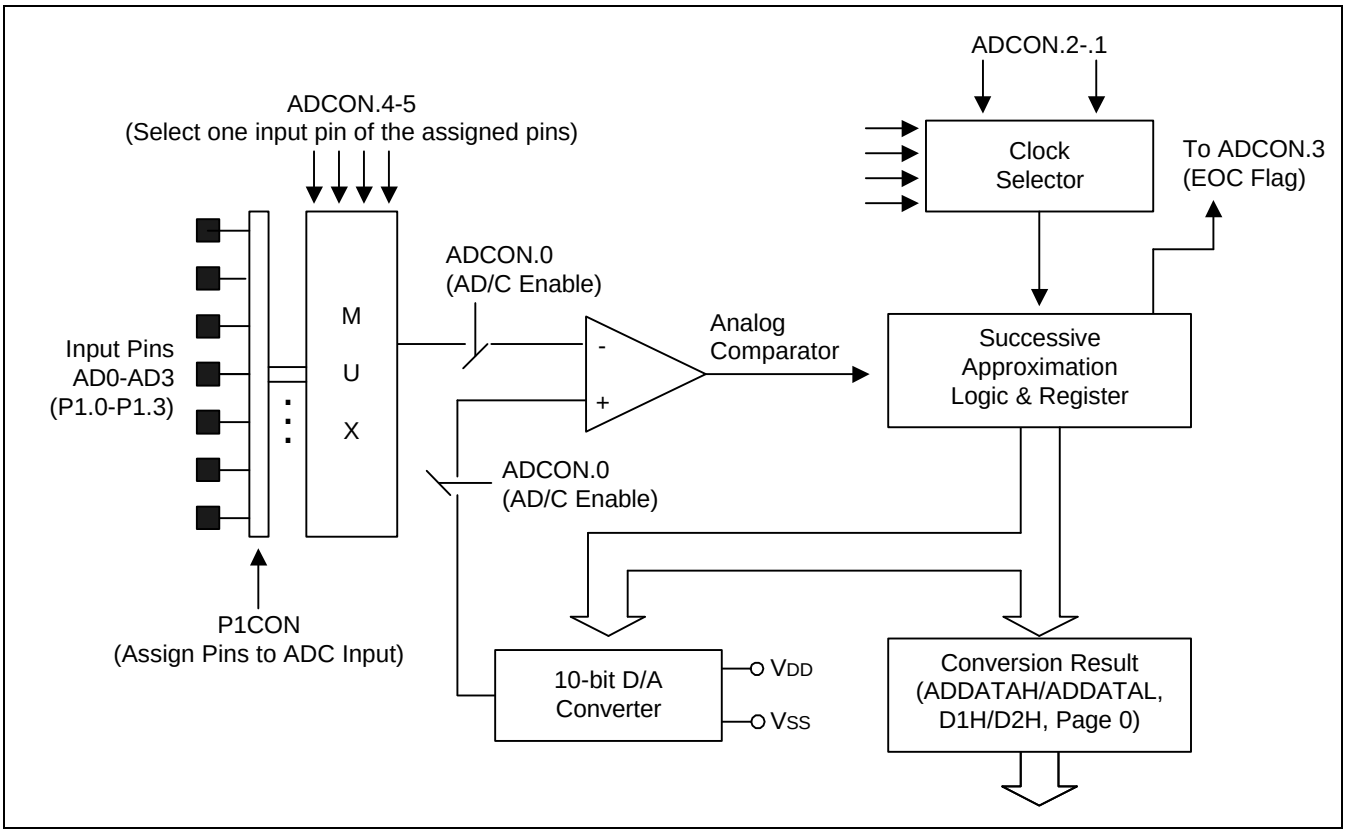


Figure 14-3. A/D Converter Functional Block Diagram

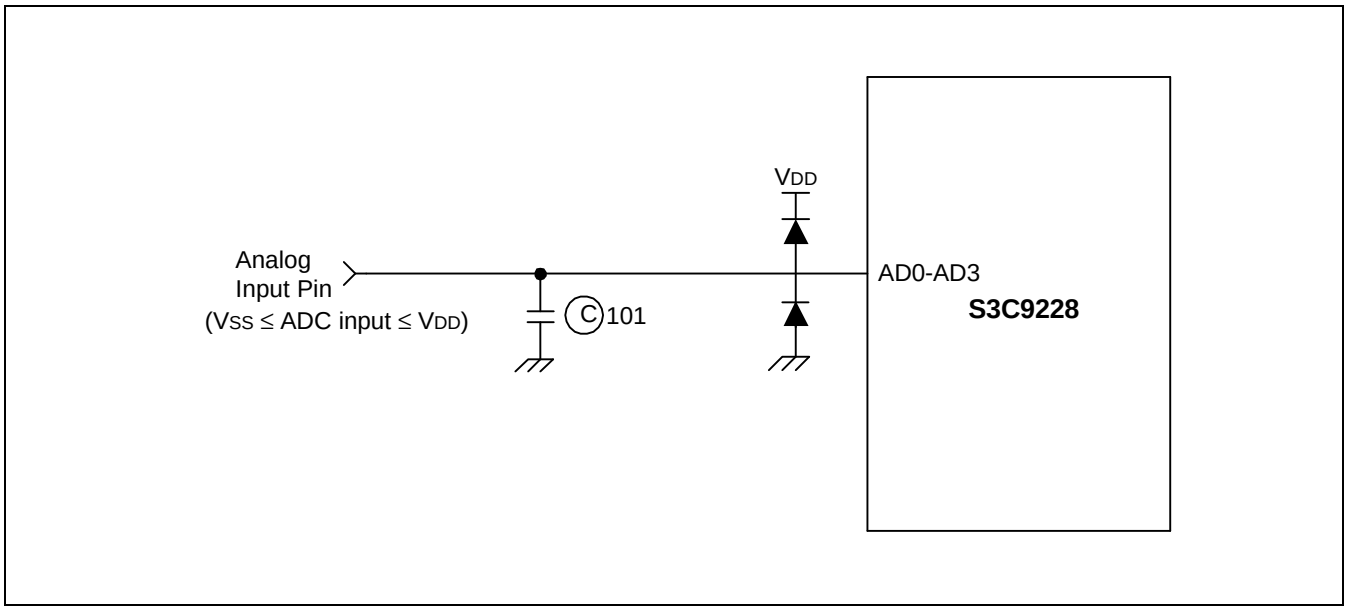


Figure 14-4. Recommended A/D Converter Circuit for Highest Absolute Accuracy

15 SERIAL I/O INTERFACE

OVERVIEW

Serial I/O modules, SIO can interface with various types of external device that require serial data transfer. The components of SIO function block are:

— 8-bit control register (SIOCON)
— Clock selector logic
— 8-bit data buffer (SIODATA)
— 8-bit prescaler (SIOPS)
— 3-bit serial clock counter
— Serial data I/O pins (SI, SO)
— Serial clock input/output pin (SCK)

The SIO module can transmit or receive 8-bit serial data at a frequency determined by its corresponding control register settings. To ensure flexible data transmission rates, you can select an internal or external clock source.

PROGRAMMING PROCEDURE

To program the SIO module, follow these basic steps:

1. Configure the I/O pins at port (SCK/SI/SO) by loading the appropriate value to the P2CON register if necessary.
2. Load an 8-bit value to the SIOCON control register to properly configure the serial I/O module. In this operation, SIOCON.2 must be set to "1" to enable the data shifter.
3. For interrupt generation, set the serial I/O interrupt enable bit (SIOCON) to "1".
4. When you transmit data to the serial buffer, write data to SIODATA and set SIOCON.3 to 1, the shift operation starts.
5. When the shift operation (transmit/receive) is completed, the SIO pending bit (INTPND2.2) are set to "1" and SIO interrupt request is generated.

SIO CONTROL REGISTERS (SIOCON)

The control register for serial I/O interface module, SIOCON, is located at E1H in page 0. It has the control setting for SIO module.

— Clock source selection (internal or external) for shift clock
— Interrupt enable
— Edge selection for shift operation
— Clear 3-bit counter and start shift operation
— Shift operation (transmit) enable
— Mode selection (transmit/receive or receive-only)
— Data direction selection (MSB first or LSB first)

A reset clears the SIOCON value to "00H". This configures the corresponding module with an internal clock source at the SCK, selects receive-only operating mode, and clears the 3-bit counter. The data shift operation and the interrupt are disabled. The selected data direction is MSB-first.

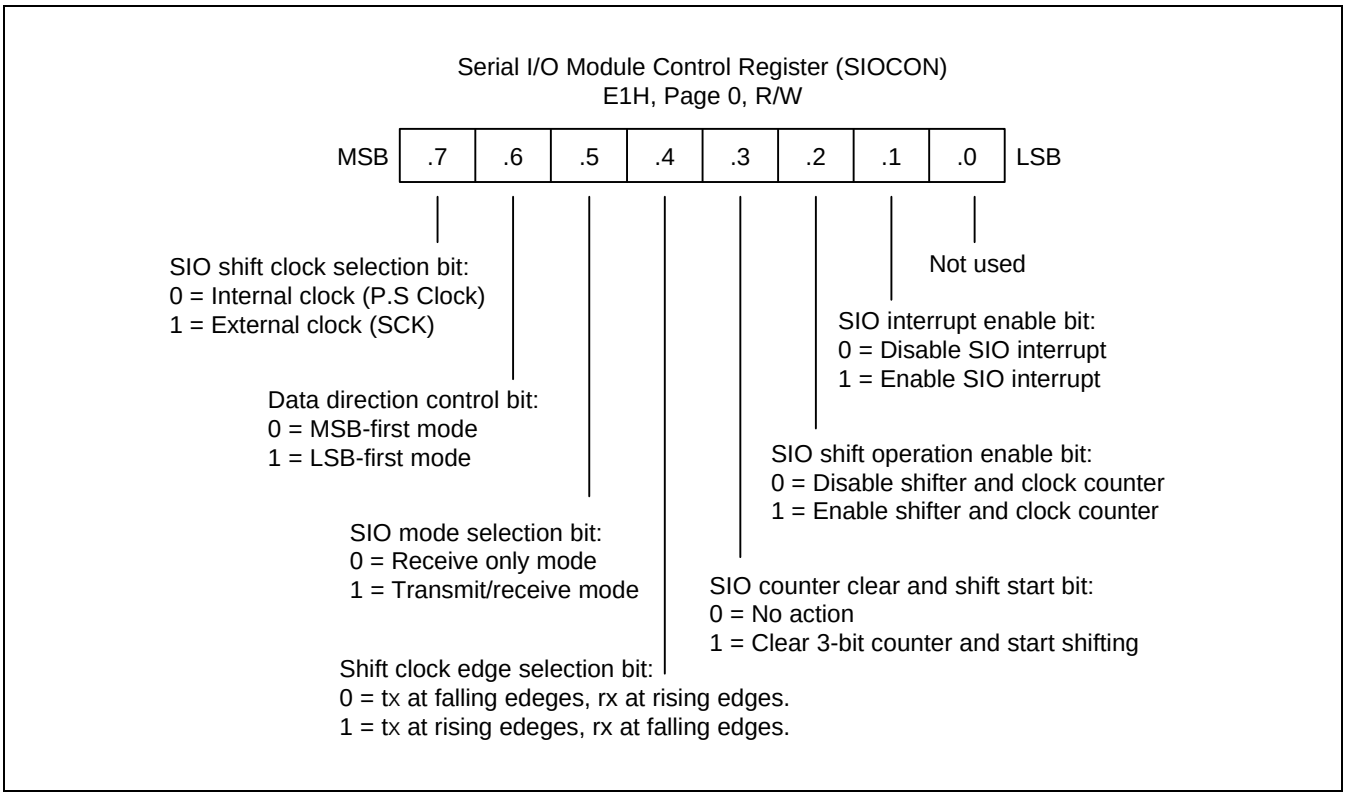


Figure 15-1. Serial I/O Module Control Register (SIOCON)

SIO PRE-SCALER REGISTER (SIOPS)

The prescaler register for serial I/O interface module, SIOPS, are located at E3H in page 0.
The value stored in the SIO pre-scale register, SIOPS, lets you determine the SIO clock rate (baud rate) as follows:

Baud rate = Input clock (fxx/4)/(Prescaler value + 1), or SCK input clock.

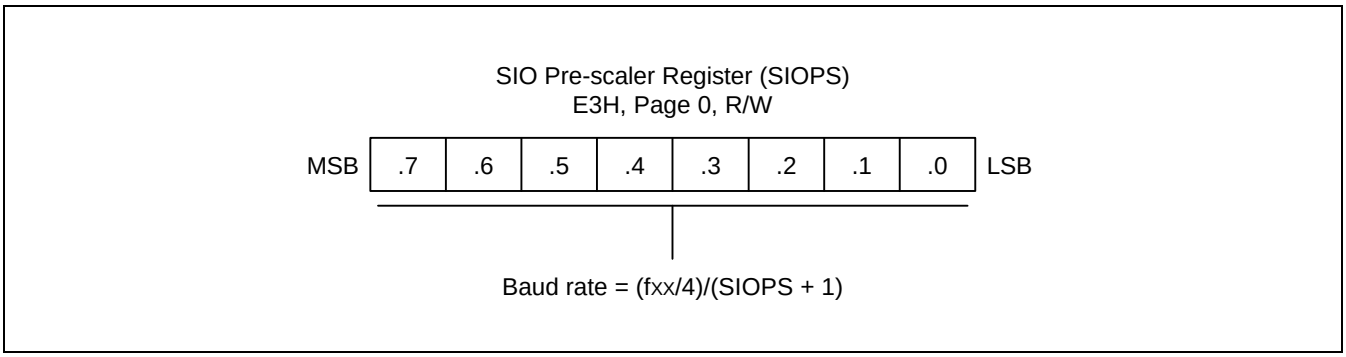


Figure 15-2. SIO Prescaler Register (SIOPS)

SIO BLOCK DIAGRAM

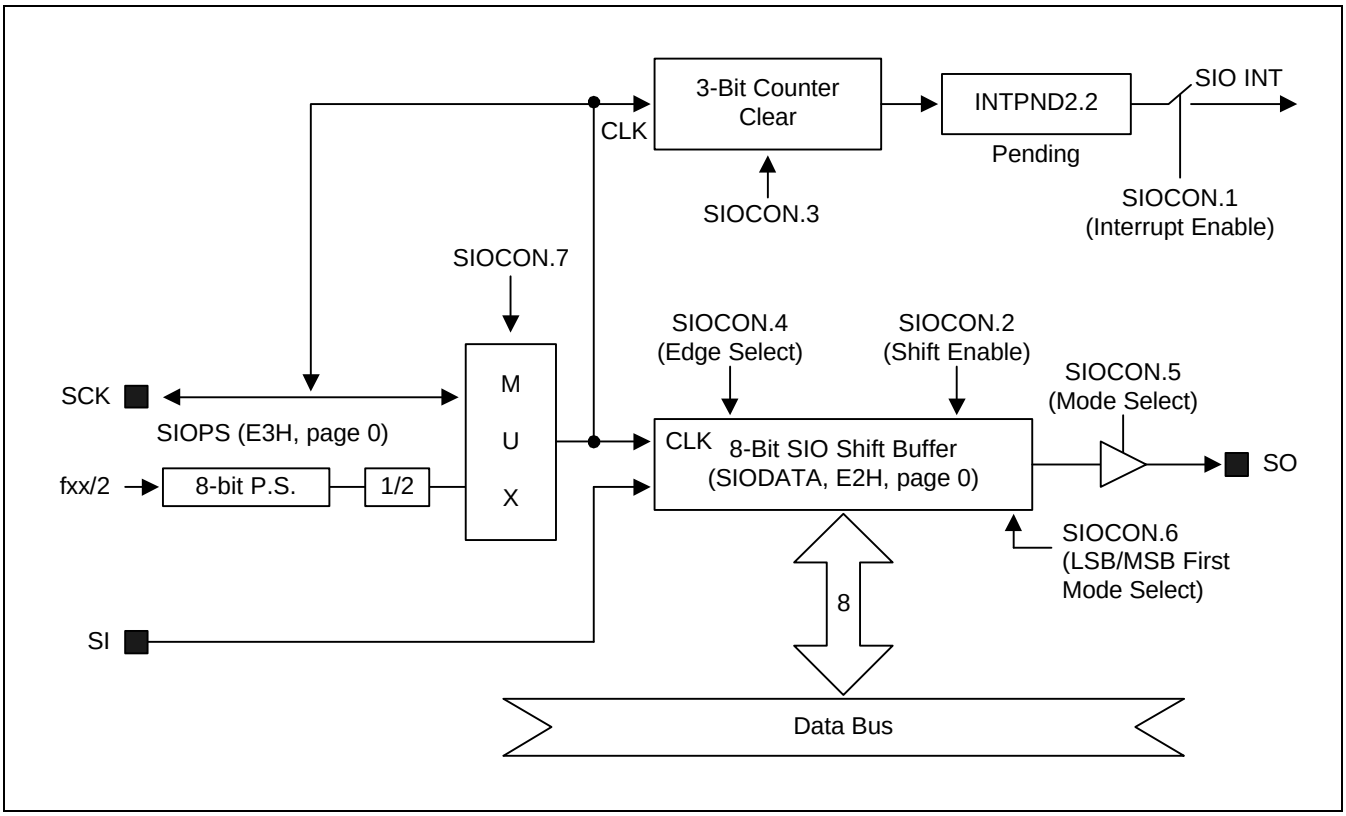


Figure 15-3. SIO Functional Block Diagram

SERIAL I/O TIMING DIAGRAM (SIO)

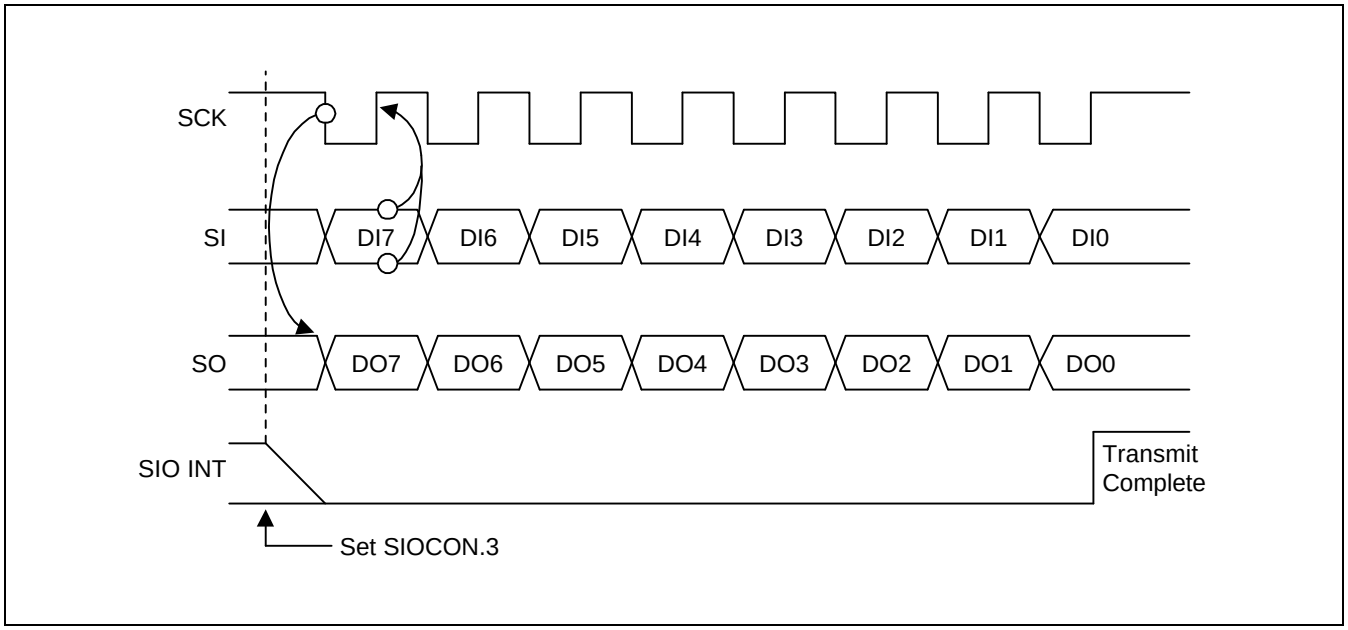


Figure 15-4. Serial I/O Timing in Transmit/Receive Mode (Tx at falling, SIOCON.4 = 0)

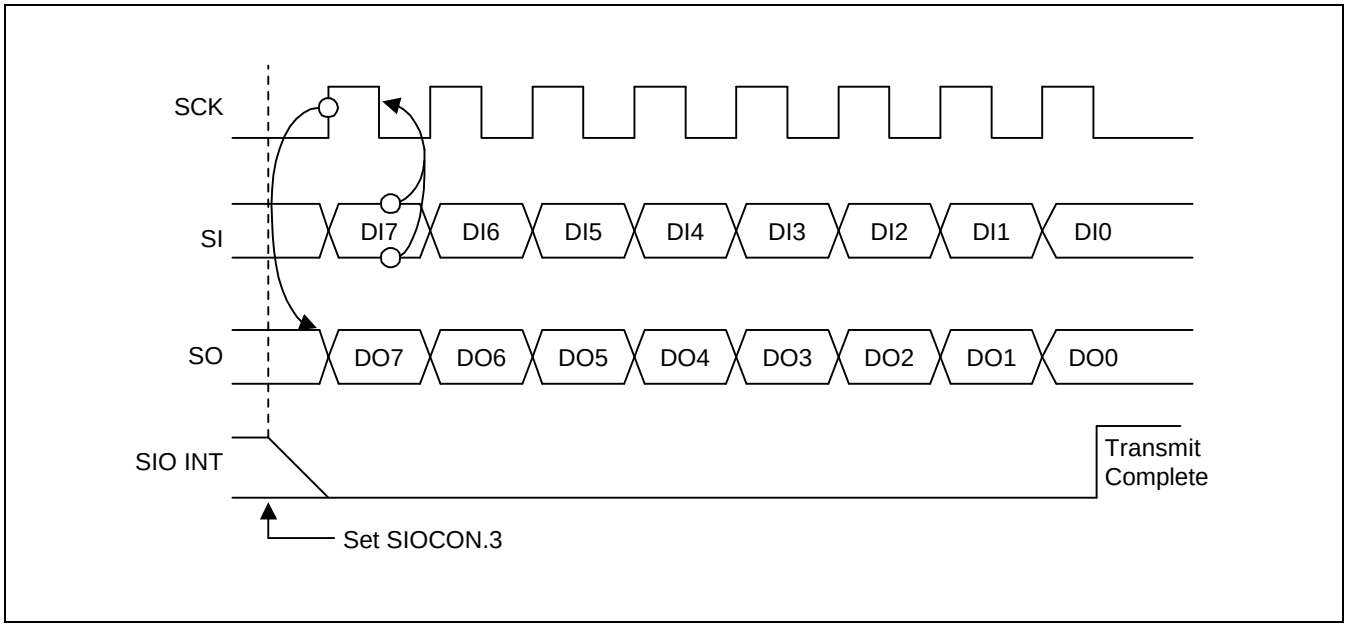


Figure 15-5. Serial I/O Timing in Transmit/Receive Mode (Tx at rising, SIOCON.4 = 1)

16 ELECTRICAL DATA

OVERVIEW

In this chapter, S3C9228/P9228 electrical characteristics are presented in tables and graphs. The information is arranged in the following order:

— Absolute maximum ratings
— D.C. electrical characteristics
— Data retention supply voltage in Stop mode
— Stop mode release timing when initiated by an external interrupt
— Stop mode release timing when initiated by a Reset
— I/O capacitance
— A.C. electrical characteristics
— A/D converter electrical characteristics
— Input timing for external interrupt
— Input timing for RESET
— Serial data transfer timing
— Oscillation characteristics
— Oscillation stabilization time
— Operating voltage range

Table 16-1. Absolute Maximum Ratings

(T_A = 25°C)

Parameter	Symbol	Conditions	Rating	Unit
Supply voltage	V_{DD}	–	– 0.3 to + 6.5	V
Input voltage	V_{IN}	Ports 0–6	– 0.3 to V_{DD} + 0.3	V
Output voltage	V_O	–	– 0.3 to V_{DD} + 0.3	V
Output current High	I_{OH}	One I/O pin active	– 15	mA
		All I/O pins active	– 60	
Output current Low	I_{OL}	One I/O pin active	+ 30	mA
		Total pin current for ports	+ 100	
Operating temperature	T_A	–	– 25 to + 85	°C
Storage temperature	T_{STG}	–	– 65 to + 150	°C

Table 16-2. D.C. Electrical Characteristics

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Operating Voltage	V_{DD}	fx = 0.4–4MHz, fxt = 32.8kHz	2.0	–	5.5	V
		fx = 0.4–8MHz	2.7	–	5.5	
Input High voltage	V_{IH1}	Ports 4–6	0.7 V_{DD}	–	V_{DD}	V
	V_{IH2}	Ports 0–3, RESET	0.8 V_{DD}		V_{DD}	
	V_{IH3}	X_{IN}, X_{OUT} and XT_{IN}, XT_{OUT}	V_{DD} – 0.1		V_{DD}	
Input Low voltage	V_{IL1}	Ports 4–6	–	–	0.3 V_{DD}	V
	V_{IL2}	Ports 0–3, RESET			0.2 V_{DD}	
	V_{IL3}	X_{IN}, X_{OUT}, XT_{IN}, XT_{OUT}			0.1	
Output High voltage	V_{OH}	V_{DD} = 4.5 to 5.5 V; All output ports; I_{OH} = –1 mA	V_{DD} – 1.0	–	V_{DD}	V
Output Low voltage	V_{OL}	V_{DD} = 4.5 to 5.5 V; All output ports; I_{OL} = 10 mA	–	–	2.0	V
Input High leakage current	I_{LIH1}	V_I = V_{DD}; All input pins except X_{IN}, X_{OUT}, XT_{IN}, XT_{OUT}	–	–	3	µA
	I_{LIH2}	V_I = V_{DD}; X_{IN}, X_{OUT}, XT_{IN}, XT_{OUT}			20	

Table 16-2. D.C. Electrical Characteristics (Continued)

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input Low leakage current	I_{LIL1}	V_I = 0 V; All input pins except RESET, X_{OUT}, XT_{IN}, XT_{OUT}	–	–	–3	µA
	I_{LIL2}	V_I = 0 V; X_{IN}, X_{OUT}, XT_{IN}, XT_{OUT}			–20	
Output High leakage current	I_{LOH}	$V_O = V_{DD}$ All output pins	–	–	3	
Output Low leakage current	I_{LOL}	V_O = 0 V All output pins	–	–	–3	
Pull-Up Resistor	R_{L1}	V_I = 0 V; V_{DD} = 5V, T_A = 25°C Ports 0–6	25	47	100	kΩ
		V_{DD} = 3V, T_A = 25°C	50	90	150	
	R_{L2}	V_I = 0 V; V_{DD} = 5V, T_A = 25°C RESET	150	250	400	
		V_{DD} = 3V, T_A = 25°C	300	500	700	
Oscillator Feed back Resistors	R_{OSC1}	V_{DD} = 5 V, T_A = 25 °C $X_{IN} = V_{DD}$, X_{OUT} = 0V	300	600	1500	kΩ
	R_{OSC2}	V_{DD} = 5 V, T_A = 25 °C $XT_{IN} = V_{DD}$, XT_{OUT} = 0 V	1500	3000	4500	
LCD Voltage Dividing Resistor	R_{LCD}	T_A = 25 °C	50	70	90	kΩ
$\lvert V_{LCD}\text{-COMi} \rvert$ Voltage Drop (i = 0-7)	V_{DC}	V_{DD} = 2.7 V to 5.5 V - 15 µA per common pin	–	–	120	mV
$\lvert V_{LCD}\text{-SEGx} \rvert$ Voltage Drop (x = 0–19)	V_{DS}	V_{DD} = 2.7 V to 5.5 V - 15 µA per common pin	–	–	120	
Middle Output Voltage	V_{LC2}	V_{DD} = 2.7 V to 5.5 V, LCD clock = 0Hz, $V_{LC1} = V_{DD}$	$0.8V_{DD}$–0.2	$0.8V_{DD}$	$0.8V_{DD}$+ 0.2	V
	V_{LC3}		$0.6V_{DD}$–0.2	$0.6V_{DD}$	$0.6V_{DD}$+ 0.2	
	V_{LC4}		$0.4V_{DD}$–0.2	$0.4V_{DD}$	$0.4V_{DD}$+ 0.2	
	V_{LC5}		$0.2V_{DD}$–0.2	$0.2V_{DD}$	$0.2V_{DD}$+ 0.2	

NOTE: Low leakage current is absolute value.

Table 16-2. D.C. Electrical Characteristics (Concluded)

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Supply current (1)	I_{DD1}	Run mode: V_{DD} = 5 V ± 10%	8.0 MHz	–	6.0	12.0	mA
		Crystal oscillator C1 = C2 = 22pF	4.19 MHz		3.0	6.0	
		V_{DD} = 3 V ± 10%	8.0 MHz		2.5	5.0	
			4.19 MHz		1.5	3.0	
	I_{DD2}	Idle mode: V_{DD} = 5 V ± 10%	8.0 MHz		1.3	3.0	
		Crystal oscillator C1 = C2 = 22pF	4.19 MHz		1.0	2.0	
		V_{DD} = 3 V ± 10%	8.0 MHz		0.8	1.6	
			4.19 MHz		0.4	0.8	
	I_{DD3}	Run mode: V_{DD} = 3 V ± 10%, 32 kHz crystal oscillator			15	30	µA
	I_{DD4}	Idle mode: V_{DD} = 3 V ± 10%, 32 kHz crystal oscillator			6	15	
	I_{DD5}	Stop mode; V_{DD} = 5 V ± 10%, T_A = 25 °C			0.5	3	
		Stop mode; V_{DD} = 3 V ± 10%, T_A = 25 °C			0.3	2	

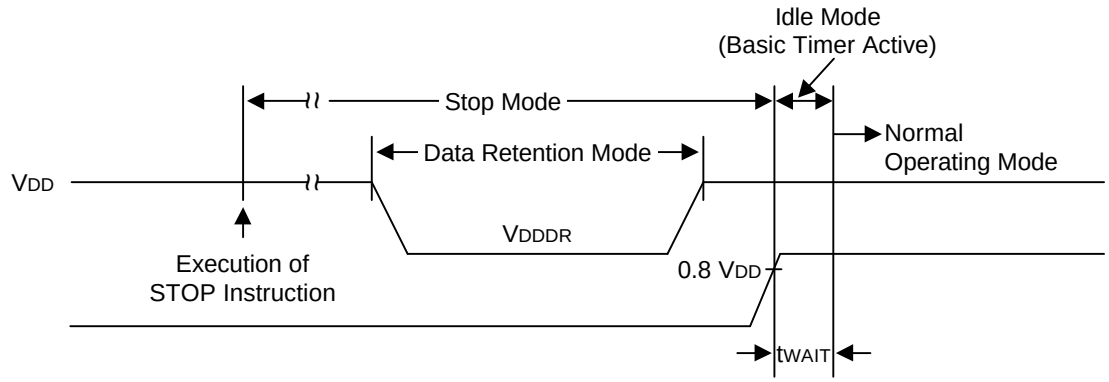
NOTES:

1. Supply current does not include current drawn through internal pull-up resistors, LCD voltage dividing resistors, and ADC.
2. I_{DD1} and I_{DD2} include power consumption for subsystem clock oscillation.
3. I_{DD3} and I_{DD4} are current when main system clock oscillation stops and the subsystem clock is used.
4. I_{DD5} is current when main system clock and subsystem clock oscillation stops.

Table 16-3. Data Retention Supply Voltage in Stop Mode

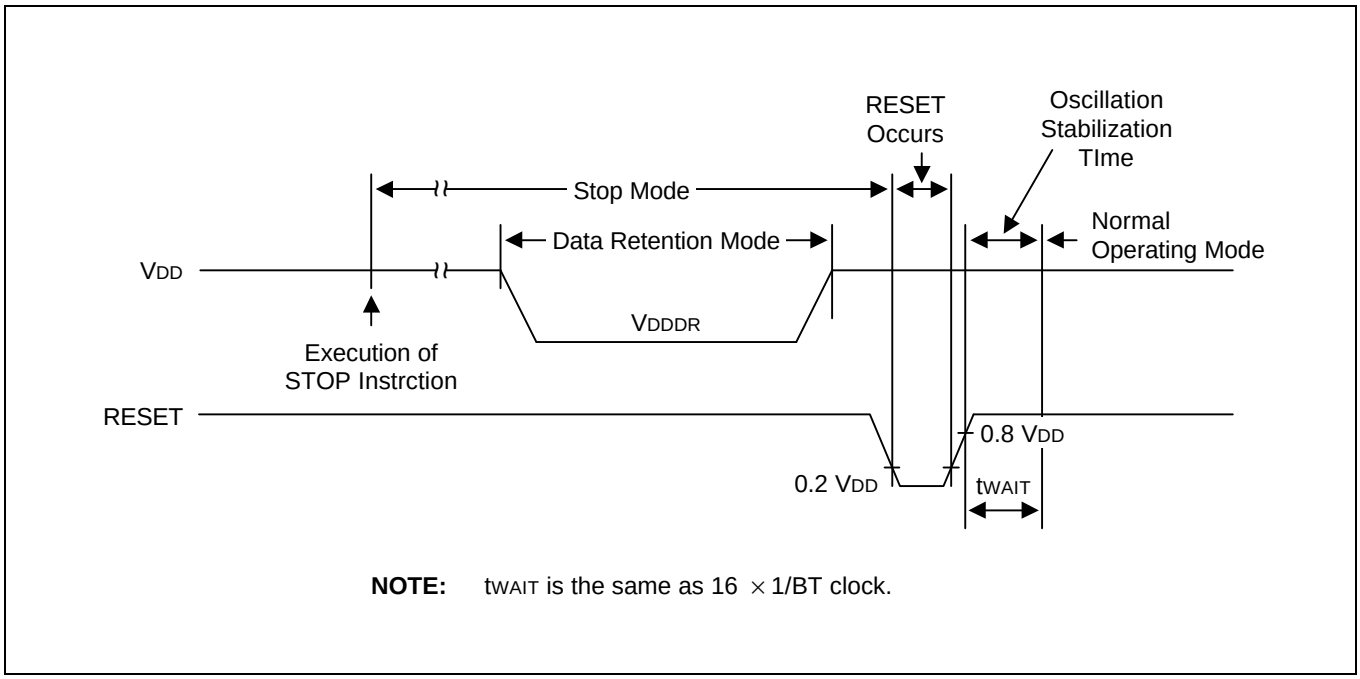
(T_A = – 25 °C to + 85 °C)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V_{DDDR}	–	2.0	–	5.5	V
Data retention supply current	I_{DDDR}	Stop mode, T_A = 25 °C V_{DDDR} = 2.0 V	–	–	1	µA



NOTE: tWAIT is the same as 16 x 1/BT clock.

Figure 16-1. Stop Mode Release Timing When Initiated by an External Interrupt



NOTE: tWAIT is the same as 16 × 1/BT clock.

Figure 16-2. Stop Mode Release Timing When Initiated by a RESET

Table 16-4. Input/Output Capacitance

(T_A = 25 °C, V_{DD} = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input capacitance	C_{IN}	f = 1 MHz; unmeasured pins are connected to V_{SS}	–	–	10	pF
Output capacitance	C_{OUT}					
I/O capacitance	C_{IO}					

Table 16-5. A.C. Electrical Characteristics

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
SCK cycle time	t_{KCY}	External SCK source	1,000	–	–	ns
		Internal SCK source	1,000			
SCK high, low width	t_{KH}, t_{KL}	External SCK source	500			
		Internal SCK source	t_{KCY}/2–50			
SI setup time to SCK high	t_{SIK}	External SCK source	250			
		Internal SCK source	250			
SI hold time to SCK high	t_{KSI}	External SCK source	400			
		Internal SCK source	400			
Output delay for SCK to SO	t_{KSO}	External SCK source	–	–	300	ns
		Internal SCK source			250	
Interrupt input, High, Low width	t_{INTH}, t_{INTL}	All interrupt V_{DD} = 3 V	500	700	–	ns
RESET input Low width	t_{RSL}	Input V_{DD} = 3 V	10	–	–	µs

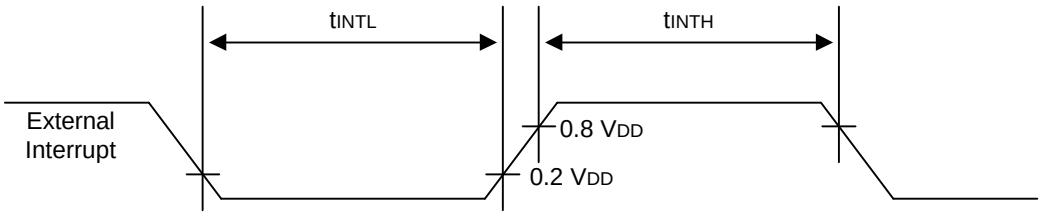
Table 16-6. A/D Converter Electrical Characteristics

(T_A = – 25°C to + 85°C, V_{DD} = 2.7 V to 5.5 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Resolution			–	10	–	bit
Total accuracy		VDD = 5.12 V	–	–	±3	LSB
Integral linearity error	ILE	fxx = 8 MHz	–	–	±2	
Differential linearity error	DLE	f_{CON} = fxx/4	–	–	±1	
Offset error of top	EOT		–	±1	±3	
Offset error of bottom	EOB		–	±0.5	±2	
Conversion time (1)	T_{CON}	10-bit resolution 50 × fxx/4, fxx = 8MHz	25	–	–	µS
Analog input voltage	V_{IAN}	–	V_{SS}	–	V_{DD}	V
Analog input impedance	R_{AN}	–	2	1000	–	MΩ
Analog input current	I_{ADIN}	V_{DD} = 5 V	–	–	10	µA
	I_{ADC}	V_{DD} = 5 V	–	1	3	mA
		V_{DD} = 3 V		0.5	1.5	
		V_{DD} = 5 V When 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 an operating current during A/D conversion.



NOTE: The unit tCPU means one CPU clock period.

Figure 16-3. Input Timing for External Interrupts

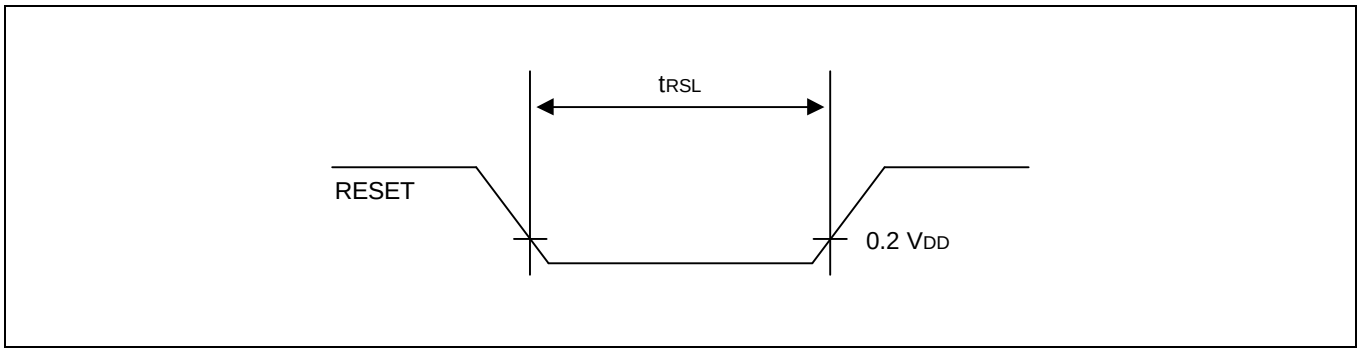


Figure 16-4. Input Timing for RESET

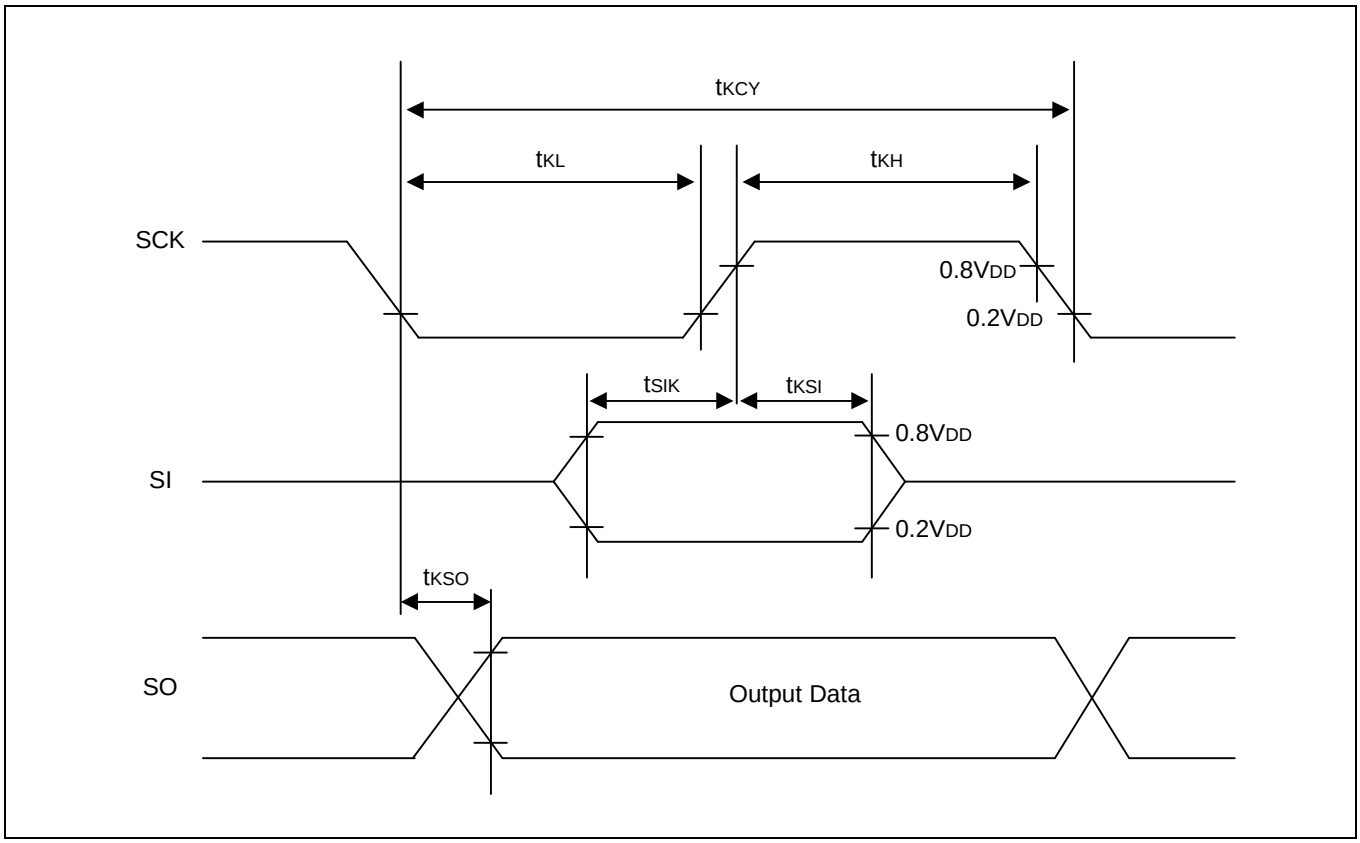


Figure 16-5. Serial Data Transfer Timing

Table 16-7. Main Oscillation Characteristics

(T_A = – 25°C to + 85°C)

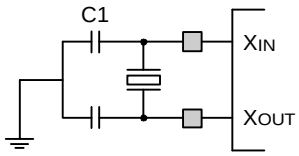
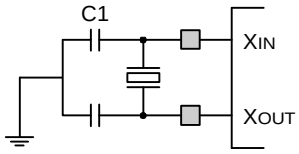
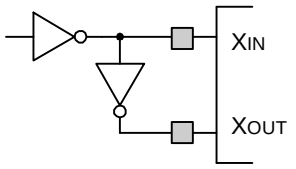
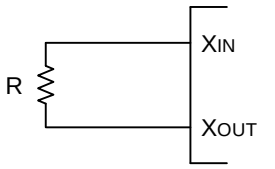
Oscillator	Clock Configuration	Parameter	Test Condition	Min	Typ	Max	Units
Crystal		Main oscillation frequency	2.7 V – 5.5 V	0.4	–	8	MHz
			2.0 V – 5.5 V	0.4	–	4	
Ceramic Oscillator		Main oscillation frequency	2.7 V – 5.5 V	0.4	–	8	
			2.0 V – 5.5 V	0.4	–	4	
External Clock		X_{IN} input frequency	2.7 V – 5.5 V	0.4	–	8	
			2.0 V – 5.5 V	0.4	–	4	
RC Oscillator		Frequency	5.0 V	0.4	–	2	MHz
		Frequency	3.0 V	0.4	–	1	

Table 16-8. Sub Oscillation Characteristics

(T_A = – 25°C to + 85°C)

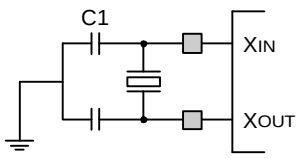
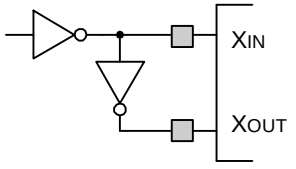
Oscillator	Clock Configuration	Parameter	Test Condition	Min	Typ	Max	Units
Crystal		Sub oscillation frequency	2.0 V – 5.5 V	32	32.768	35	kHz
External clock		XT_{IN} input frequency	2.0 V – 5.5 V	32	–	100	

Table 16-9. Main Oscillation Stabilization Time

(T_A = – 25 °C to + 85 °C, V_{DD} = 2.0 V to 5.5 V)

Oscillator	Test Condition	Min	Typ	Max	Unit
Crystal	fx > 1 MHz	–	–	30	ms
Ceramic	Oscillation stabilization occurs when VDD is equal to the minimum oscillator voltage ranage.	–	–	10	ms
External clock	X_{IN} input high and low width (t_{XH}, t_{XL})	62.5	–	1250	ns

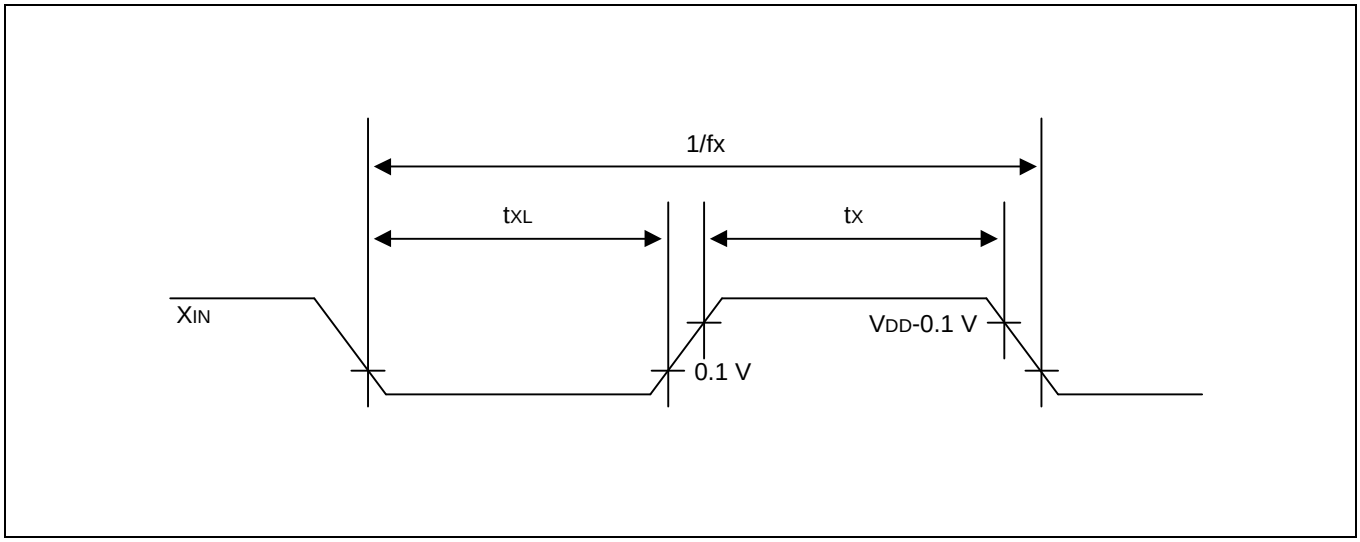


Figure 16-6. Clock Timing Measurement at X_{IN}

Table 16-10. Sub Oscillation Stabilization Time

(T_A = – 25 °C to + 85 °C, V_{DD} = 2.0 V to 5.5 V)

Oscillator	Test Condition	Min	Typ	Max	Unit
Crystal	–	–	–	10	s
External clock	XT_{IN} input high and low width (t_{XH}, t_{XL})	5	–	15	µs

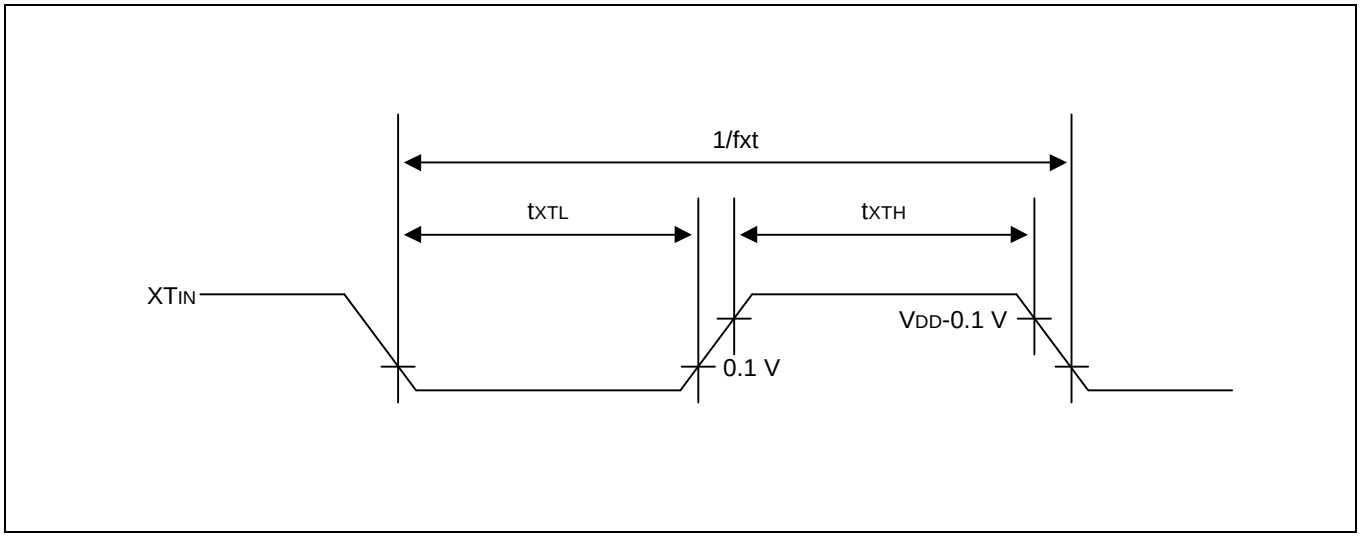
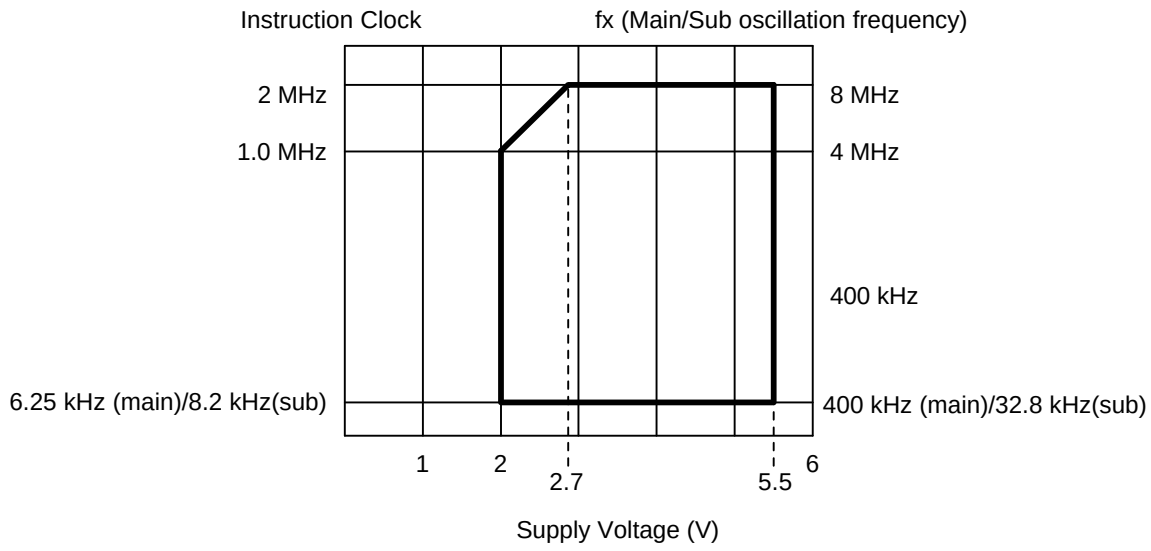


Figure 16-7. Clock Timing Measurement at XT_{IN}



Instruction Clock = 1/4n x oscillator frequency (n = 1, 2, 8, 16)

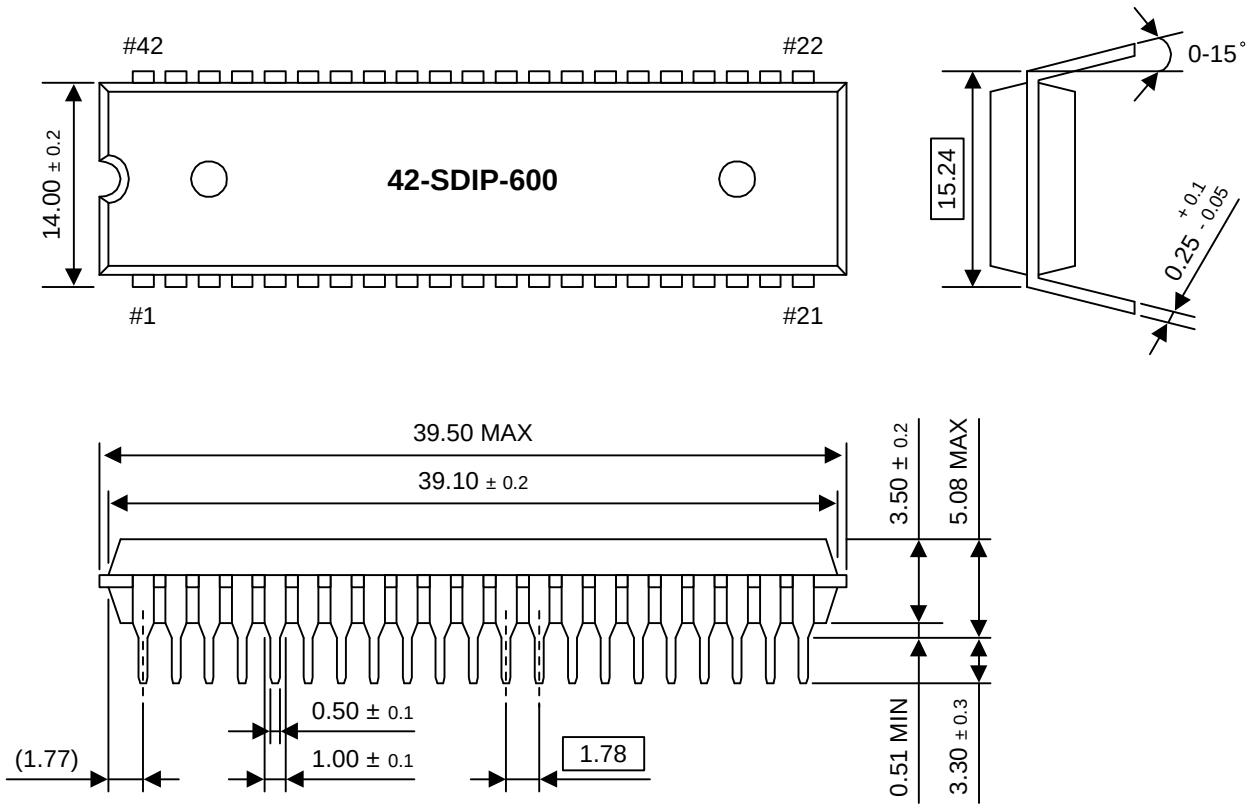
Figure 16-8. Operating Voltage Range

NOTES

17 MECHANICAL DATA

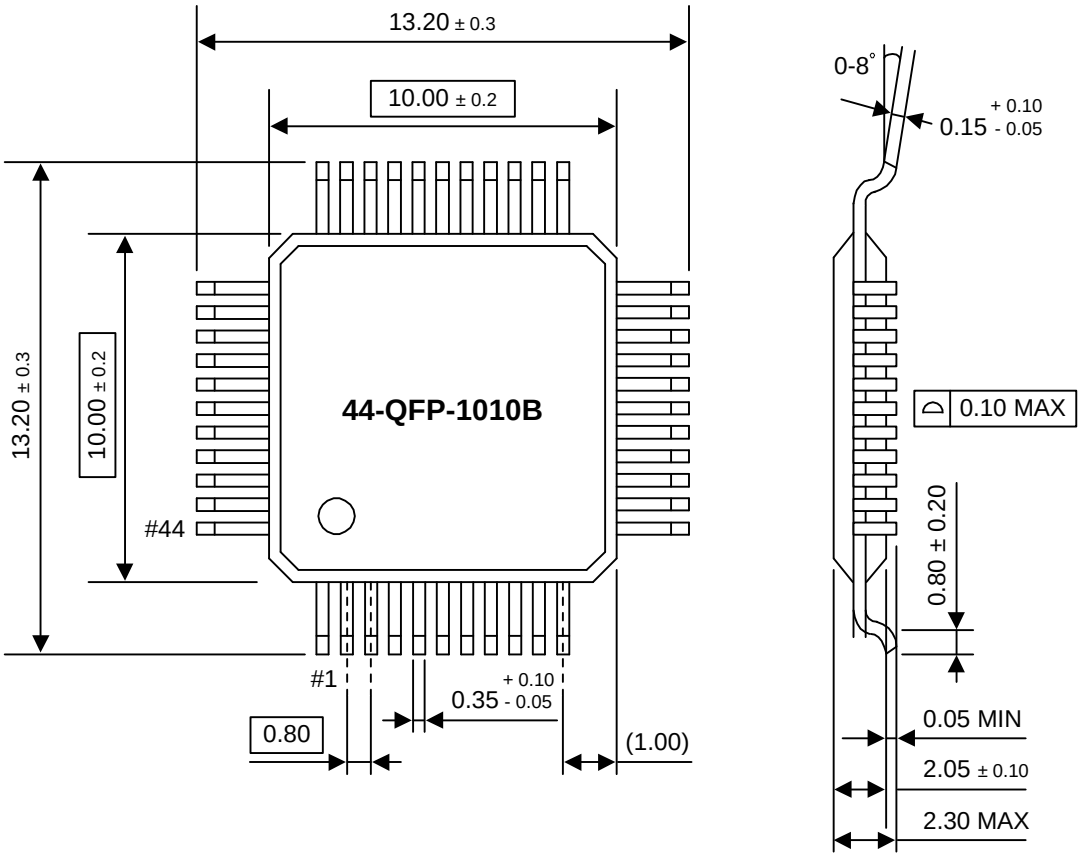
OVERVIEW

The S3C9228/P9228 microcontroller is currently available in a 42-pin SDIP and 44-pin QFP package.



NOTE: Dimensions are in millimeters.

Figure 17-1. 42-SDIP-600 Package Dimensions



NOTE: Dimensions are in millimeters.

Figure 17-2. 44-QFP-1010B Package Dimensions

18 S3P9228 OTP

OVERVIEW

The S3P9228 single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the S3C9228 microcontroller. It has an on-chip OTP ROM instead of masked ROM. The EPROM is accessed by serial data format.

The S3P9228 is fully compatible with the S3C9228, both in function and in pin configuration. Because of its simple programming requirements, the S3P9228 is ideal for use as an evaluation chip for the S3C9228.

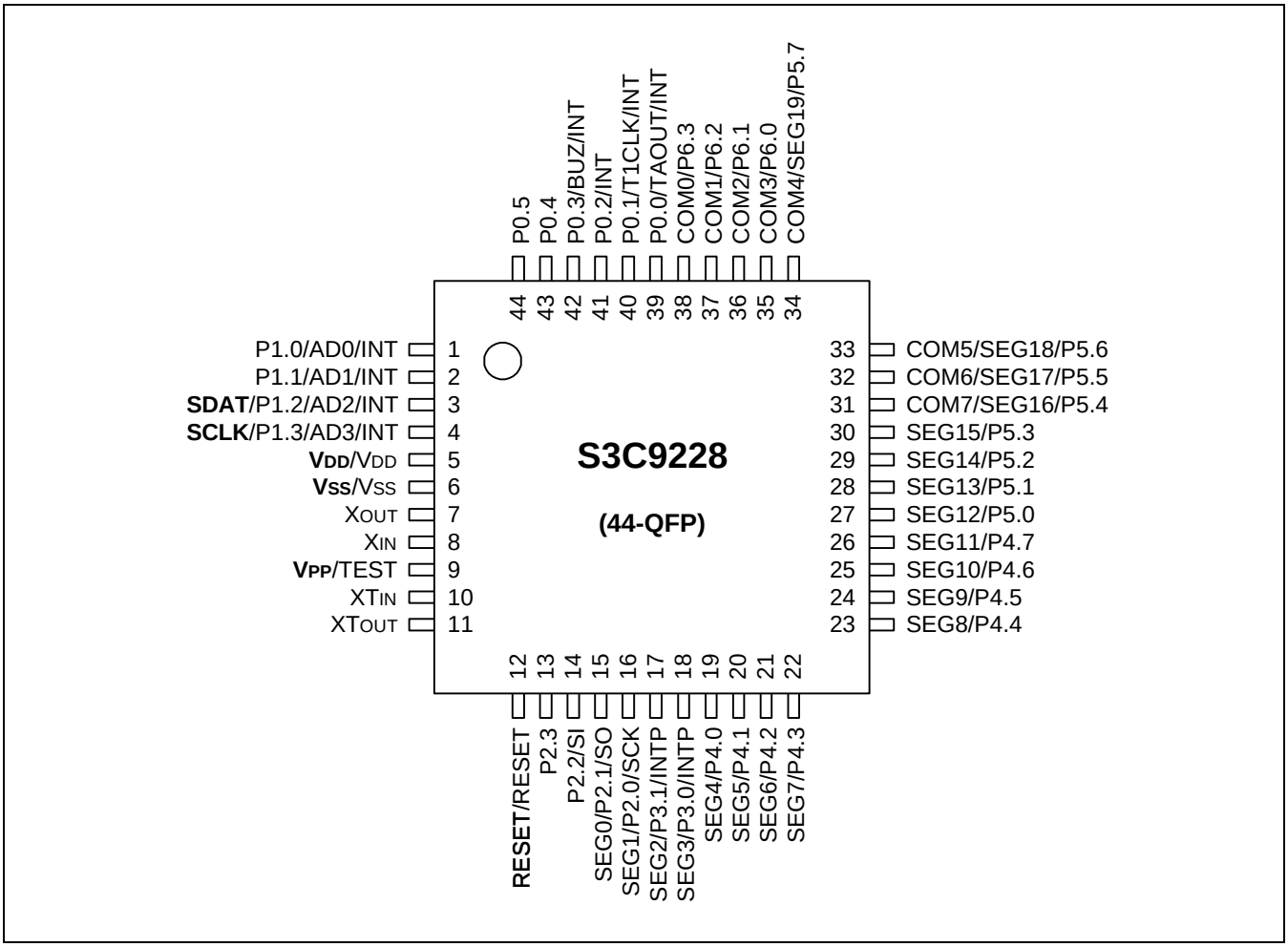


Figure 18-1. S3P9228 44-QFP Pin Assignments

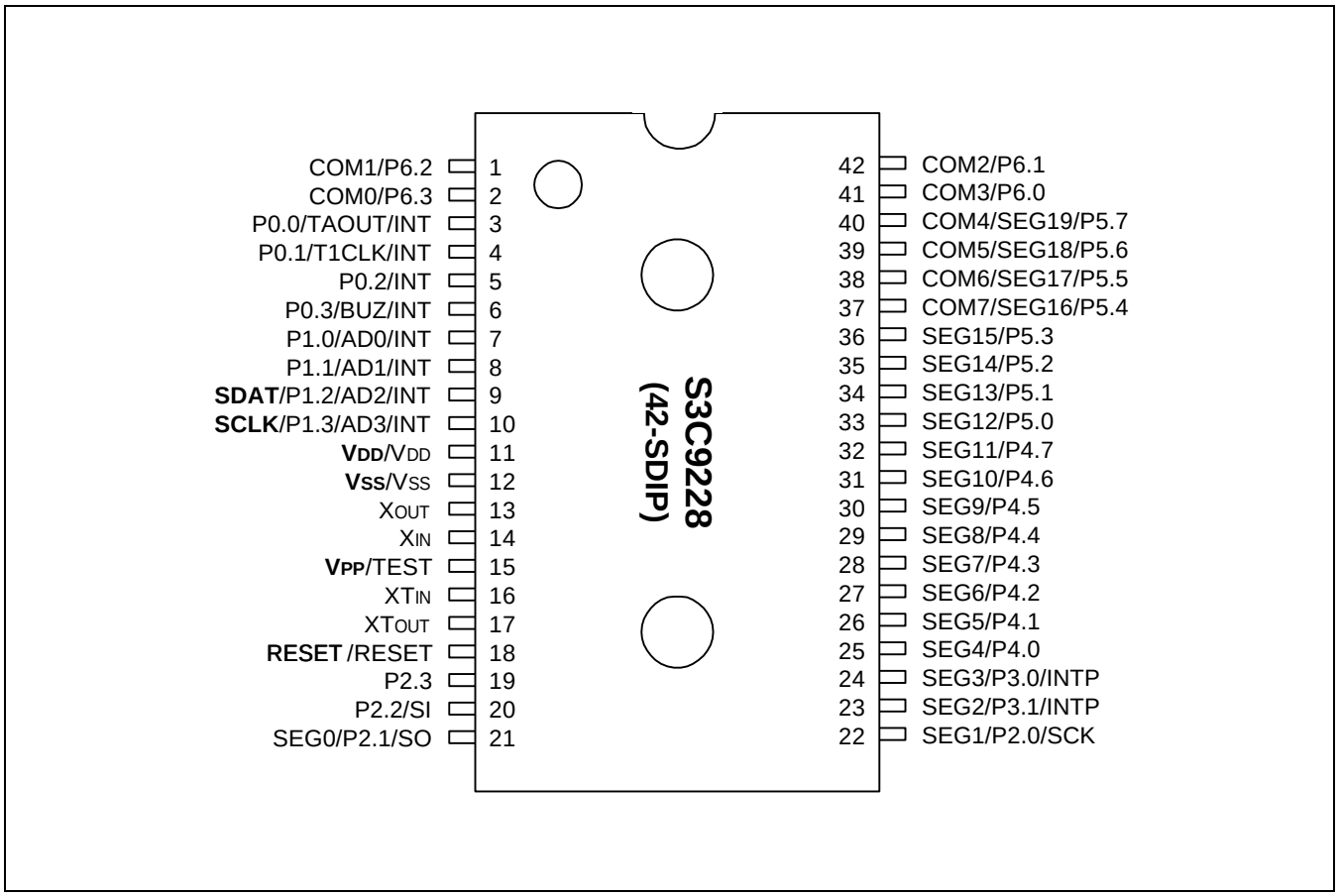


Figure 18-2. S3P9228 42-SDIP Pin Assignments

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

Main Chip	During Programming			
Pin Name	Pin Name	Pin No.	I/O	Function
P1.2	SDAT	3 (9)	I/O	Serial data pin. Output port when reading and input port when writing. Can be assigned as a Input/push-pull output port.
P1.3	SCLK	4 (10)	I/O	Serial clock pin. Input only pin.
TEST	V_{PP}(TEST)	9 (15)	I	Power supply pin for EPROM cell writing (indicates that OTP enters into the writing mode). When 12.5 V is applied, OTP is in writing mode and when 5 V is applied, OTP is in reading mode. (Option)
RESET	RESET	12 (18)	I	Chip initialization
V_{DD}/V_{SS}	V_{DD}/V_{SS}	5/6 (11/12)	I	Logic power supply pin. V_{DD} should be tied to + 5 V during programming.

NOTE: Parentheses indicate pin number for 42-SDIP package.

Table 18-2. Comparison of S3P9228 and S3C9228 Features

Characteristic	S3P9228	S3C9228
Program Memory	8 Kbyte EPROM	8 Kbyte mask ROM
Operating Voltage (V_{DD})	2.0 V to 5.5 V	2.0 V to 5.5 V
OTP Programming Mode	V_{DD} = 5 V, V_{PP}(TEST)=12.5V	
Pin Configuration	44-QFP, 42-SDIP	44-QFP, 42-SDIP
EPROM Programmability	User Program 1 time	Programmed at the factory

OPERATING MODE CHARACTERISTICS

When 12.5 V is supplied to the V_{PP}(TEST) pin of the S3P72C8, the EPROM programming mode is entered. The operating mode (read, write, or read protection) is selected according to the input signals to the pins listed in Table 17-3 below.

Table 18-3. Operating Mode Selection Criteria

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

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

Table 18-4. D.C. Electrical Characteristics

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Supply current (1)	I_{DD1}	Run mode: V_{DD} = 5 V ± 10%	8.0 MHz	–	6.0	12.0	mA
		Crystal oscillator C1 = C2 = 22pF	4.19 MHz		3.0	6.0	
		V_{DD} = 3 V ± 10%	8.0 MHz		2.5	5.0	
			4.19 MHz		1.5	3.0	
	I_{DD2}	Idle mode: V_{DD} = 5 V ± 10%	8.0 MHz		1.3	3.0	
		Crystal oscillator C1 = C2 = 22pF	4.19 MHz		1.0	2.0	
		V_{DD} = 3 V ± 10%	8.0 MHz		0.8	1.6	
			4.19 MHz		0.4	0.8	
	I_{DD3}	Run mode: V_{DD} = 3 V ± 10%, 32 kHz crystal oscillator			15	30	µA
	I_{DD4}	Idle mode: V_{DD} = 3 V ± 10%, 32 kHz crystal oscillator			6	15	
	I_{DD5}	Stop mode; V_{DD} = 5 V ± 10%, T_A = 25 °C			0.5	3	
		Stop mode; V_{DD} = 3 V ± 10%, T_A = 25 °C			0.3	2	

NOTES:

1. Supply current does not include current drawn through internal pull-up resistors, LCD voltage dividing resistors, and ADC.
2. I_{DD1} and I_{DD2} include power consumption for subsystem clock oscillation.
3. I_{DD3} and I_{DD4} are current when main system clock oscillation stops and the subsystem clock is used.
4. I_{DD5} is current when main system clock and subsystem clock oscillation stops.

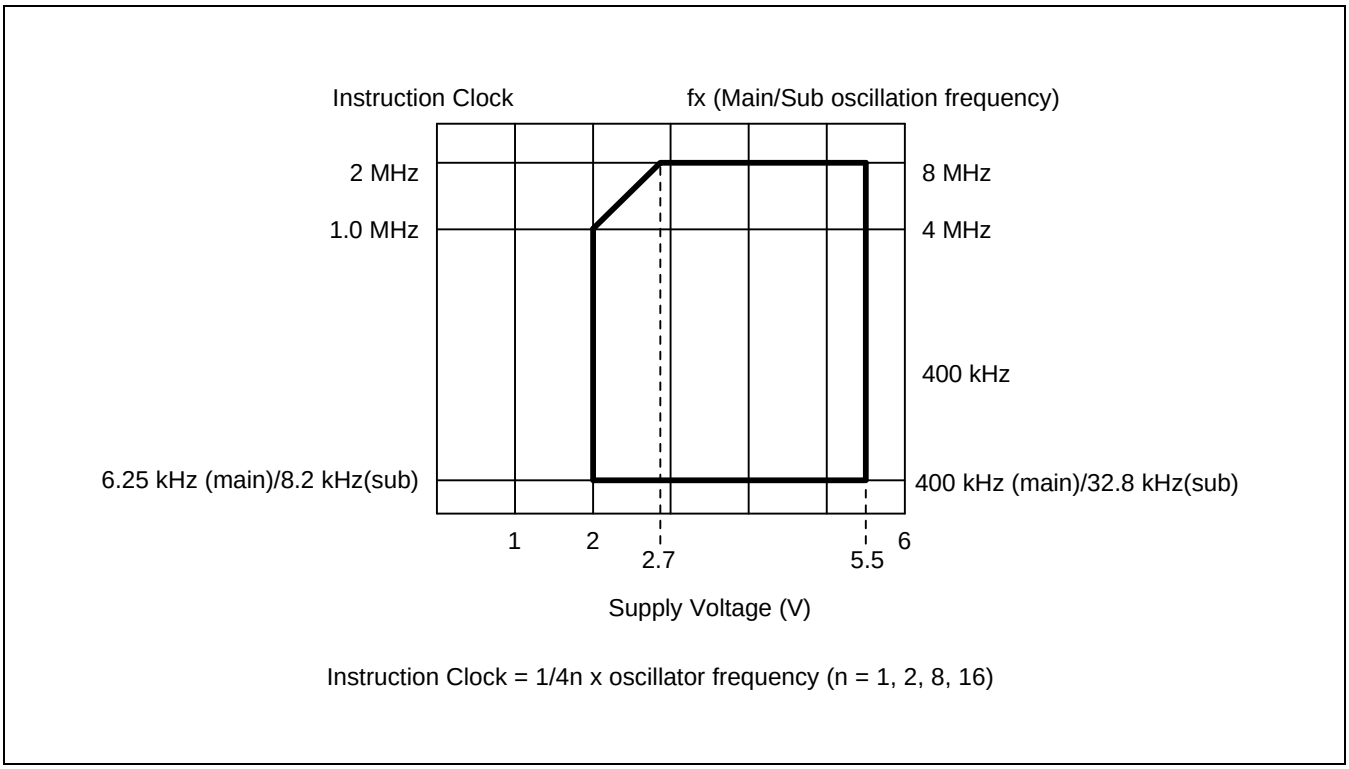


Figure 18-3. Standard Operating Voltage Range

NOTES

19 DEVELOPMENT TOOLS

OVERVIEW

Samsung provides a powerful and easy-to-use development support system in turn key 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, S3C8, S3C9 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.

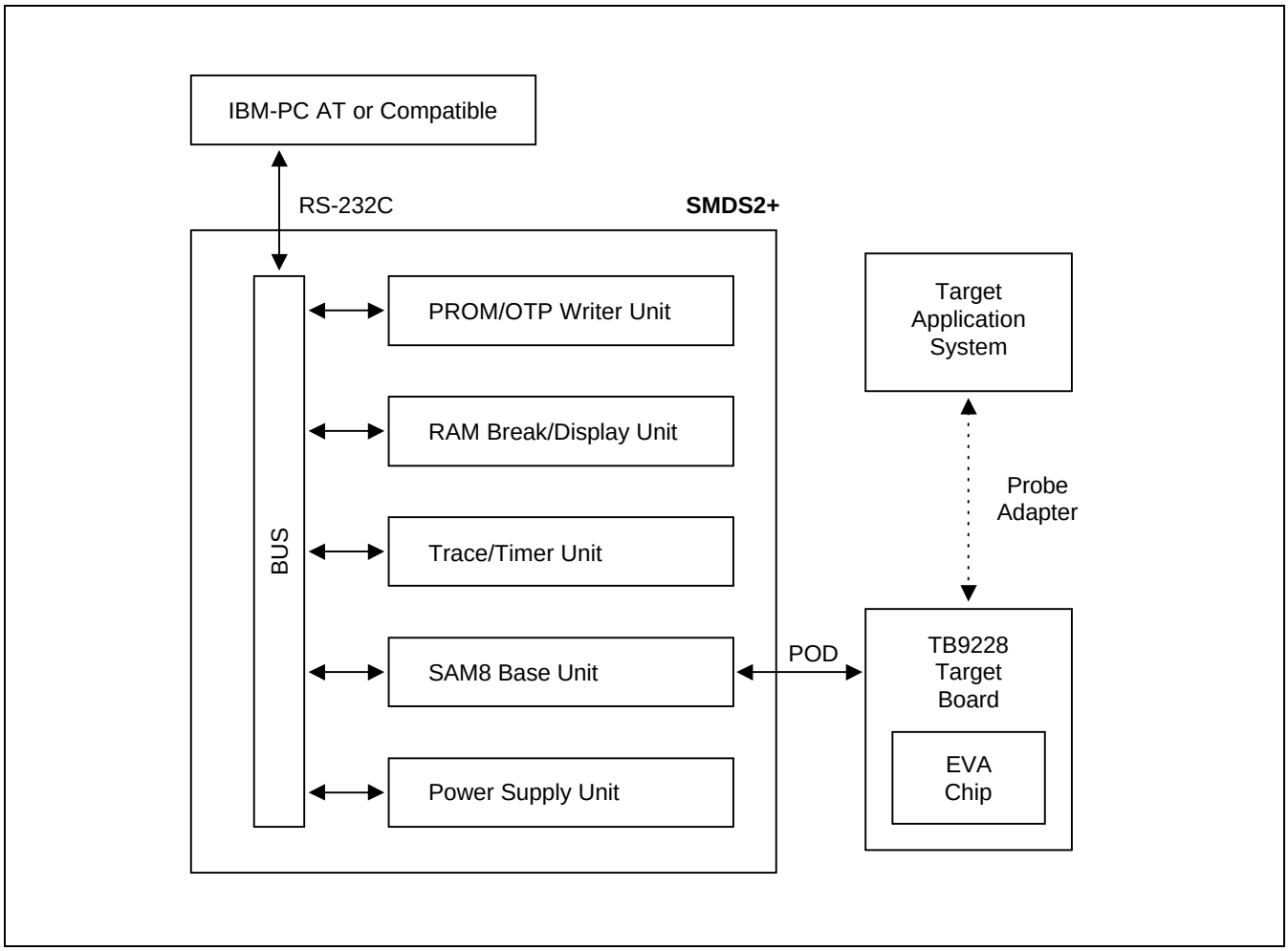


Figure 19-1. SMDS Product Configuration (SMDS2+)

TB9228 TARGET BOARD

The TB9228 target board is used for the S3C9228 microcontroller. It is supported by the SMDS2+ development system.

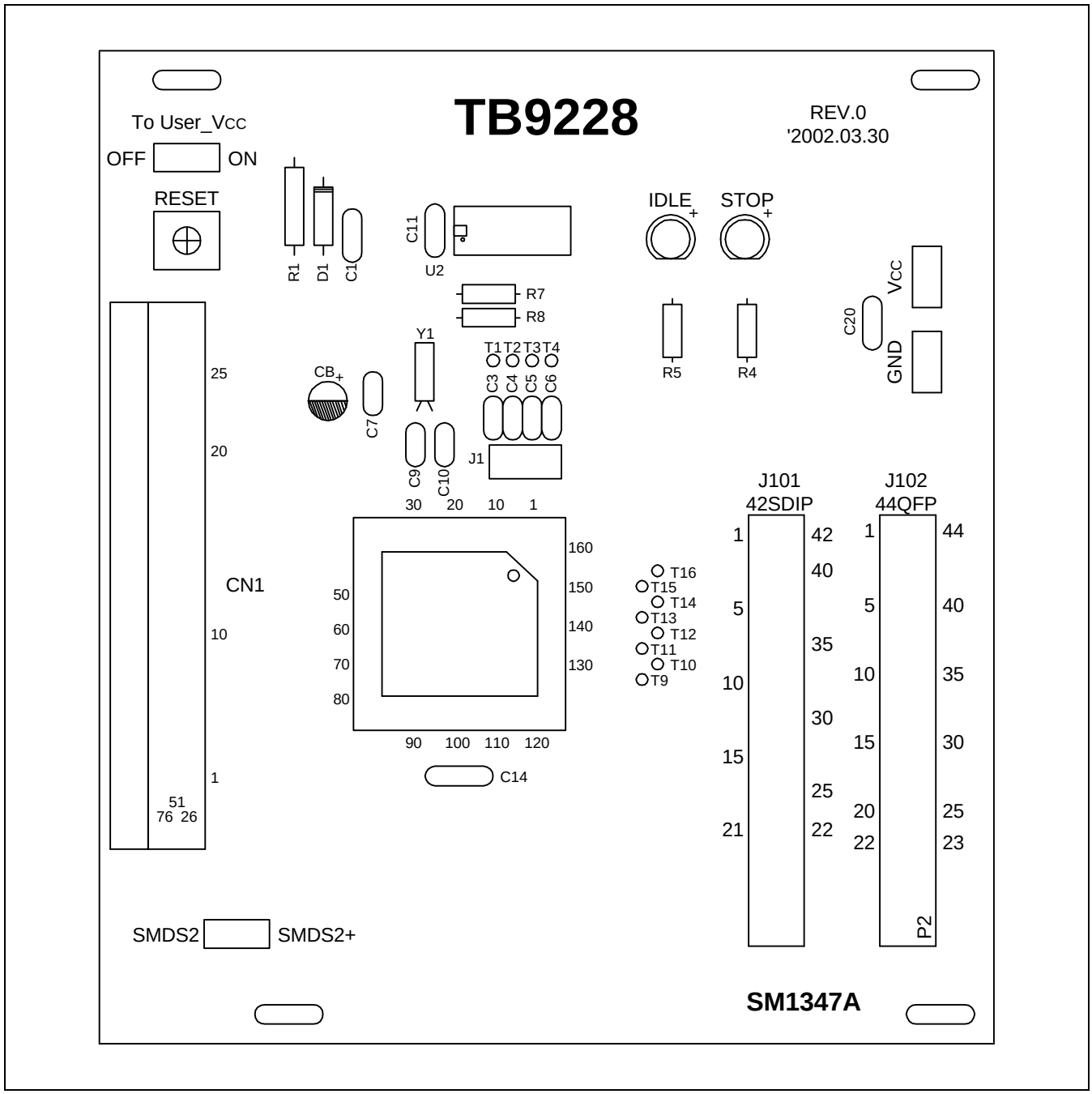
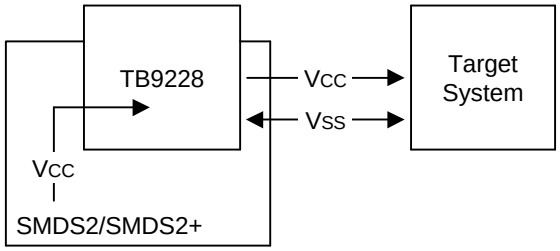
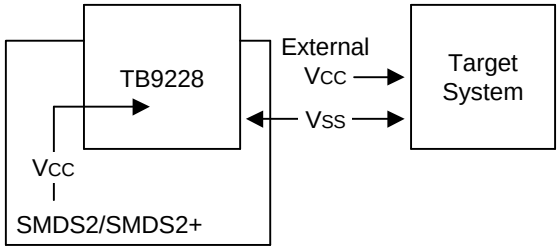


Figure 19-2. TB9228 Target Board Configuration

Table 19-1. Power Selection Settings for TB9228

"To User_V_{CC}" Settings	Operating Mode	Comments
To User_Vcc Off On		The SMDS2/SMDS2+ supplies V_{CC} to the target board (evaluation chip) and the target system.
To User_Vcc Off On		The SMDS2/SMDS2+ supplies V_{CC} only to the target board (evaluation chip). The target system must have its own power supply.

NOTE: The following symbol in the "To User_V_{CC}" Setting column indicates the electrical short (off) configuration:

SMDS2+ Selection (SAM8)

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

Table 19-2. The SMDS2+ Tool Selection Setting

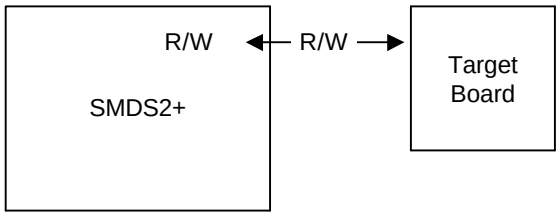
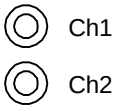
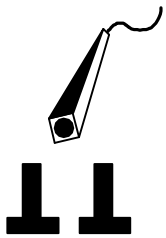
"SW1" Setting	Operating Mode
SMDS2 SMDS2+	

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

Target Board Part	Comments
External Triggers 	 Connector from External Trigger Sources of the Application System 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.

IDLE LED

The Green LED is ON when the evaluation chip (S3E9220) is in idle mode.

STOP LED

The Red LED is ON when the evaluation chip (S3E9220) is in stop mode.

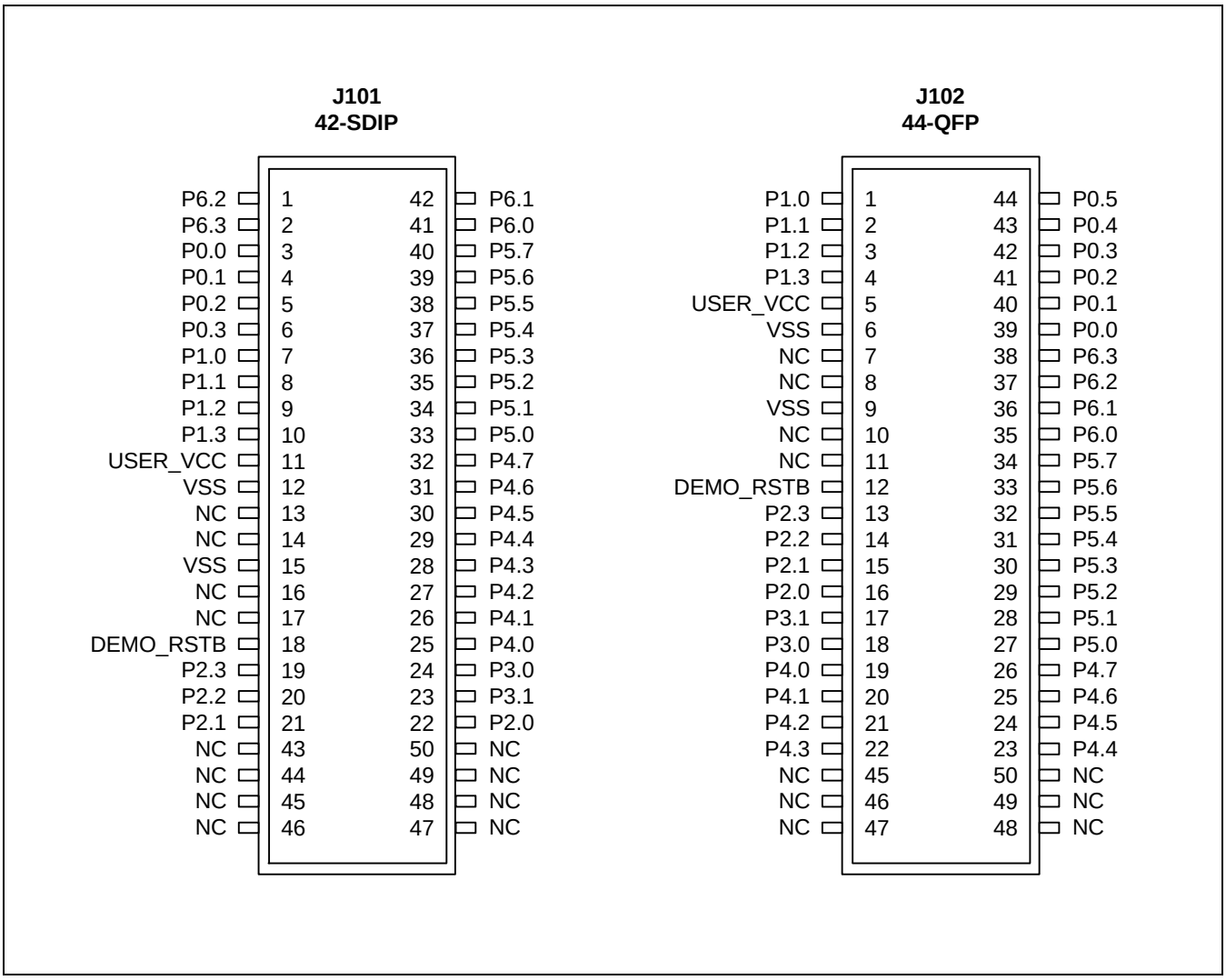


Figure 19-3. Connectors (J101, J102) for TB9228

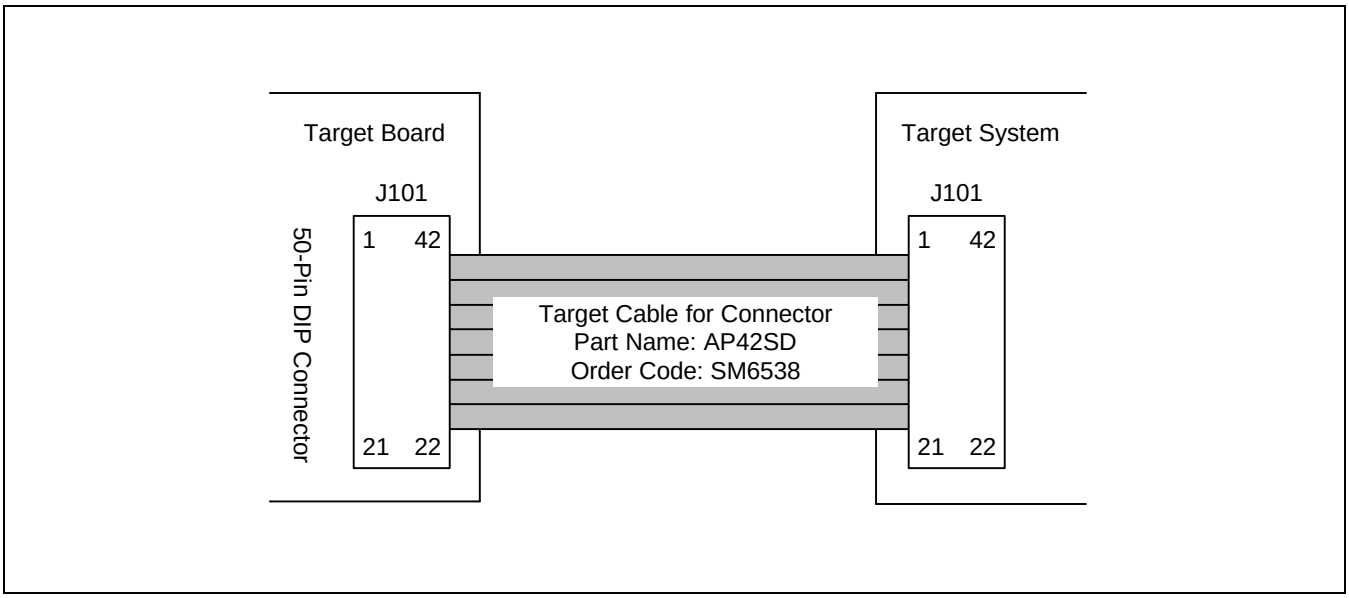


Figure 19-4. S3C9228 Probe Adapter for 42-SDIP Package

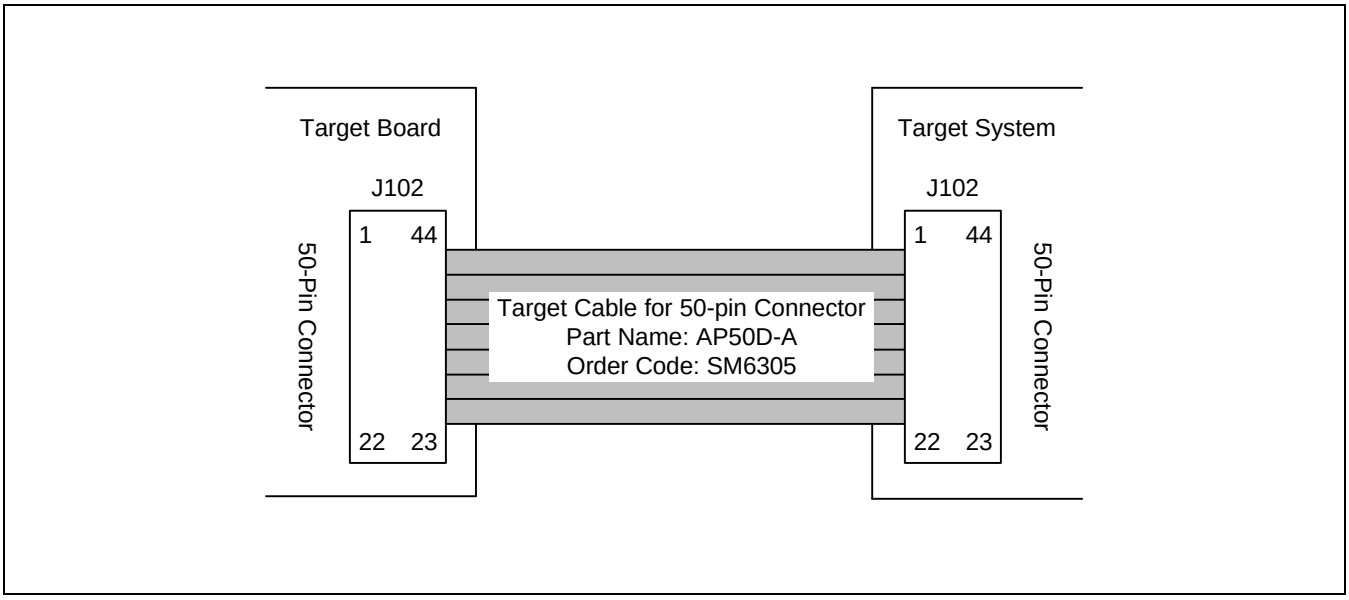


Figure 19-5. S3C9228 Probe Adapter for 44-QFP Package

NOTES

1 PRODUCT OVERVIEW

SAM88RCRI 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 supports 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.

S3C9228/P9228 MICROCONTROLLER

The S3C9228 can be used for dedicated control functions in a variety of applications, and is especially designed for application with FRS or etc.

The S3C9228/P9228 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 S3C9228/P9228 has 8K-byte of program ROM, and 264-byte of RAM (including 16-byte of working register and 20-byte LCD display RAM).

Using the SAM88RCRI design approach, the following peripherals were integrated with the SAM88RCRI core:

— 7 configurable I/O ports including ports shared with segment/common drive outputs
— 10-bit programmable pins for external interrupts
— One 8-bit basic timer for oscillation stabilization and watch-dog functions
— Two 8-bit timer/counters with selectable operating modes
— Watch timer for real time
— 4 channel A/D converter
— 8-bit serial I/O interface

OTP

The S3C9228 microcontroller is also available in OTP (One Time Programmable) version. S3P9228 microcontroller has an on-chip 8K-byte one-time-programmable EPROM instead of masked ROM. The S3P9228 is comparable to S3C9228, both in function and in pin configuration.

FEATURES

CPU

- SAM88RCRI CPU core

Memory

- 8192 × 8 bits program memory (ROM)
- 264 × 8 bits data memory (RAM)
  (Including LCD data memory)

Instruction Set

- 41 instructions
- Idle and Stop instructions added for power-down modes

36 I/O Pins

- I/O: 34 pins (44-pin QFP, 42-pin SDIP)
- Output only: 2 pins (44-pin QFP)

Interrupts

- 14 interrupt source and 1 vector
- One interrupt level

8-Bit Basic Timer

- Watchdog timer function
- 3 kinds of clock source

Two 8-Bit Timer/Counters

- The programmable 8-bit timer/counters
- External event counter function
- Configurable as one 16-bit timer/counters

Watch Timer

- Interval time: 3.91mS, 0.25S, 0.5S, and 1S at 32.768 kHz
- 0.5/1/2/4 kHz Selectable buzzer output
- Clock source generation for LCD

LCD Controller/Driver

- 16 segments and 8 common terminals
- 3, 4, and 8 common selectable
- Internal resistor circuit for LCD bias

8-bit Serial I/O Interface

- 8-bit transmit/receive mode
- 8-bit receive mode
- LSB-first or MSB-first transmission selectable
- Internal or external clock source

A/D Converter

- 10-bit converter resolution
- 50us conversion speed at 1MHz f_{ADC} clock
- 4-channel

Two Power-Down Modes

- Idle: only CPU clock stops
- Stop: system clock and CPU clock stop

Oscillation Sources

- Crystal, ceramic, or RC for main clock
- Main clock frequency: 0.4 MHz - 8MHz
- 32.768 kHz crystal oscillation circuit for sub clock

Instruction Execution Times

- 500nS at 8MHz fx(minimum)

Operating Voltage Range

- 2.0 V to 5.5 V at 0.4 - 4.2MHz
- 2.7 V to 5.5 V at 0.4 - 8MHz

Operating Temperature Range

- -25 °C to +85 °C

Package Type

- 44-pin QFP, 42-pin SDIP

BLOCK DIAGRAM

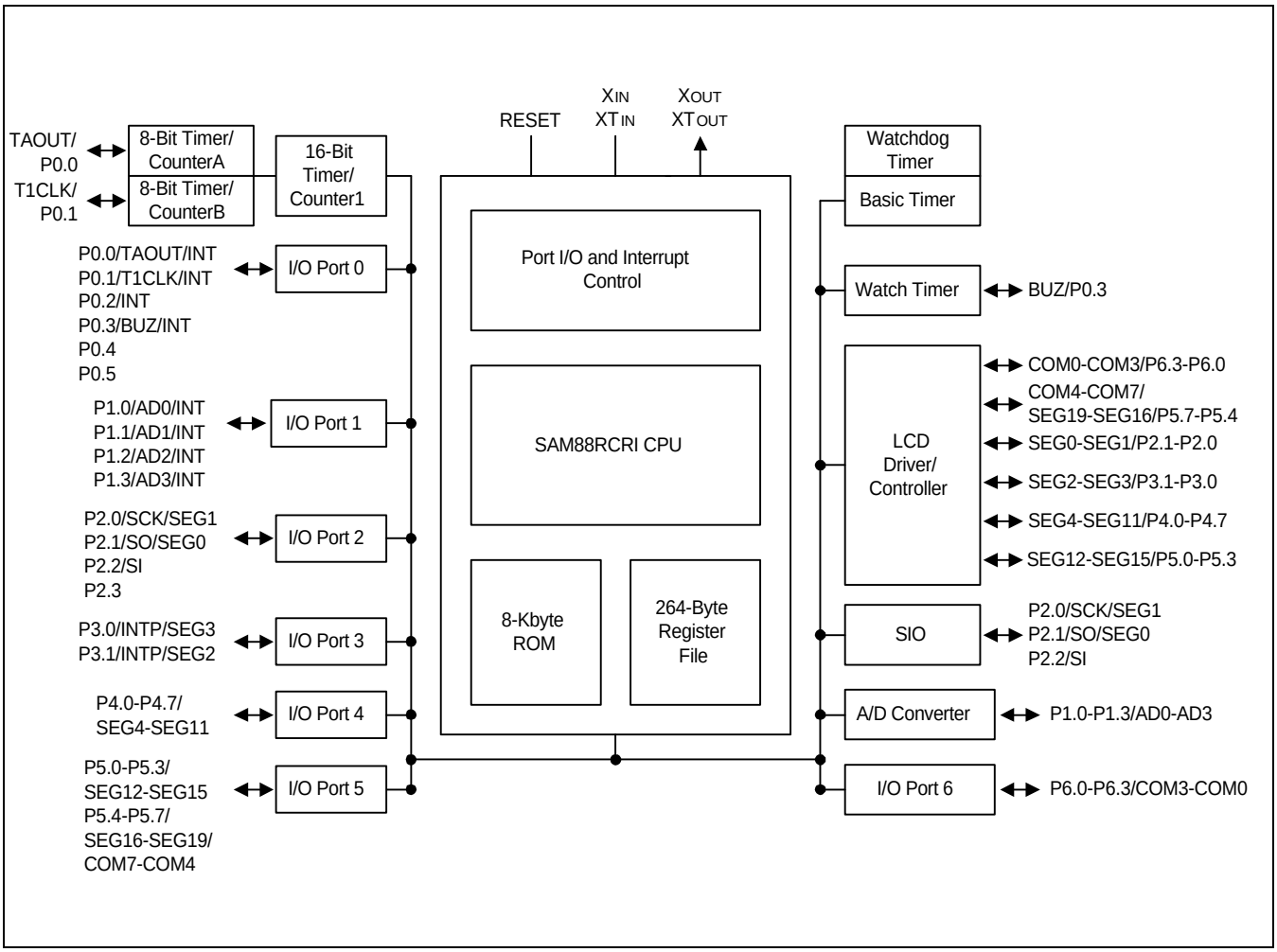


Figure 1-1. Block Diagram

PIN ASSIGNMENTS

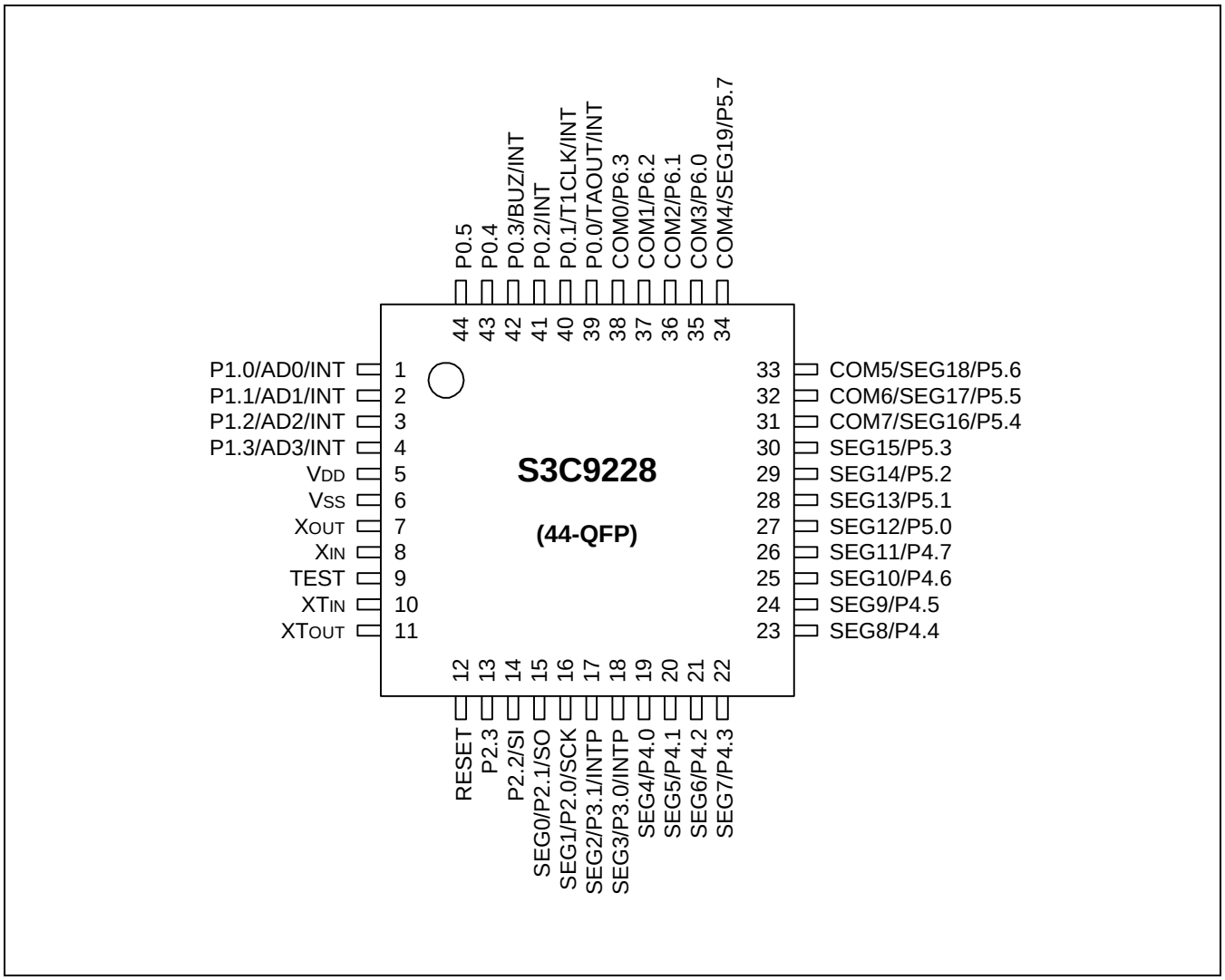


Figure 1-2. S3C9228 44-QFP Pin Assignments

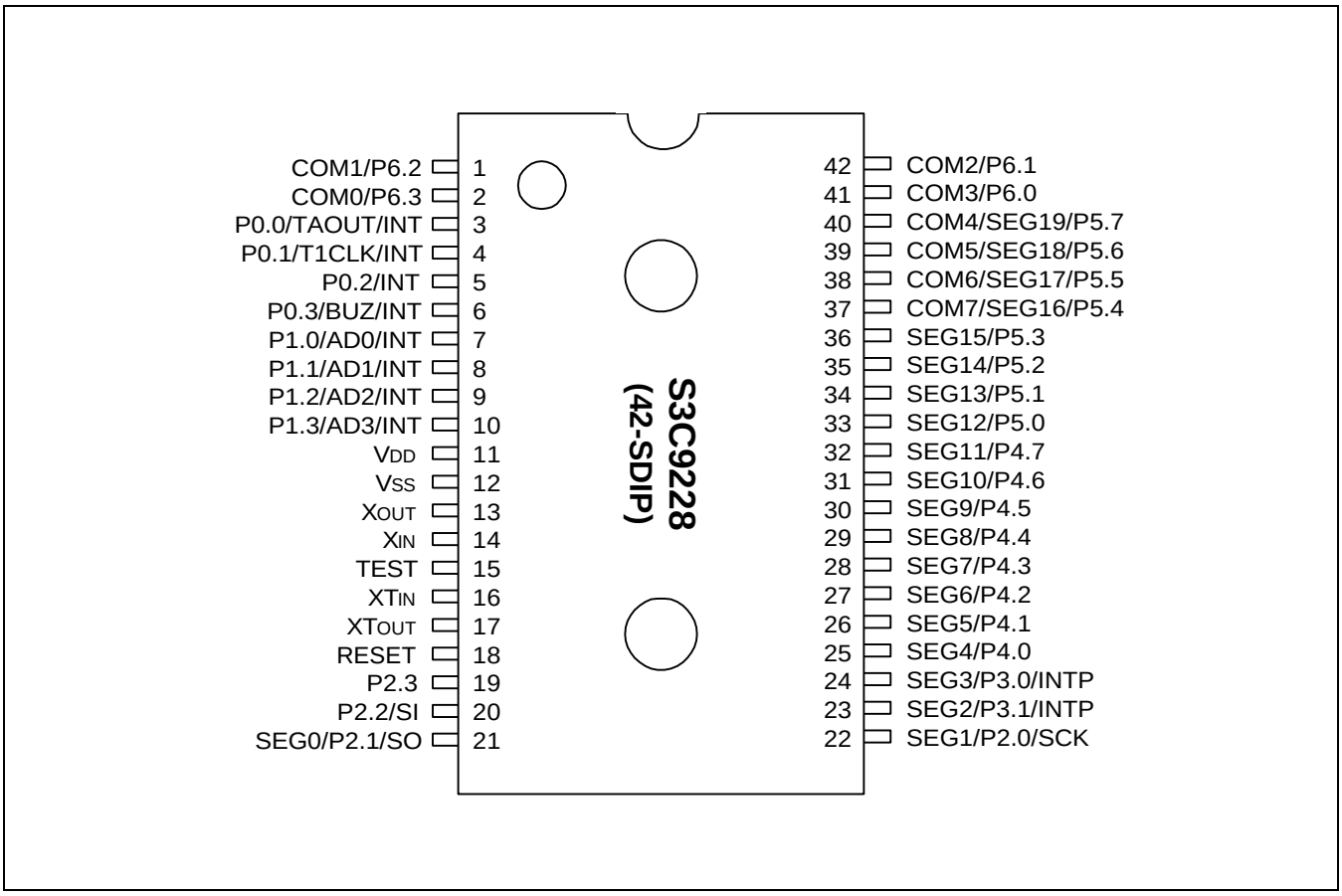


Figure 1-3. S3C9228 42-SDIP Pin Assignments

PIN DESCRIPTIONS

Table 1-1. Pin Descriptions

Pin Names	Pin Type	Pin Description	Circuit Number	Pin Numbers	Share Pins
P0.0 P0.1 P0.2 P0.3	I/O	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	E-4	39(3) 40(4) 41(5) 42(6)	TAOUT/INT T1CLK/INT INT BUZ/INT
P0.4-P0.5	O	1-bit programmable output port.	C	43-44	
P1.0 P1.1 P1.2 P1.3	I/O	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	F-16A	1(7) 2(8) 3(9) 4(10)	AD0/INT AD1/INT AD2/INT AD3/INT
P2.0 P2.1	I/O	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain	H-32A	16(22) 15(21)	SCK/SEG1 SO/SEG0
P2.2 P2.3	I/O	output and software assignable pull-ups.	E-4	14(20) 13(19)	SI –
P3.0 P3.1	I/O	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	H-32B	18(24) 17(23)	INTP/SEG3 INTP/SEG2
P4.0–P4.7	I/O	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups.	H-32	19-26(25-32)	SEG4-SEG11
P5.0–P5.3	I/O	1-bit programmable I/O port. Input or push-pull, open-drain output and	H-32	27-30(33-36)	SEG12-SEG15
P5.4–P5.7		software assignable pull-ups.		31-34(37-40)	SEG16-SEG19 /COM7-COM4
P6.0-P6.3	I/O	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups.	H-32	35-38 (41-42,1-2)	COM3-COM0

NOTE: Parentheses indicate pin number for 42-SDIP-600 package.

Table 1-1. Pin Descriptions (Continued)

Pin Names	Pin Type	Pin Description	Circuit Number	Pin Numbers	Share Pins
V_{DD}, V_{SS}	–	Power input pins for internal power block	–	5,6(11,12)	–
X_{OUT}, X_{IN}	–	Main oscillator pins for main clock	–	7,8(13,14)	
XT_{OUT}, XT_{IN}	–	Sub oscillator pins for sub clock	–	11,10(17,16)	–
TEST	–	Chip test input pin Hold GND when the device is operating	–	9(15)	–
RESET	I	RESET signal input pin. Schmitt trigger input with internal pull-up resistor.	B	12(18)	–
INT	I/O	External interrupts input.	E-4 F-16A	39-42(3-6) 1-4(7-10)	P0.0-P0.3 P1.0-P1.3
INTP	I/O	Key scan interrupts inputs.	H-32B	17-18(23-24)	P3.1-P3.0
T1CLK	I/O	Timer 1/A external clock input.	E-4	40(4)	P0.1
TAOUT	I/O	Timer 1/A clock output.	E-4	39(3)	P0.0
AD0-AD3	I/O	Analog input pins for A/D converts module.	F-16A	1-4(7-10)	P1.0-P1.3
BUZ	I/O	Buzzer signal output.	E-4	42(6)	P0.3
SCK SO	I/O	Serial clock, serial data output, serial data input	H-32A	16-15(22-21)	P2.0-P2.1
SI			E-4	14(20)	P2.2
SEG0-SEG1	I/O	LCD segment signal output	H-32A	15-16(21-22)	P2.1-P2.0
SEG2-SEG3			H-32B	17-18(23-24)	P3.1-P3.0
SEG4-SEG19			H-32	19-34(25-40)	P4.0-P4.7 P5.0-P5.7
COM0-COM7	I/O	LCD common signal output	H-32	38-31 (2-1,42-37)	P6.3-P6.0 P5.7-P5.4

NOTE: Parentheses indicate pin number for 42-SDIP-600 package.

PIN CIRCUIT DIAGRAMS

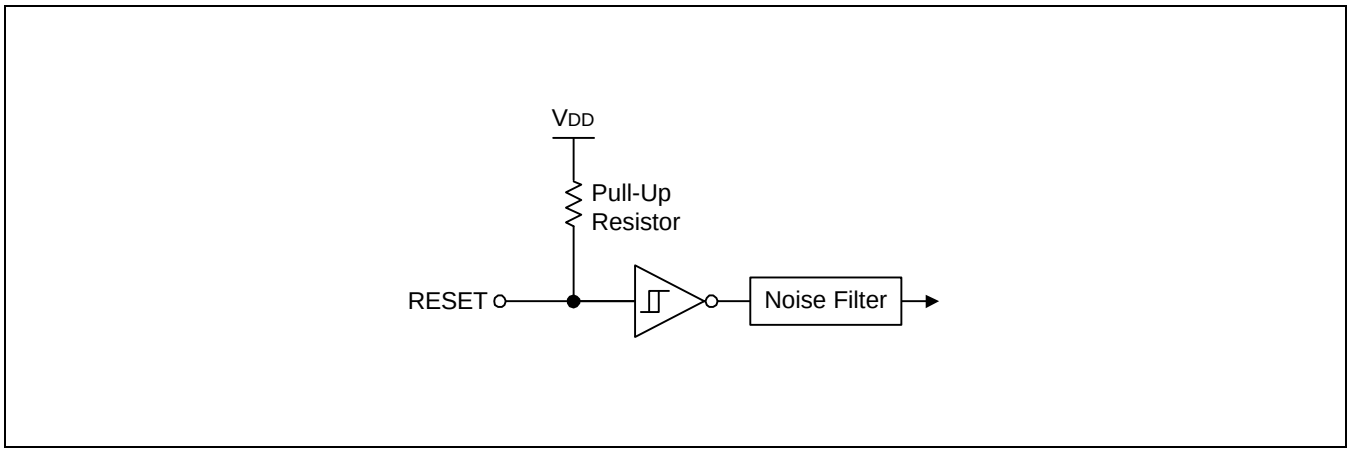


Figure 1-4. Pin Circuit Type B

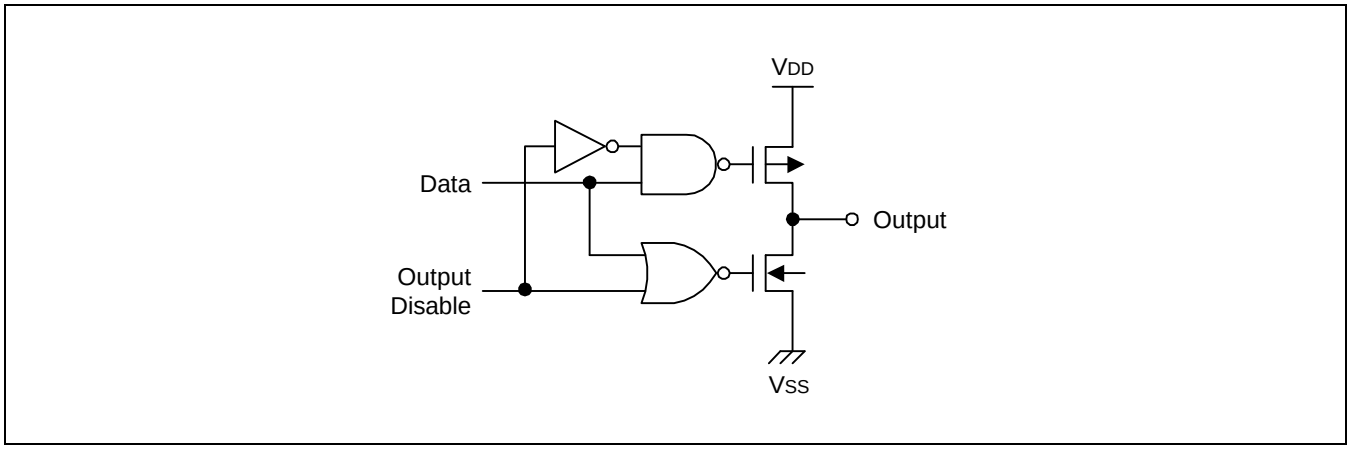


Figure 1-5. Pin Circuit Type C

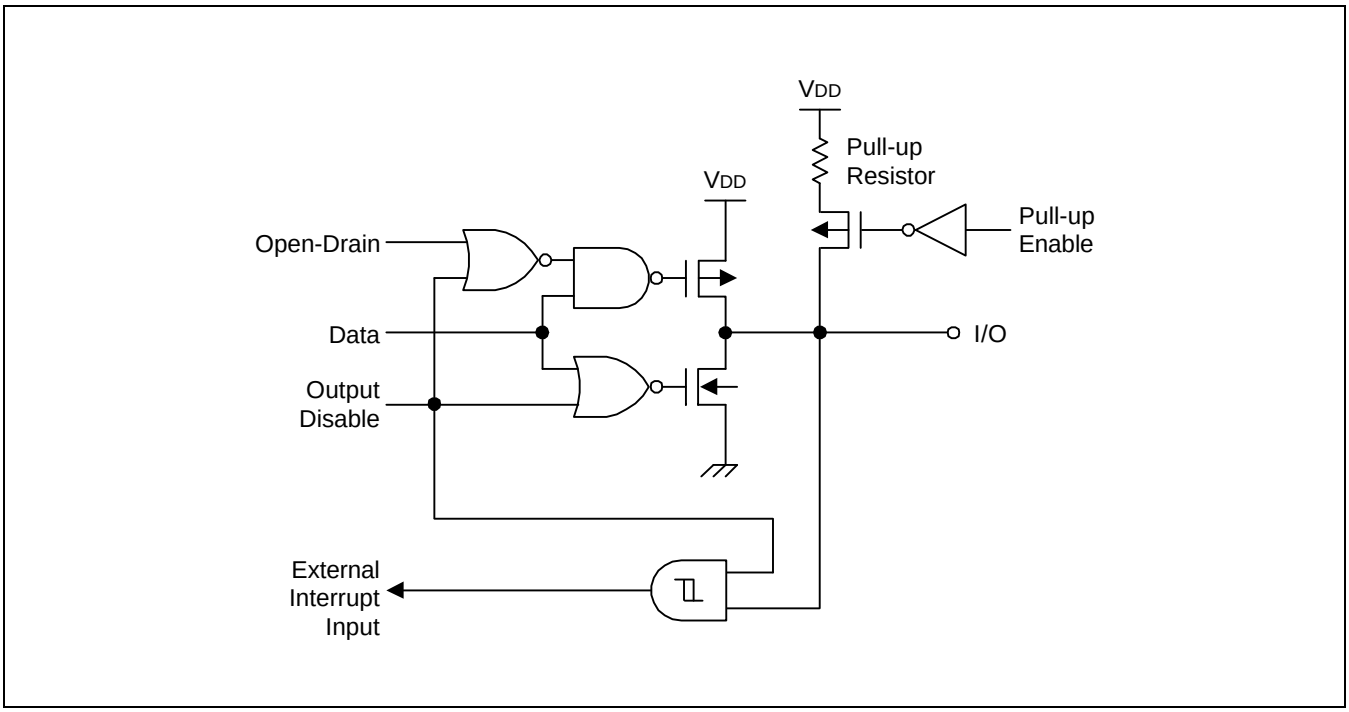


Figure 1-7. Pin Circuit Type E-4

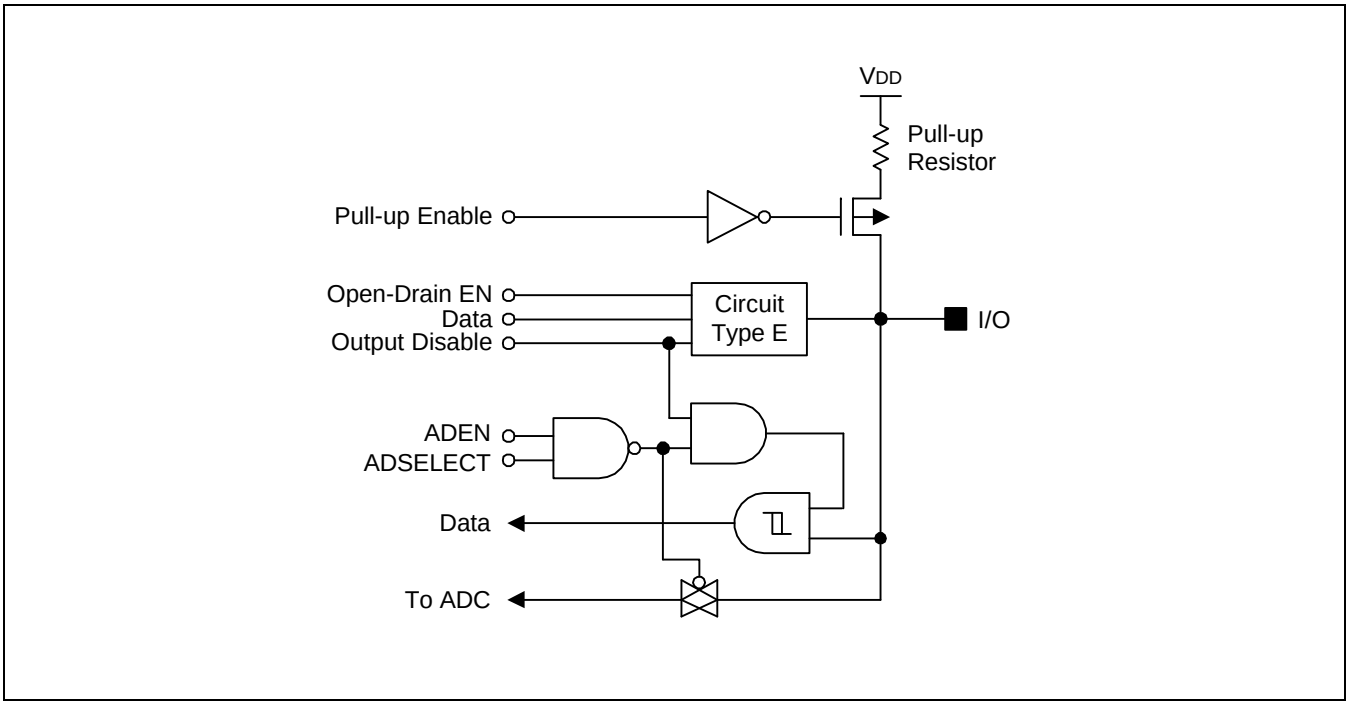


Figure 1-8. Pin Circuit Type F-16A

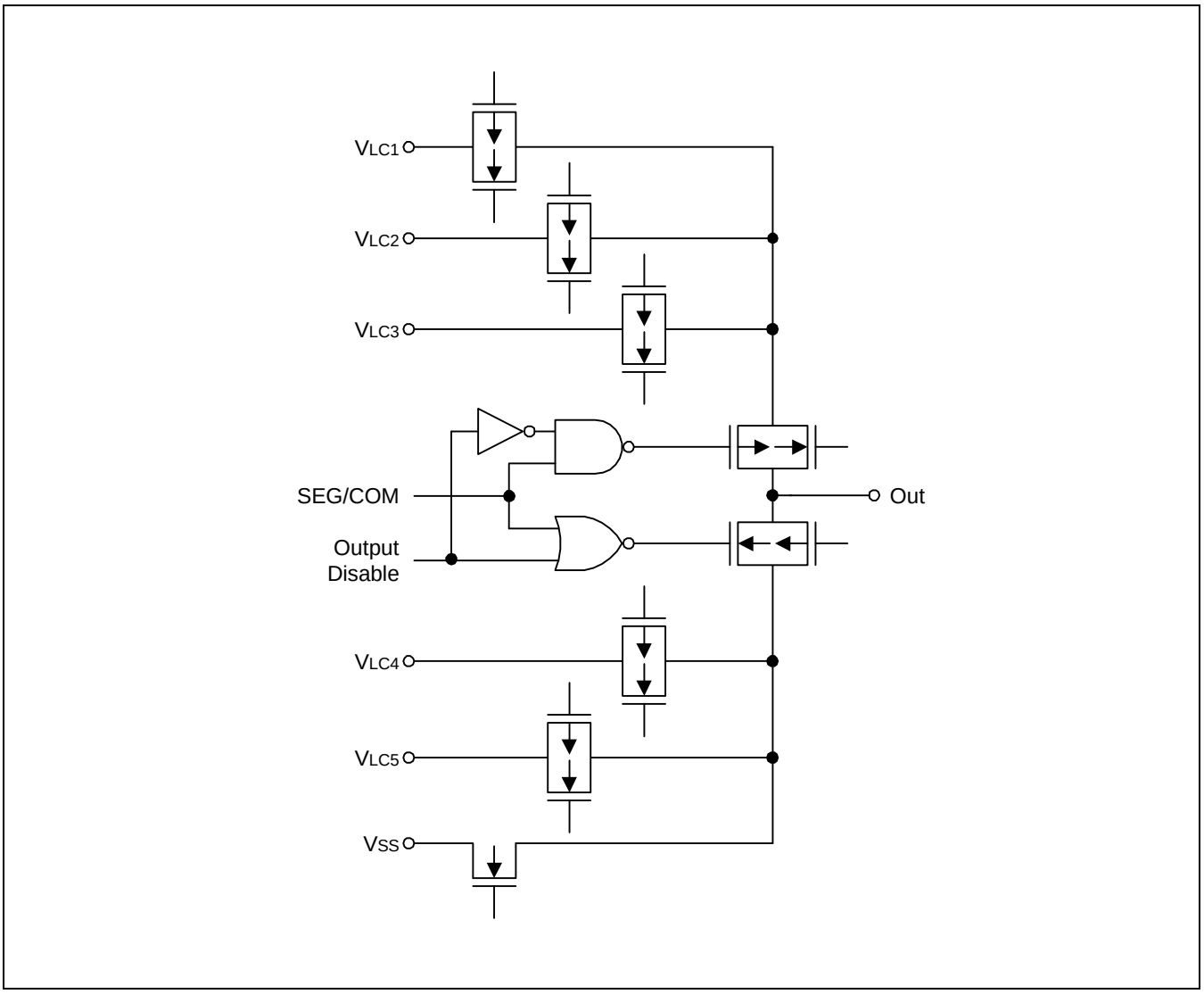


Figure 1-9. Pin Circuit Type H-23

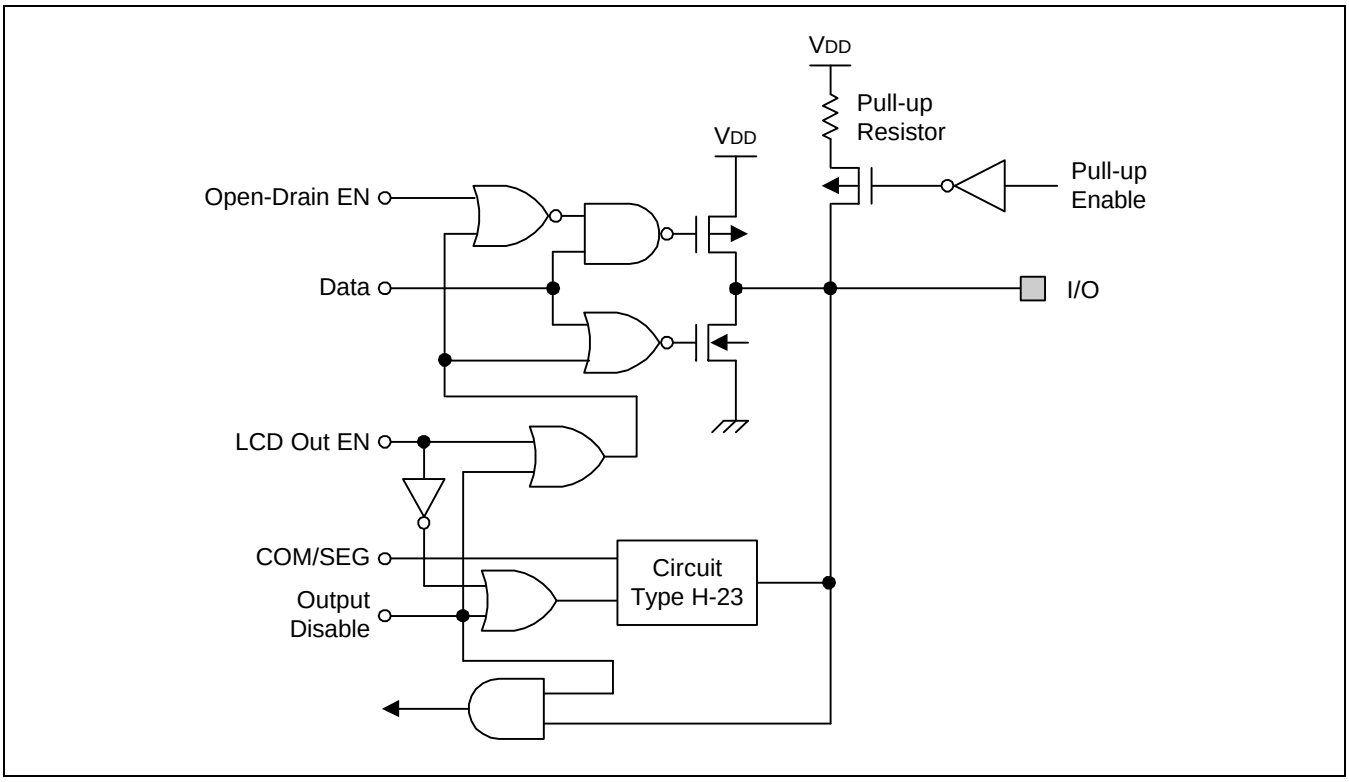


Figure 1-10. Pin Circuit Type H-32

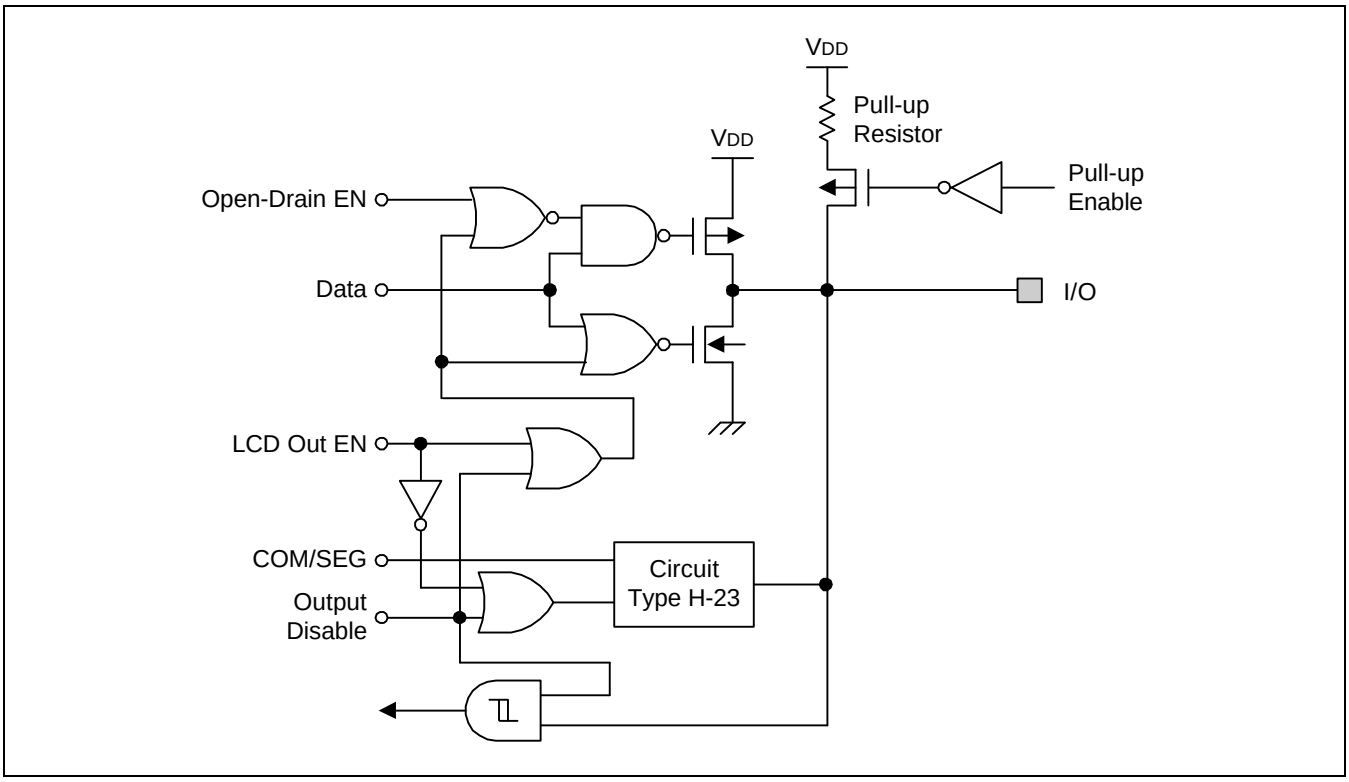


Figure 1-11. Pin Circuit Type H-32A

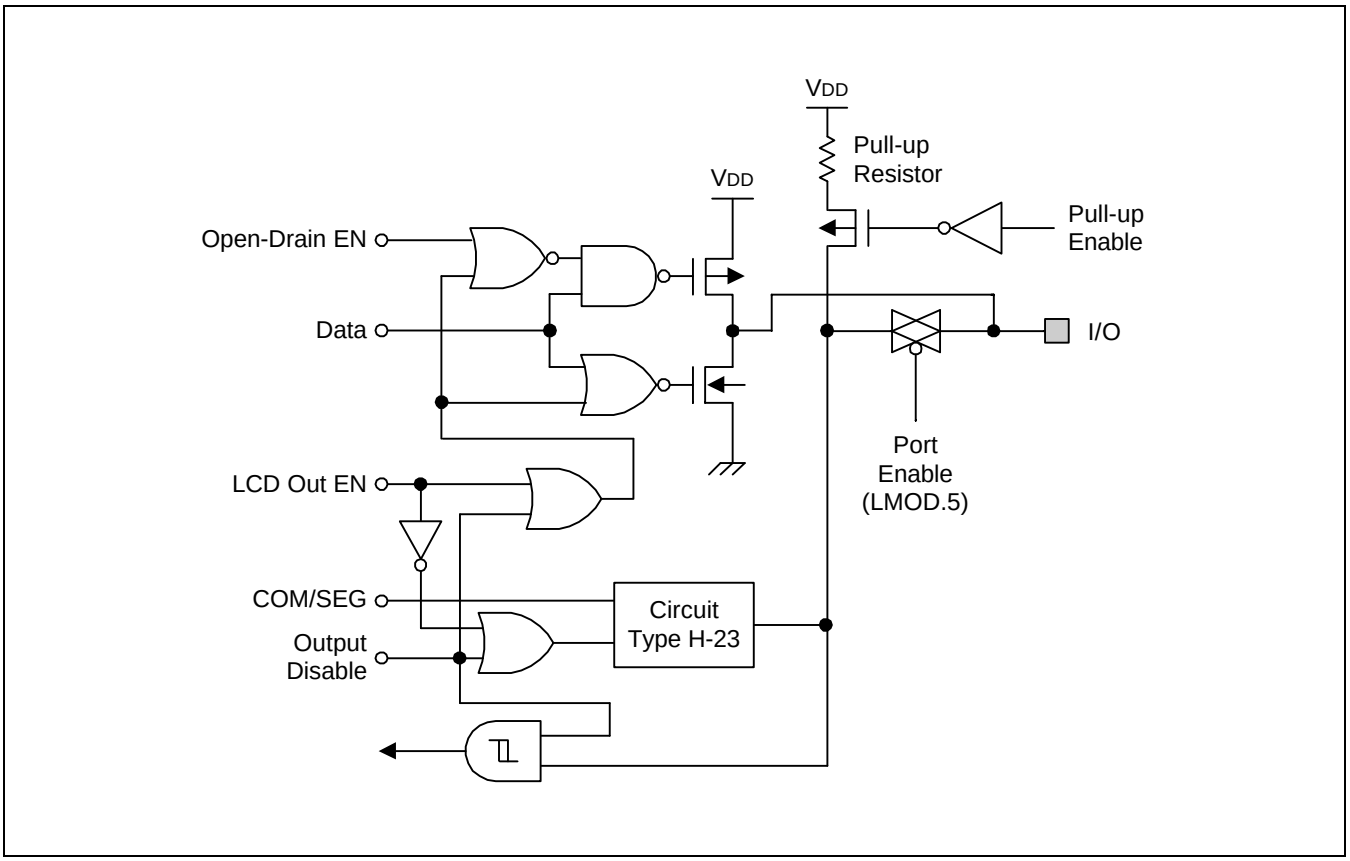


Figure 1-12. Pin Circuit Type H-32B

2 ADDRESS SPACES

OVERVIEW

The S3C9228/P9228 microcontroller has three kinds of address space:

— Program memory (ROM)
— Internal register file
— LCD display register file

A 16-bit address bus supports program memory operations. Special instructions and related internal logic determine when the 16-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 S3C9228 has 8K bytes of mask-programmable program memory on-chip. The S3C9228/P9228 microcontroller has 244 bytes general-purpose registers in its internal register file and the 20 bytes for LCD display memory is implemented in the internal register file too. Fifty-six bytes in the register file are mapped for system and peripheral control functions.

PROGRAM MEMORY (ROM)

Program memory (ROM) stores program code or table data. The S3C9228 has 8K bytes of mask-programable program memory. The program memory address range is therefore 0H-1FFFH. The first 2 bytes of the ROM (0000H–0001H) are an interrupt vector address. The program reset address in the ROM is 0100H.

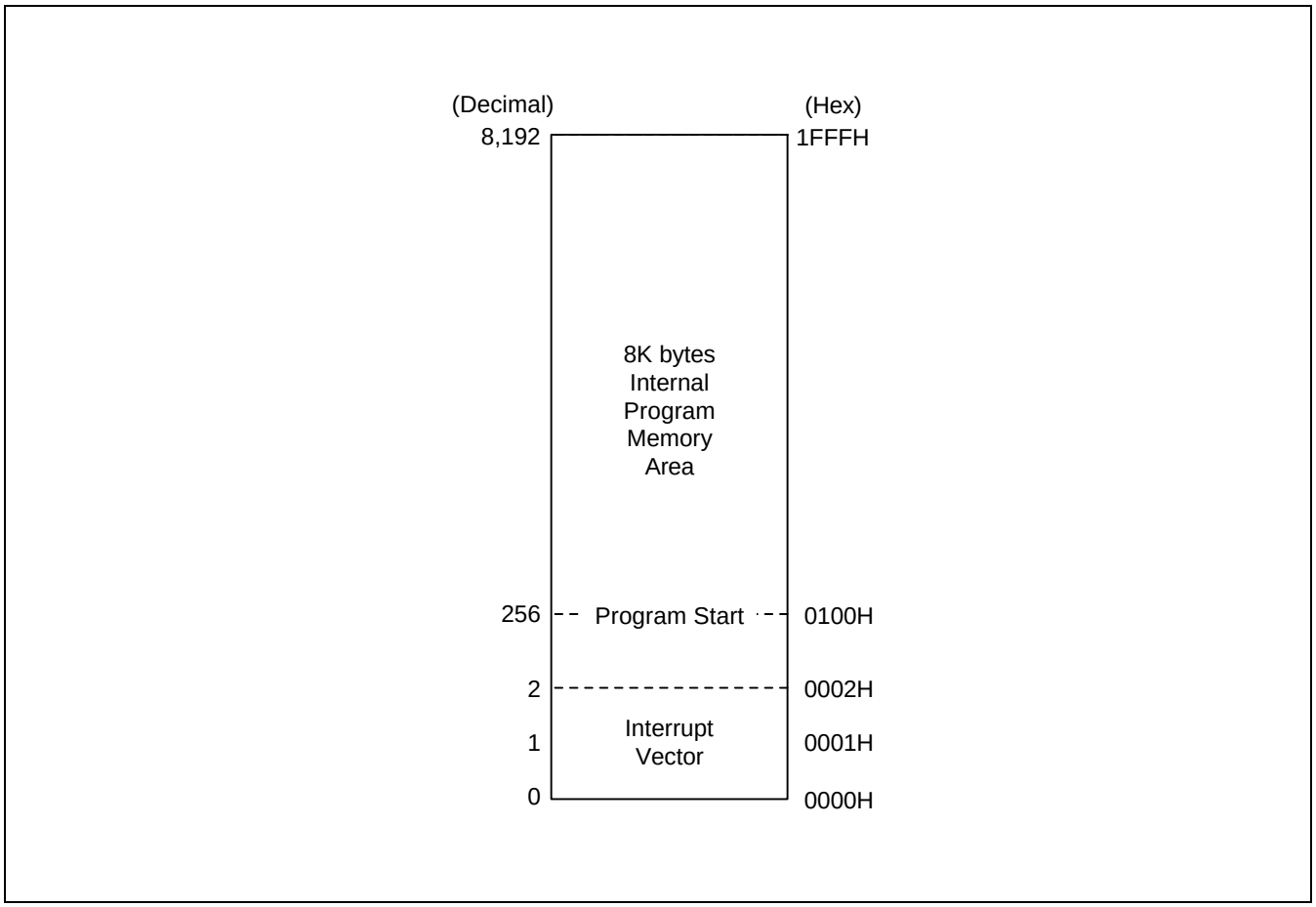


Figure 2-1. S3C9228/P9228 Program Memory Address Space

REGISTER ARCHITECTURE

The upper 72 bytes of the S3C9228/P9228's internal register file are addressed as working registers, system control registers and peripheral control registers. The lower 184 bytes of internal register file (00H–B7H) 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 implemented by page 1 in the S3C9228/P9228. The page 1 (20 × 8 bits) is for LCD display register and can be used as general-purpose registers.

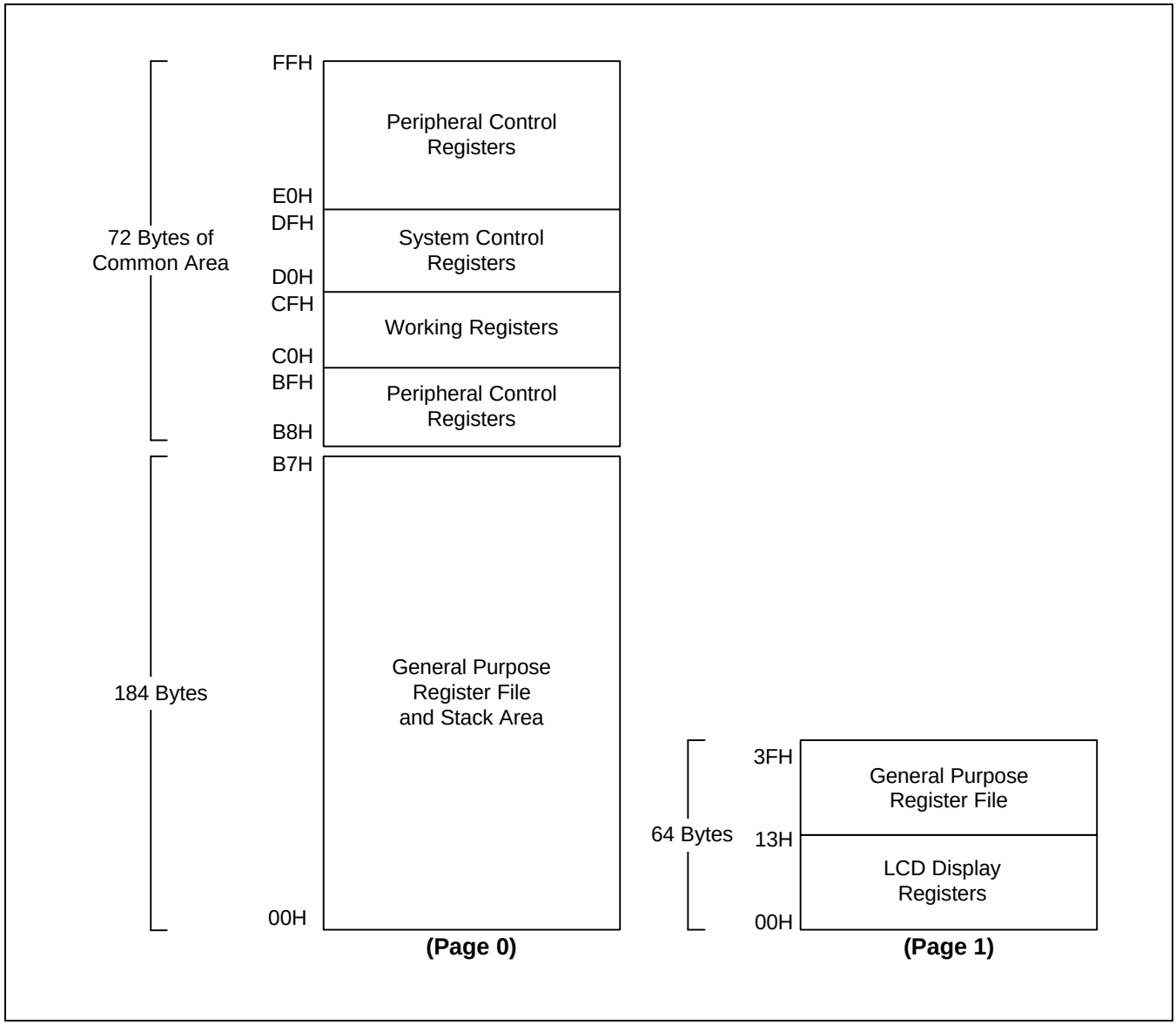


Figure 2-2. Internal Register File Organization

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.

This16-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.

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.

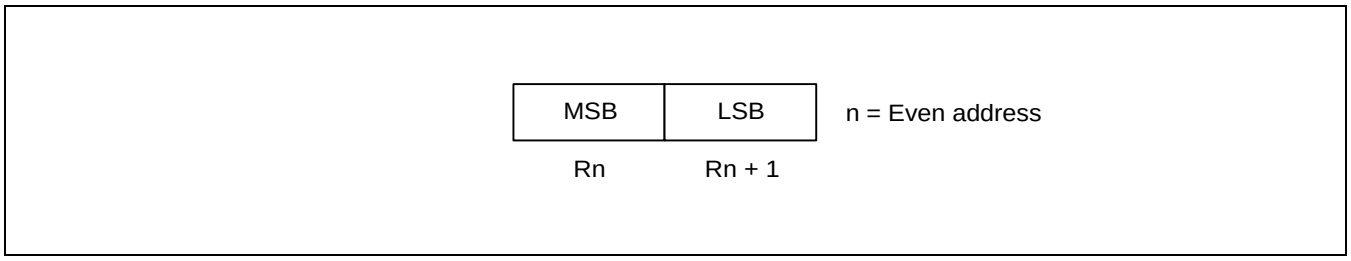


Figure 2-3. 16-Bit Register Pairs

☞ 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 S3C9228/P9228 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.

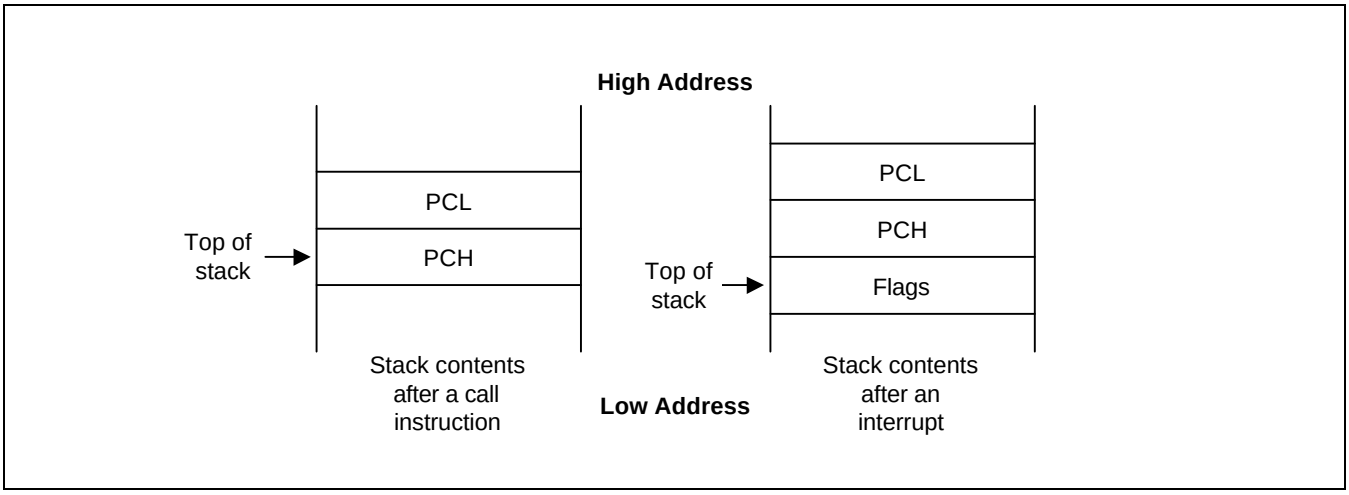


Figure 2-4. Stack Operations

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 S3C9228/P9228, the SP must be initialized to an 8-bit value in the range 00H–B7H.

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,#0B8H        ; SP ← B8H (Normally, the SP is set to 0B8H by the
                        ; initialization routine)
•
•
•
PUSH    SYM             ; Stack address 0B7H ← SYM
PUSH    WTCON           ; Stack address 0B6H ← WTCON
PUSH    20H             ; Stack address 0B5H ← 20H
PUSH    R3              ; Stack address 0B4H ← R3
•
•
•
POP     R3              ; R3 ← Stack address 0B4H
POP     20H             ; 20H ← Stack address 0B5H
POP     WTCON           ; WTCON ← Stack address 0B6H
POP     SYM             ; SYM ← Stack address 0B7H
```

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).

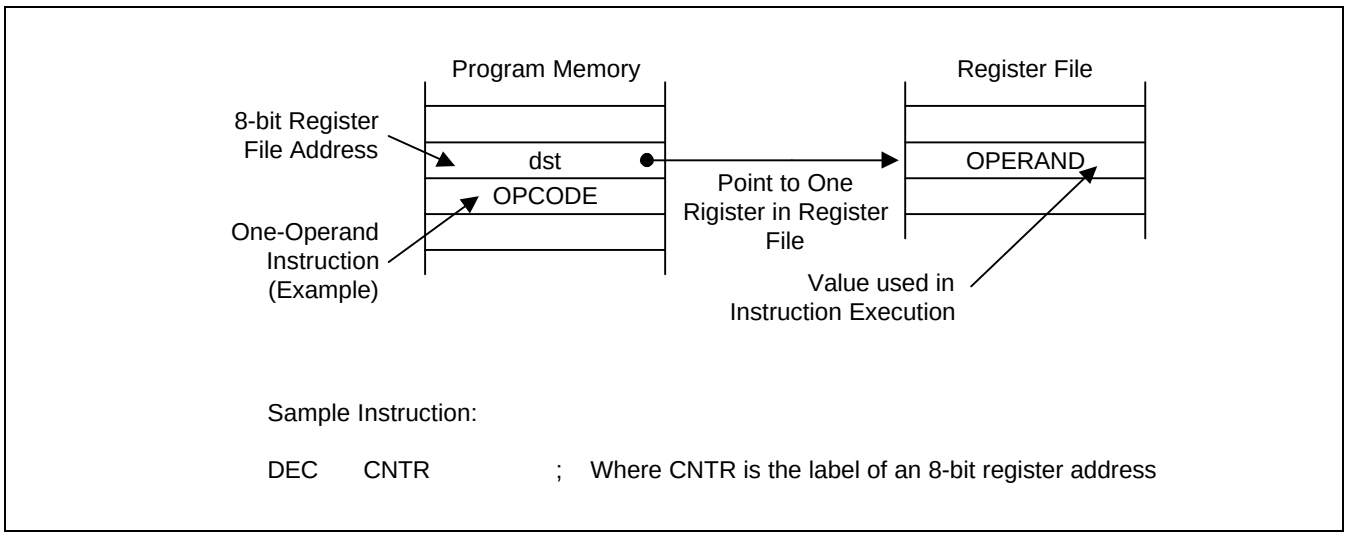


Figure 3-1. Register Addressing

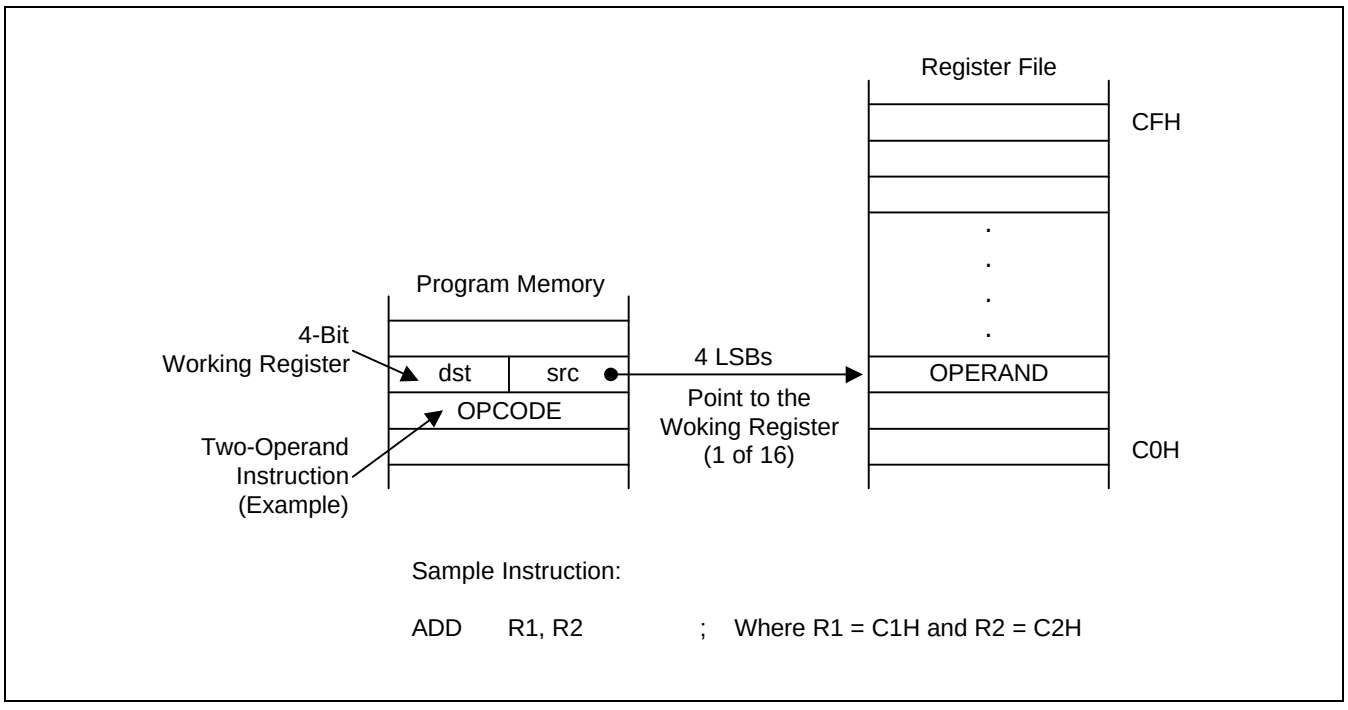


Figure 3-2. Working Register Addressing

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.

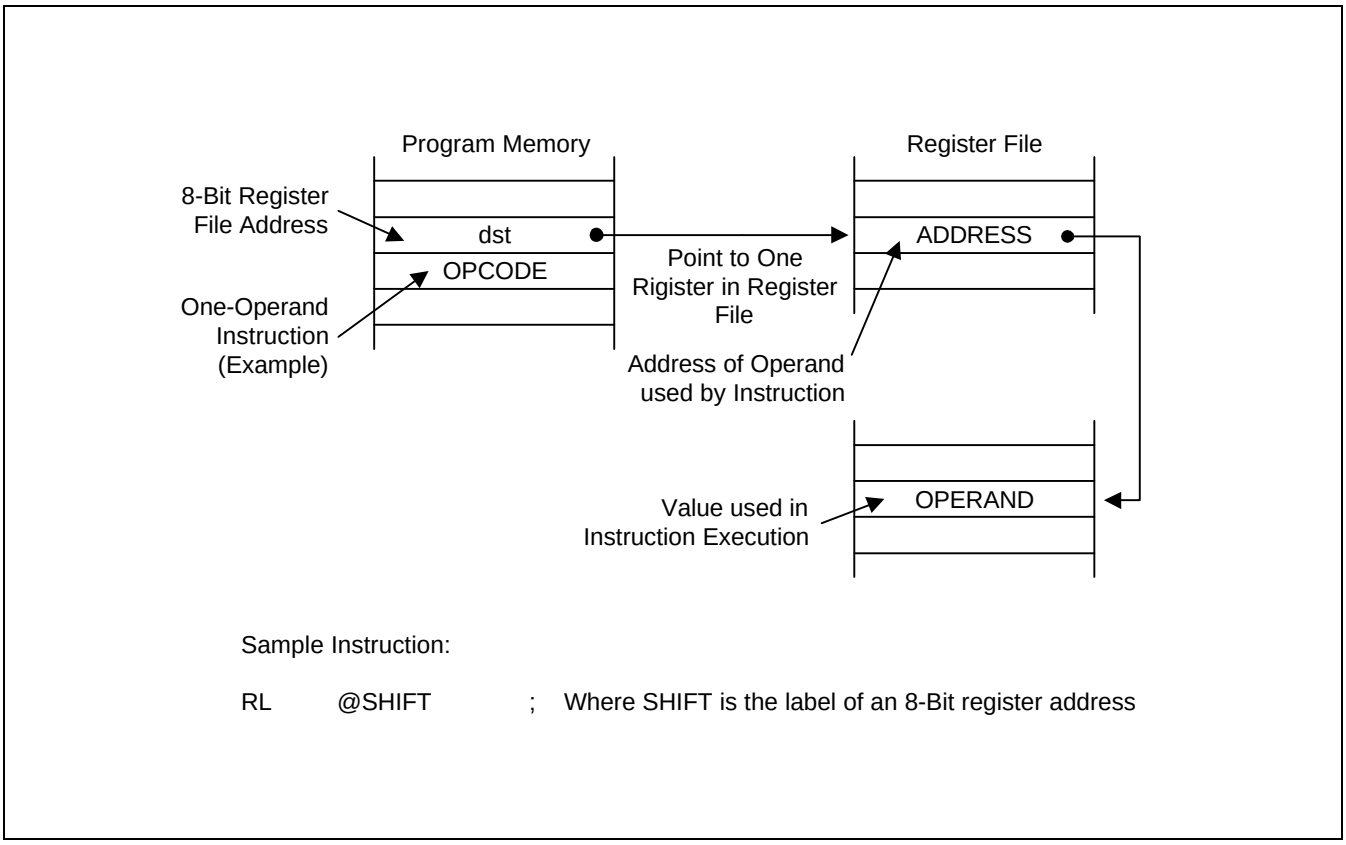


Figure 3-3. Indirect Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Continued)

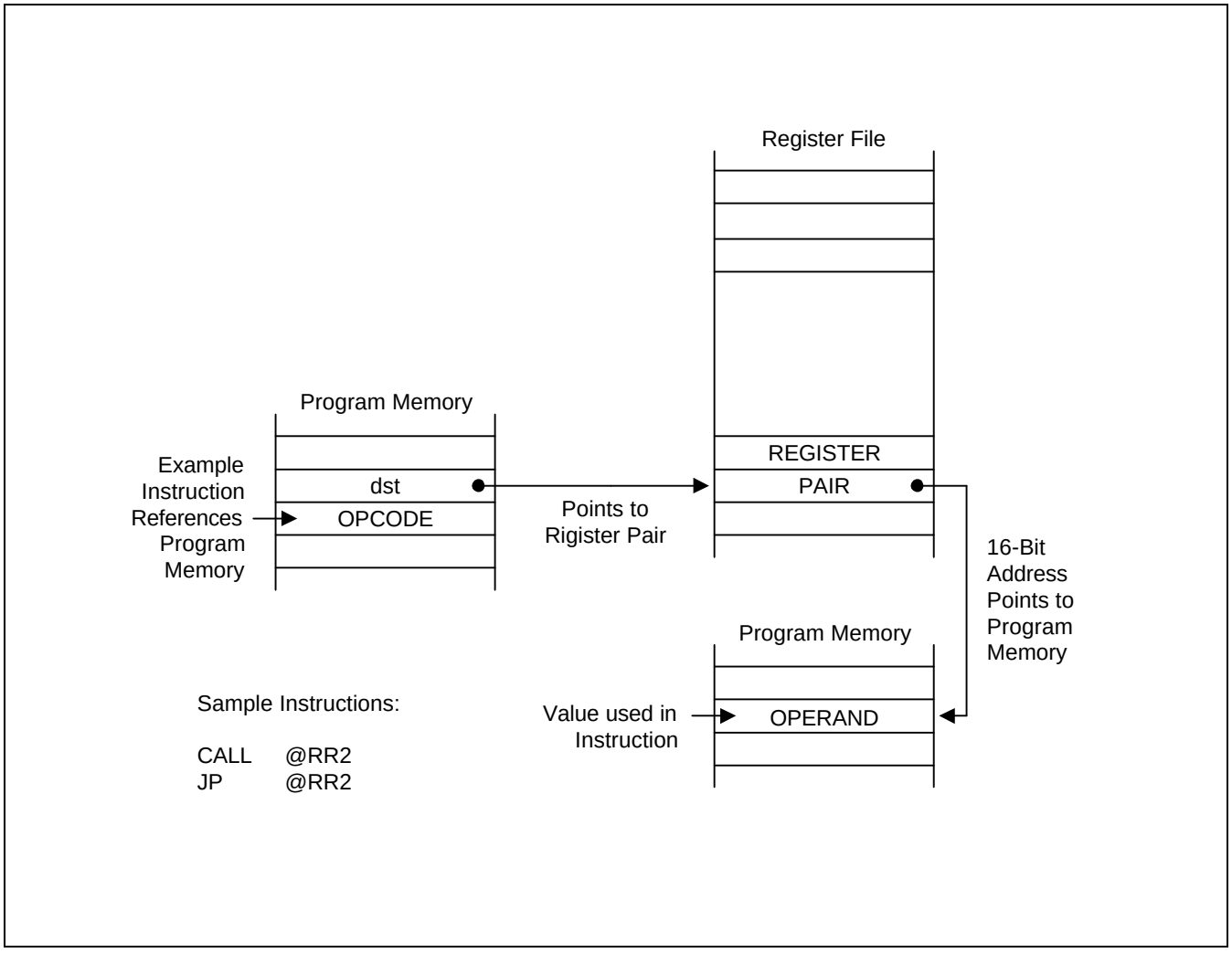


Figure 3-4. Indirect Register Addressing to Program Memory

INDIRECT REGISTER ADDRESSING MODE (Continued)

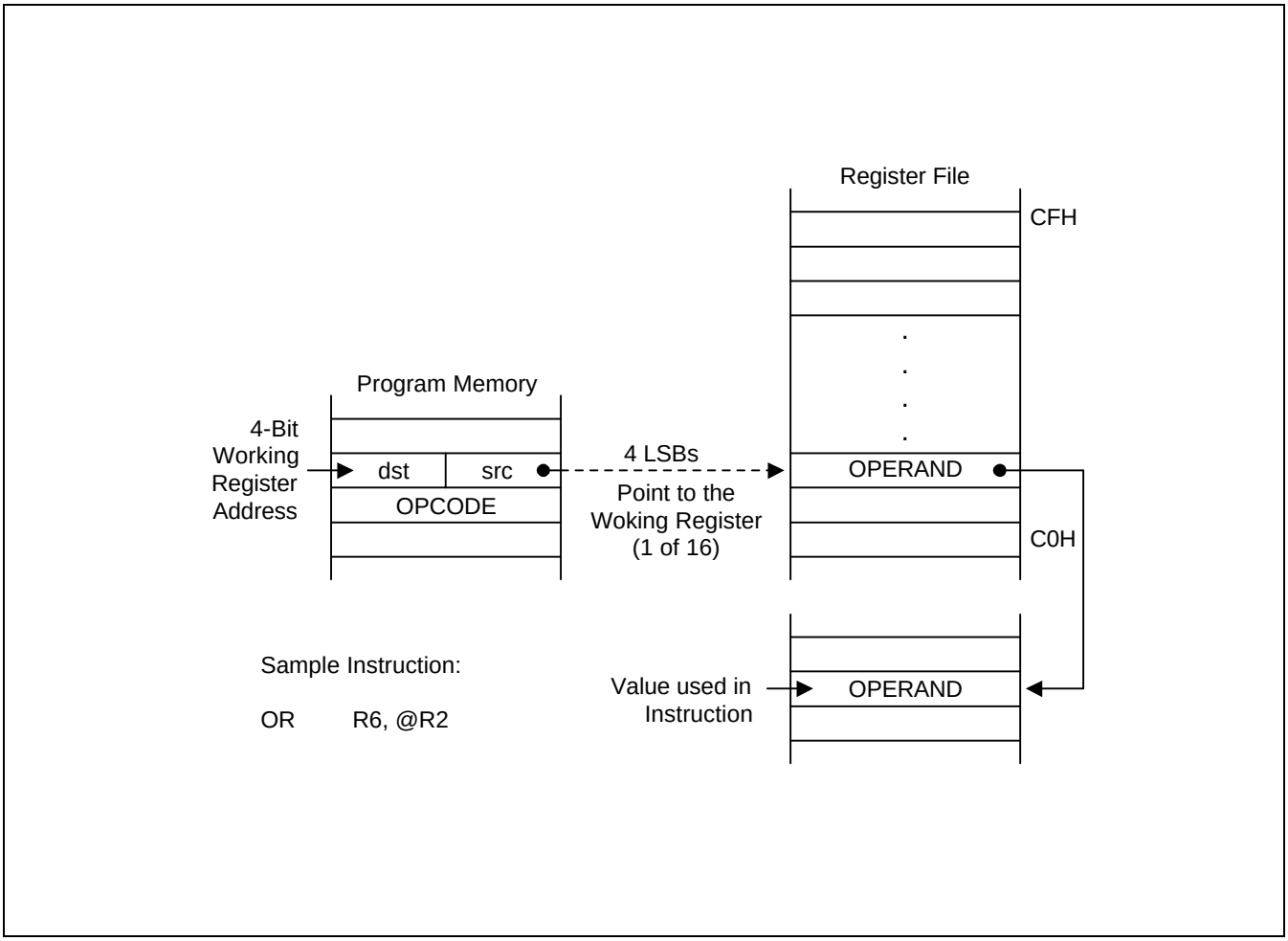


Figure 3-5. Indirect Working Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Concluded)

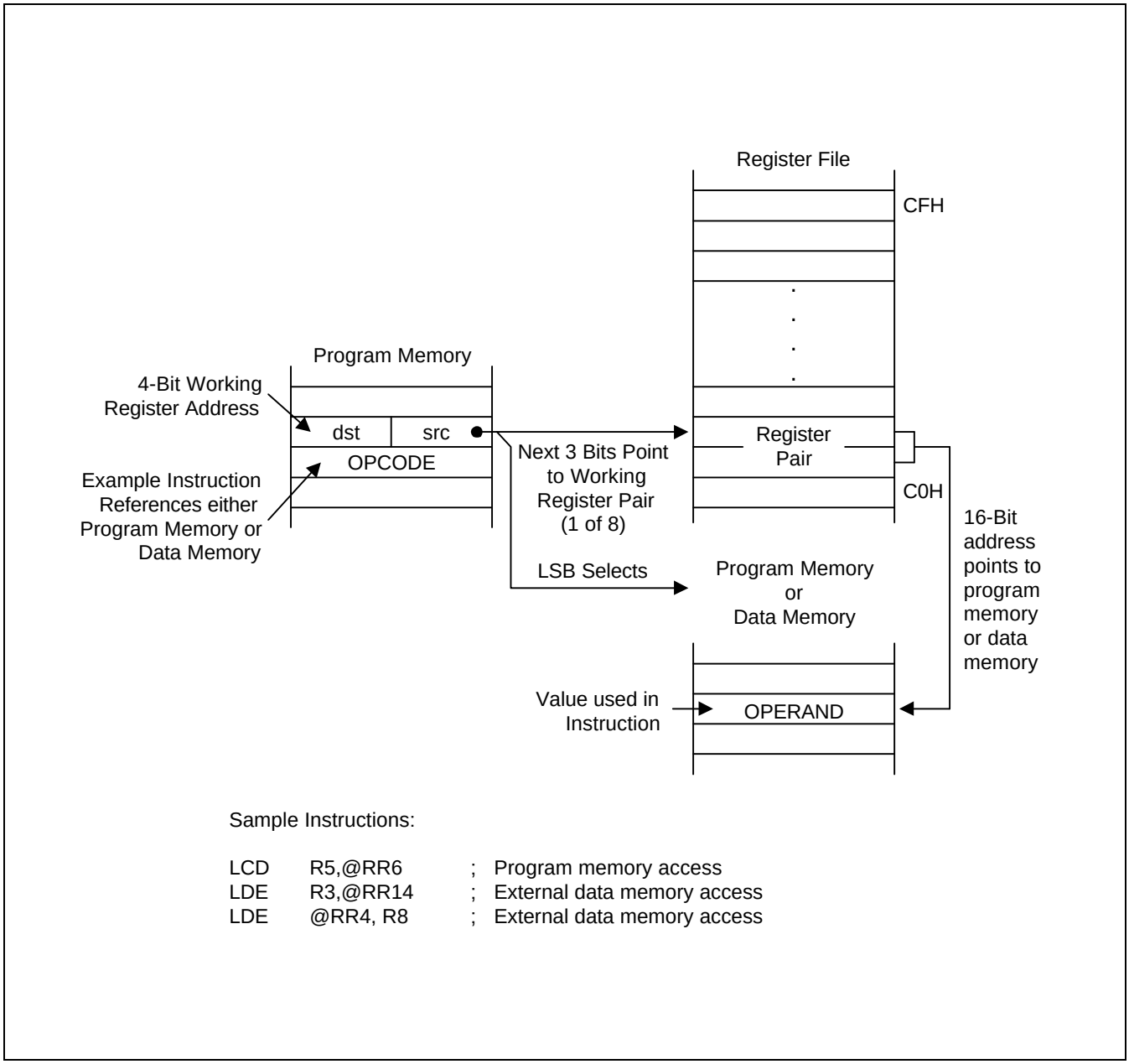


Figure 3-6. Indirect Working Register Addressing to Program or Data Memory

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.

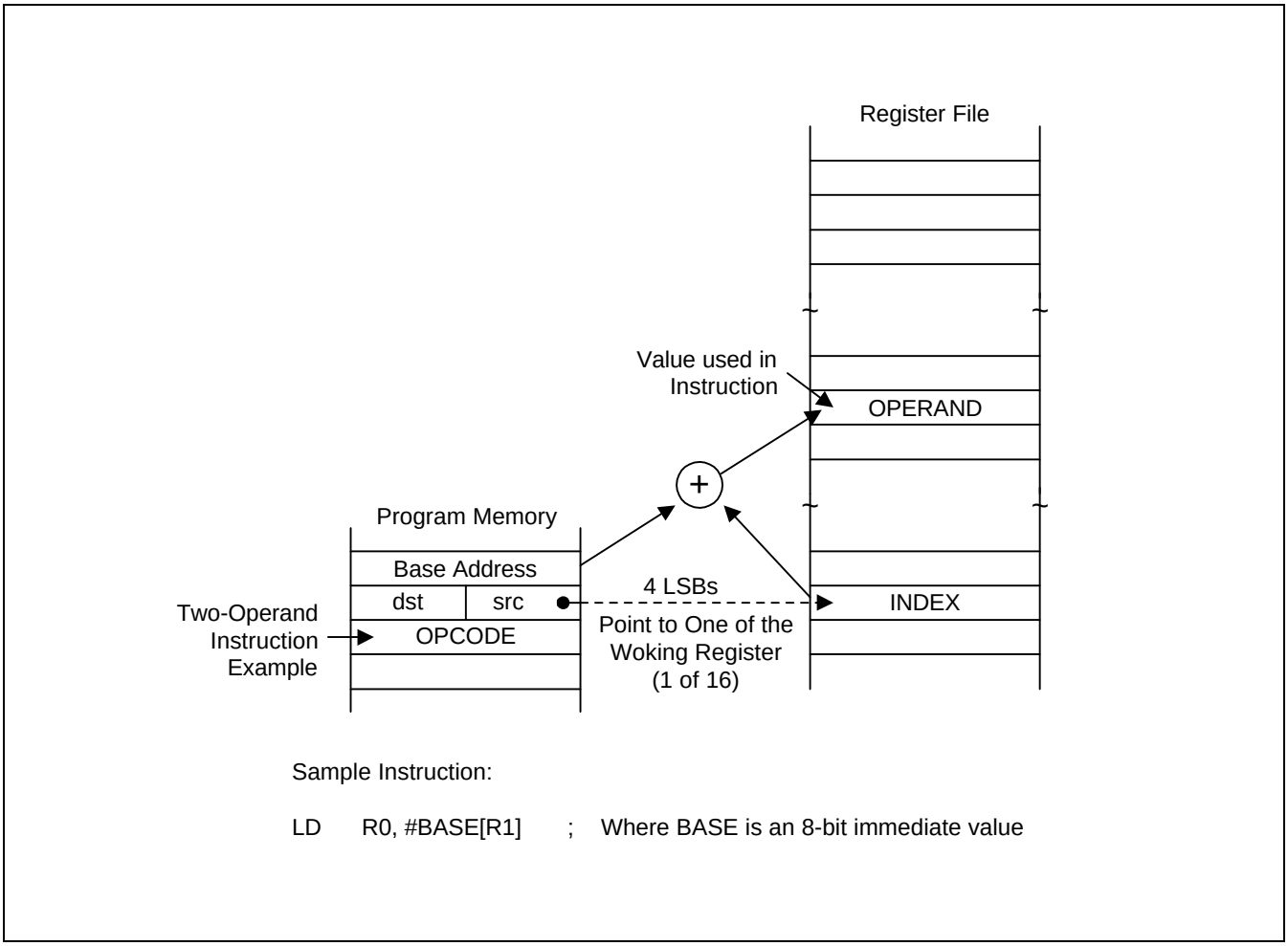
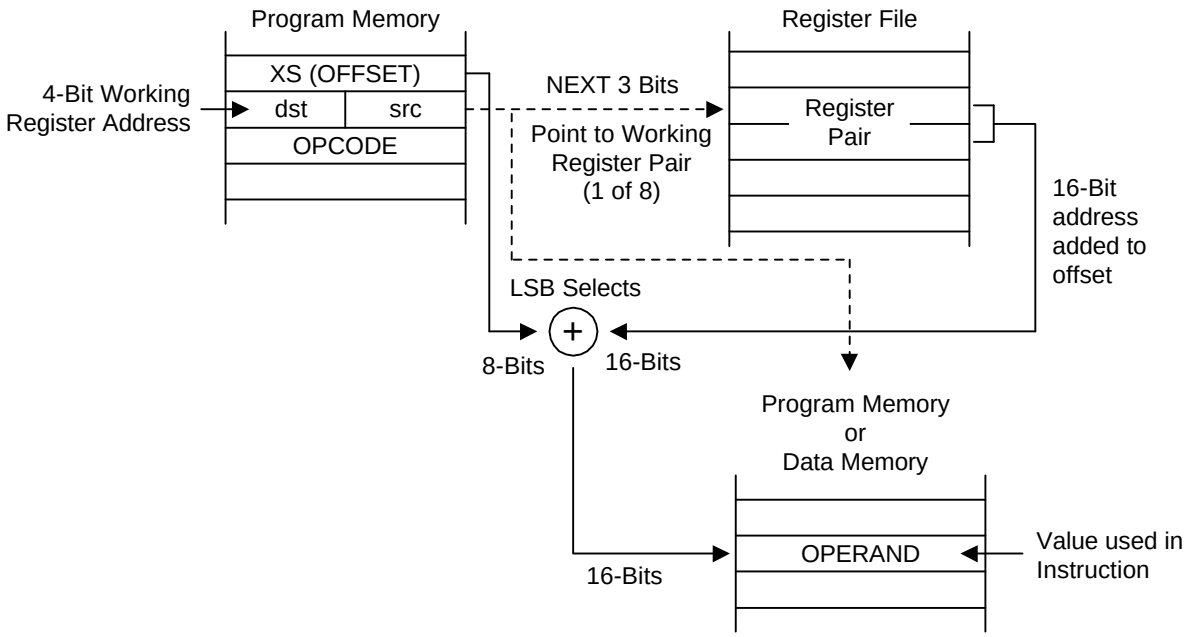


Figure 3-7. Indexed Addressing to Register File

INDEXED ADDRESSING MODE (Continued)

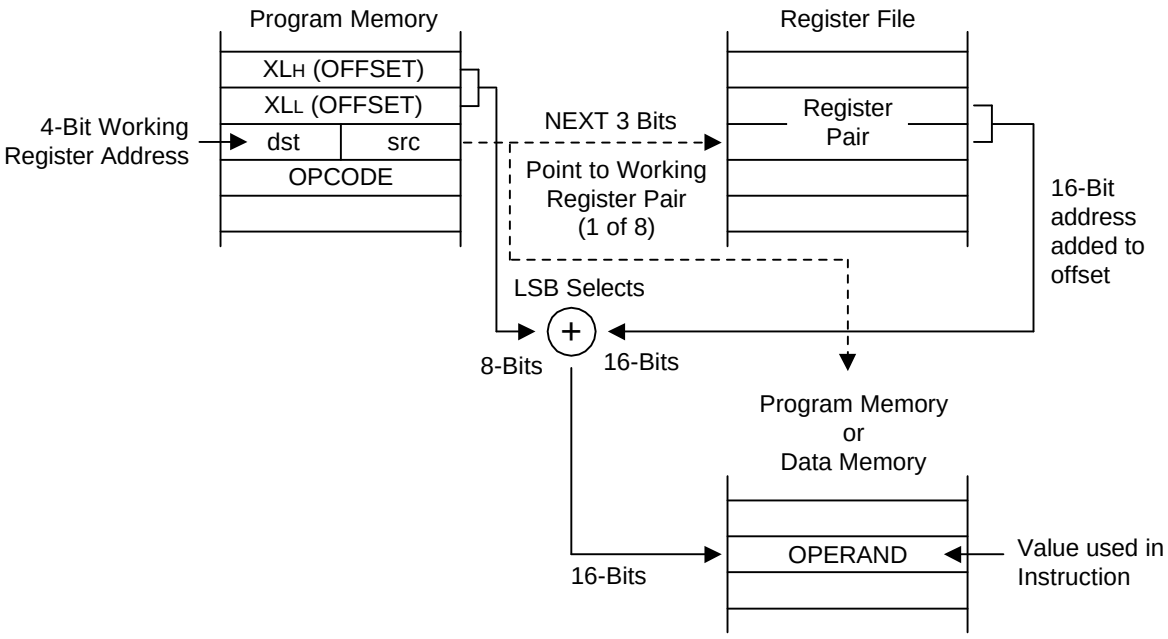


Sample Instructions:

```
LDC     R4, #04H[RR2]     ; The values in the program address (RR2 + #04H)
                          ; are loaded into register R4.
LDE     R4,#04H[RR2]      ; Identical operation to LDC example, except that
                          ; external program memory is accessed.
```

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

INDEXED ADDRESSING MODE (Concluded)



Sample Instructions:

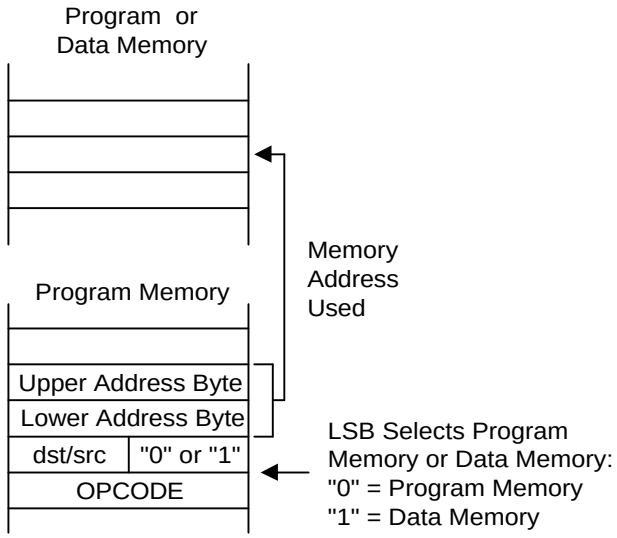
```
LDC     R4, #1000H[RR2]     ; The values in the program address (RR2 +
#1000H)
                            ; are loaded into register R4.
LDE     R4, #1000H[RR2]     ; Identical operation to LDC example, except that
                            ; external program memory is accessed.
```

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.

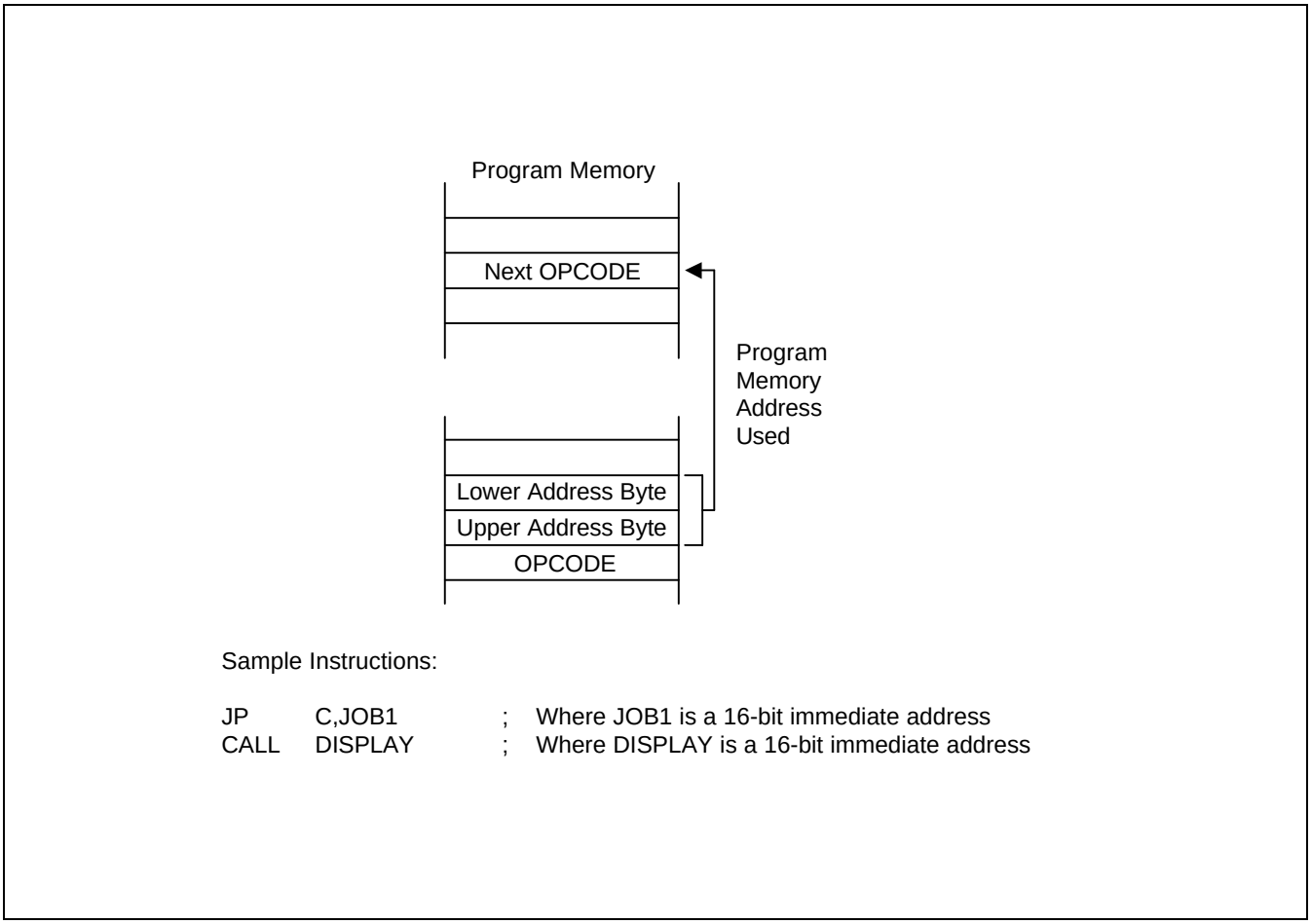


Sample Instructions:

```
LDC     R5,1234H        ;   The values in the program address (1234H)
                            are loaded into register R5.
LDE     R5,1234H        ;   Identical operation to LDC example, except that
                            external program memory is accessed.
```

Figure 3-10. Direct Addressing for Load Instructions

DIRECT ADDRESS MODE (Continued)



Sample Instructions:

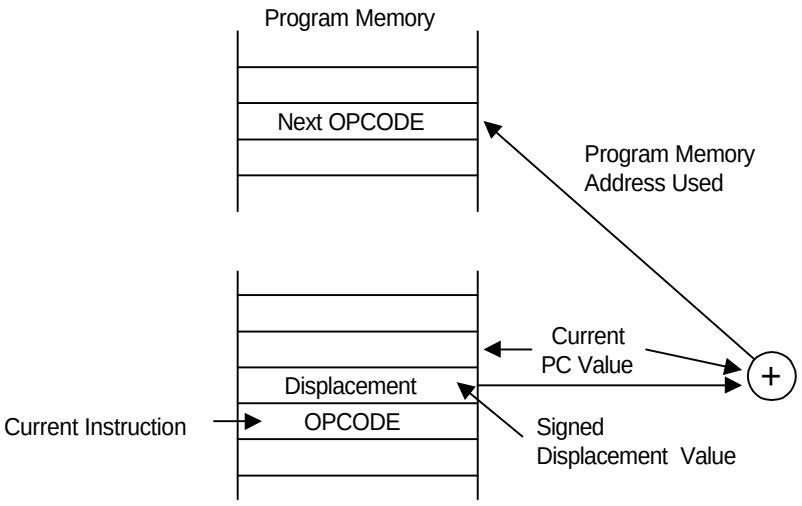
```
JP      C,JOB1      ;   Where JOB1 is a 16-bit immediate address
CALL    DISPLAY     ;   Where DISPLAY is a 16-bit immediate address
```

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.



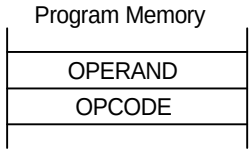
Sample Instructions:

```
JR      ULT,$ + OFFSET      ;   Where OFFSET is a value in the range + 127 to - 128
```

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.



(The Operand value is in the instruction)

Sample Instruction:

```
LD      R0,#0AAH
```

Figure 3-13. Immediate Addressing

4 CONTROL REGISTERS

OVERVIEW

In this section, detailed descriptions of the S3C9228/P9228 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. System and Peripheral Control Registers (Page 0)

Register Name	Mnemonic	Address (Page 0)		R/W
		Decimal	Hex	
Port 0 Control Register	P0CON	235	EBH	R/W
Port 0 Pull-up Resistor Enable Register	P0PUR	236	ECH	R/W
Port 0 Interrupt Control Register	P0INT	237	EDH	R/W
Port 0 Interrupt Edge Selection Register	P0EDGE	238	EEH	R/W
Port 1 Control Register	P1CON	239	EFH	R/W
Port 1 Pull-up Resistor Enable Register	P1PUR	240	F0H	R/W
Port 1 Interrupt Control Register	P1INT	241	F1H	R/W
Port 1 Interrupt Edge Selection Register	P1EDGE	242	F2H	R/W
Port 2 Control Register	P2CON	243	F3H	R/W
Port 2 Pull-up Resistor Enable Register	P2PUR	244	F4H	R/W
Port 3 Control Register	P3CON	245	F5H	R/W
Port 3 Pull-up Resistor Enable Register	P3PUR	246	F6H	R/W
Port 3 Interrupt Control Register	P3INT	247	F7H	R/W
Port 3 Interrupt Edge Selection Register	P3EDGE	248	F8H	R/W
Port 4 Control Register (High Byte)	P4CONH	249	F9H	R/W
Port 4 Control Register (Low Byte)	P4CONL	250	FAH	R/W
Port 5 Control Register (High Byte)	P5CONH	251	FBH	R/W
Port 5 Control Register (Low Byte)	P5CONL	252	FCH	R/W
Port 6 Control Register	P6CON	253	FDH	R/W
LCD Mode Register	LMOD	254	FEH	R/W
Location FFH is not mapped.				

Table 4-1. System and Peripheral Control Registers (Page 0)

Register Name	Mnemonic	Address (Page 0)		R/W
		Decimal	Hex	
Locations D8H-B9H are not mapped.				
Timer B Control Register	TBCON	202	BAH	R/W
Timer 1/A Control Register	TACON	203	BBH	R/W
Timer B Data Register	TBDATA	204	BCH	R/W
Timer A Data Register	TADATA	205	BDH	R/W
Timer B Counter	TBCNT	206	BEH	R
Timer A Counter	TACNT	207	BFH	R
A/D Converter Control Register	ADCON	208	D0H	R/W
A/D Converter Data Register (High Byte)	ADDATAH	209	D1H	R/W
A/D Converter Data Register (Low Byte)	ADDATAL	210	D2H	R/W
Oscillator Control Register	OSCCON	211	D3H	R/W
System Clock Control Register	CLKCON	212	D4H	R/W
System Flags Register	FLAGS	213	D5H	R/W
Interrupt Pending Register 1	INTPND1	214	D6H	R/W
Interrupt Pending Register 2	INTPND2	215	D7H	R/W
LCD Port Control Register	LPOT	216	D8H	R/W
Stack Pointer	SP	217	D9H	R/W
Watch Timer Control Register	WTCON	218	DAH	R/W
Location DBH is not mapped.				
Basic Timer Control Register	BTCON	220	DCH	R/W
Basic Timer Counter	BTCNT	221	DDH	R
Location DEH is not mapped.				
System Mode Register	SYM	223	DFH	R/W
STOP Control Register	STPCON	224	E0H	R/W
SIO Control Register	SIOCON	225	E1H	R/W
SIO Data Register	SIODATA	226	E2H	R/W
SIO Prescaler Register	SIOPS	227	E3H	R/W
Port 0 Data Register	P0	228	E4H	R/W
Port 1 Data Register	P1	229	E5H	R/W
Port 2 Data Register	P2	230	E6H	R/W
Port 3 Data Register	P3	231	E7H	R/W
Port 4 Data Register	P4	232	E8H	R/W
Port 5 Data Register	P5	233	E9H	R/W
Port 6 Data Register	P6	234	EAH	R/W

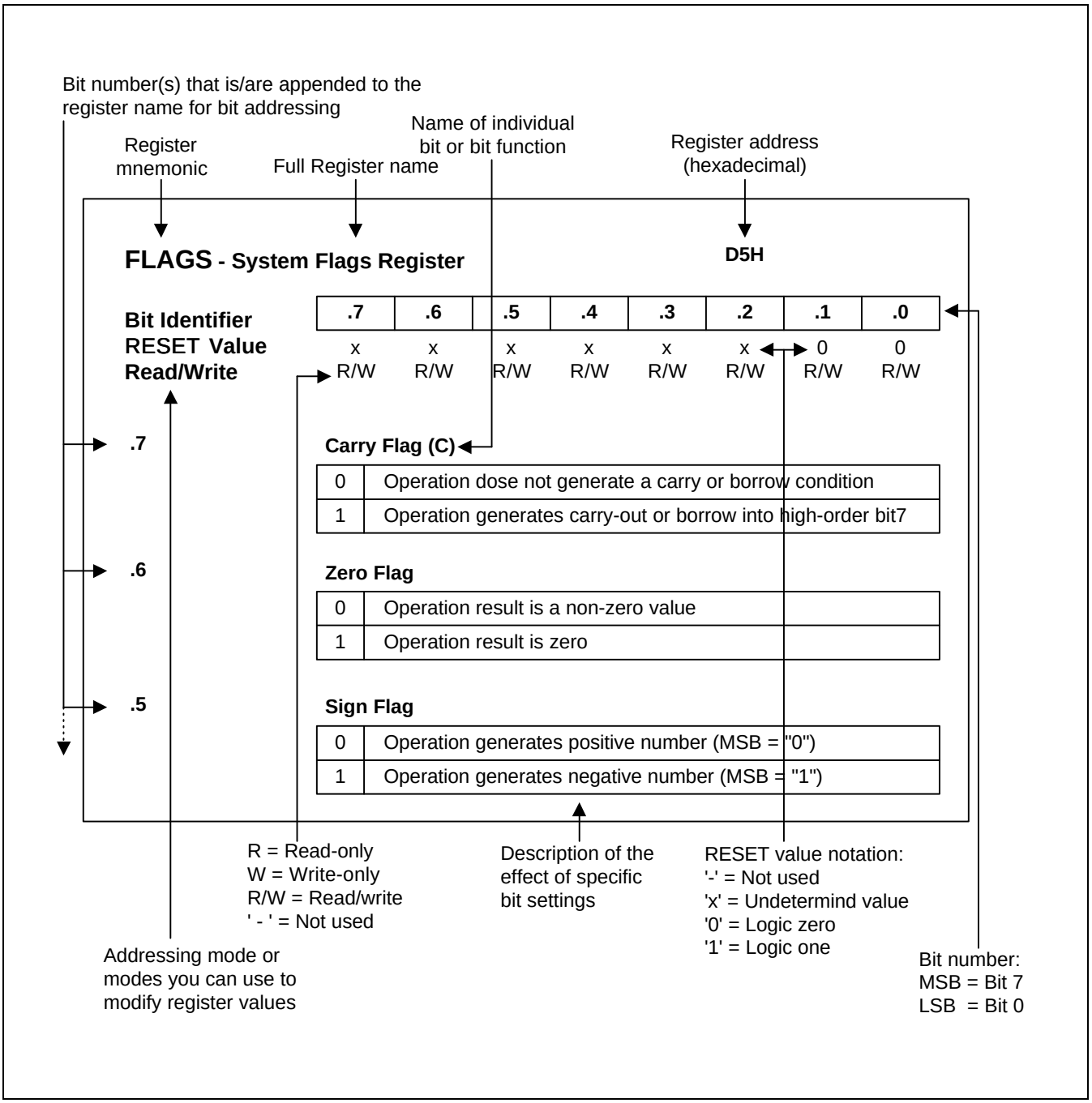


Figure 4-1. Register Description Format

ADCON — A/D Converter Control Register D0H

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	R/W	R/W	R/W

.7-.6 Not used for the S3C9228/P9228

.5-.4 A/D Input Pin Selection Bits

0	0	AD0 (P1.0)
0	1	AD1 (P1.1)
1	0	AD2 (P1.2)
1	1	AD3 (P1.3)

.3 End of Conversion Bit (Read-only)

0	Conversion not complete
1	Conversion complete

.2-.1 Clock Source Selection Bits

0	0	fxx/16
0	1	fxx/8
1	0	fxx/4
1	1	fxx

.0 Start or Enable Bit

0	Disable operation
1	Start operation (automatically disable operation after conversion complete)

BTCON — Basic Timer Control Register 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	fxx/4096
0	1	fxx/1024
1	0	fxx/128
1	1	fxx/16

.1 Basic Timer Counter Clear Bit (1)

0	No effect
1	Clear the basic timer counter value (BTCNT)

.0 Clock Frequency Divider Clear Bit for Basic Timer and Timer/Counters (2)

0	No effect
1	Clear clock frequency dividers

NOTES

1. When "1" is written to BTCON.1, the basic timer counter value is cleared to "00H". Immediately following the write operation, the BTCON.1 value is automatically cleared to "0".
2. When "1" is written to BTCON.0, the corresponding frequency divider is cleared to "00H". Immediately following the write operation, the BTCON.0 value is automatically cleared to "0".

CLKCON — System Clock Control Register D4H

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 Oscillator IRQ Wake-up Function Bit

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

.6-.5 Bits 6-5

0	Always logic zero

.4-.3 CPU Clock (System Clock) Selection Bits

0	0	Divide by 16 (fxx/16)
0	1	Divide by 8 (fxx /8)
1	0	Divide by 2 (fxx /2)
1	1	Non-divided clock (fxx)

.2-.0 Bits 2-0

0	Always logic zero

FLAGS — System Flags Register D5H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	–	–	–	–
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 $\geq$ –128
1	Operation result is $\geq$ +127 or $\leq$ –128

.3-.0 Not used for S3C9228/P9228

INTPND1 — Interrupt Pending Register 1 D6H

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 P1.3's Interrupt Pending Bit

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

.6 P1.2's Interrupt Pending Bit

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

.5 P1.1's Interrupt Pending Bit

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

.4 P1.0's Interrupt Pending Bit

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

.3 P0.3's Interrupt Pending Bit

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

.2 P0.2's Interrupt Pending Bit

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

.1 P0.1's Interrupt Pending Bit

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

.0 P0.0's Interrupt Pending Bit

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

NOTE: Refer to Page 5-6 to clear any pending bits.

INTPND2 — Interrupt Pending Register 2 D7H

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-.6 Not used for S3C9228/P9228

.5 P3.1 (INTP) Interrupt Pending Bit

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

.4 P3.0 (INTP) Interrupt Pending Bit

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

.3 Watch Timer Interrupt Pending Bit

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

.2 SIO Interrupt Pending Bit

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

.1 Timer B Interrupt Pending Bit

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

.0 Timer 1/A Interrupt Pending Bit

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

NOTE: Refer to Page 5-6 to clear any pending bits.

LMOD — LCD Mode Control Register FEH

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 S3C9228/P9228

.6 COM Pins High Impedance Control Bit

0	Normal COMs signal output
1	COM pins are at high impedance

.5 Port3 High Impedance Control Bit

0	Normal I/O
1	High impedance input

.4 LCD Display Control Bit

0	Display off (cut off the LCD voltage dividing resistors)
1	Normal display on

.3-.2 LCD Duty and Bias Selection Bits

0	0	1/3 duty, 1/3 bias; COM0–COM2/SEG0–SEG19
0	1	1/4 duty, 1/3 bias; COM0–COM3/SEG0–SEG19
1	0	1/8 duty, 1/4 bias; COM0–COM7/SEG0–SEG15
1	1	1/8 duty, 1/5 bias; COM0–COM7/SEG0–SEG15

.1-.0 LCD Clock Selection Bits

0	0	$fw/2^7$ (256 Hz when fw is 32.768 kHz)
0	1	$fw/2^6$ (512 Hz when fw is 32.768 kHz)
1	0	$fw/2^5$ (1,024 Hz when fw is 32.768 kHz)
1	1	$fw/2^4$ (2,048 Hz when fw is 32.768 kHz)

LPOT — LCD Port Control Register 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 S3C9228/P9228

.6-.4 SEG4-SEG19 and COM0-COM3 Selection Bit

			SEG4-7	SEG8-11	SEG12-15	SEG16-19/ COM7-COM4	COM0-3
			P4.0-P4.3	P4.4-P4.7	P5.0-P5.3	P5.4-P5.7	P6.0-P6.3
0	0	0	SEG	SEG	SEG	SEG/COM	COM
0	0	1	Port	SEG	SEG	SEG/COM	COM
0	1	0	Port	Port	SEG	SEG/COM	COM
0	1	1	Port	Port	Port	SEG/COM	COM
1	0	0	Port	Port	Port	Port	COM
1	0	1	Port	Port	Port	Port	Port

.3 SEG3/P3.0 Selection Bit

0	SEG port
1	Normal I/O port

.2 SEG2/P3.1 Selection Bit

0	SEG port
1	Normal I/O port

.1 SEG1/P2.0 Selection Bit

0	SEG port
1	Normal I/O port

.0 SEG0/P2.1 Selection Bit

0	SEG port
1	Normal I/O port

NOTE: SEG16-SEG19 are shared with COM4-COM7.

OSCCON — Oscillator Control Register D3H

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

.7-.4 Not used for S3C9228/P9228

.3 Main Oscillator Control Bit

0	Main oscillator RUN
1	Main oscillator STOP

.2 Sub Oscillator Control Bit

0	Sub oscillator RUN
1	Sub oscillator STOP

.1 Not used for S3C9228/P9228

.0 System Clock Selection Bit

0	Select main oscillator for system clock
1	Select sub oscillator for system clock

P0CON – Port 0 Control Register EBH

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 P0.3/BUZ/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (BUZ output)

.5-.4 P0.2/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.3-.2 P0.1/T1CLK/INT Configuration Bits

0	0	Schmitt trigger input (T1CLK input)
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.1-.0 P0.0/TAOUT/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (TAOUT output)

P0INT –Port 0 Interrupt Enable Register EDH

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 S3C9228/P9228

.3 P0.3's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.2 P0.2's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.1 P0.1's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.0 P0.0's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

P0PUR –Port 0 Pull-up Resistors Enable Register ECH

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 S3C9228/P9228

.3 P0.3's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.2 P0.2's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.1 P0.1's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.0 P0.0's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

P0EDGE –Port 0 Interrupt Edge Selection Register EEH

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 S3C9228/P9228

.3 P0.3's Interrupt Edge Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.2 P0.2's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.1 P0.1's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.0 P0.0's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

P1CON – Port 1 Control Register 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-.6 P1.3/AD3/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (ADC mode)

.5-.4 P1.2/AD2/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (ADC mode)

.3-.2 P1.1/AD1/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (ADC mode)

.1-.0 P1.0/AD0/INT Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (ADC mode)

P1INT –Port 1 Interrupt Enable Register F1H

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 S3C9228/P9228

.3 **P1.3's Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.2 **P1.2's Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.1 **P1.1's Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.0 **P1.0's Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

P1PUR –Port 1 Pull-up Resistors Enable Register F0H

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 S3C9228/P9228

.3 P1.3's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.2 P1.2's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.1 P1.1's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.0 P1.0's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

P1EDGE –Port 1 Interrupt Edge Selection Register F2H

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 S3C9228/P9228

.3 P1.3's Interrupt Edge Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.2 P1.2's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.1 P1.1's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.0 P1.0's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

P2CON – Port 2 Control Register F3H

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 P2.3 Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.5-.4 P2.2/SI Configuration Bits

0	0	Schmitt trigger input (SI)
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.3-.2 P2.1/SO Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (SO out)

.1-.0 P2.0/SCK Configuration Bits

0	0	Schmitt trigger input (SCK in)
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Alternative function (SCK out)

P2PUR –Port 2 Pull-up Resistors Enable Register F4H

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 S3C9228/P9228

.3 P2.3's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.2 P2.2's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.1 P2.1's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

.0 P2.0's Pull-up Resistor Enable Bit

0	Disable pull-up resistor
1	Enable pull-up resistor

P3CON – Port 3 Control Register F5H

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 S3C9228/P9228

.3-.2 P3.1/SEG2/INTP Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

.1-.0 P3.0/SEG3/INTP Configuration Bits

0	0	Schmitt trigger input
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Not available

P3INT –Port 3 Interrupt Enable Register F7H

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

.7-.2

Not used for S3C9228/P9228

.1 P3.1's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.0 P3.0's Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

P3PUR –Port 3 Pull-up Resistors Enable Register F6H

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

.7-.2

Not used for S3C9228/P9228

.1 **P3.1's Pull-up Resistor Enable Bit**

0	Disable pull-up resistor
1	Enable pull-up resistor

.0 **P3.0's Pull-up Resistor Enable Bit**

0	Disable pull-up resistor
1	Enable pull-up resistor

P3EDGE –Port 3 Interrupt Edge Selection Register F8H

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

.7-.4

Not used for S3C9228/P9228

.1 P3.1's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

.0 P3.0's Interrupt State Setting Bit

0	Falling edge interrupt
1	Rising edge interrupt

P4CONH – Port 4 Control Register High Byte 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

.7-.6 P4.7/SEG11 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P4.6/SEG10 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P4.5/SEG9 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P4.4/SEG8 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

P4CONL–Port 4 Control Register Low Byte FAH

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 P4.3/SEG7 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P4.2/SEG6 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P4.1/SEG5 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P4.0/SEG4 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

P5CONH – Port 5 Control Register High Byte FBH

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 P5.7/SEG19/COM4 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P5.6/SEG18/COM5 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P5.5/SEG17/COM6 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P5.4/SEG16/COM7 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

P5CONL – Port 5 Control Register Low Byte FCH

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 P5.3/SEG15 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P5.2/SEG14 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P5.1/SEG13 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P5.0/SEG12 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

P6CON – Port 6 Control Register High Byte FDH

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 P6.3/COM0 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.5-.4 P6.2/COM1 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.3-.2 P6.1/COM2 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

.1-.0 P6.0/COM3 Configuration Bits

0	0	Input mode
0	1	Push-pull output
1	0	N-channel open-drain output
1	1	Input, pull-up mode

SIOCON — SIO Control Register E1H

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 SIO Shift Clock Selection Bit

0	Internal clock (P.S clock)
1	External clock (SCK)

.6 Data Direction Control Bit

0	MSB-first mode
1	LSB-first mode

.5 SIO Mode Selection Bit

0	Receive-only mode
1	Transmit/receive mode

.4 Shift Clock Edge Selection Bit

0	Tx at falling edges, Rx at rising edges
1	Tx at rising edges, Rx at falling edges

.3 SIO Counter Clear and Shift Start Bit

0	No action
1	Clear 3-bit counter and start shifting

.2 SIO Shift Operation Enable Bit

0	Disable shifter and clock counter
1	Enable shifter and clock counter

.1 SIO Interrupt Enable Bit

0	Disable SIO interrupt
1	Enable SIO interrupt

.0

Not used for S3C9228/P9228

STPCON – Stop Control Register E0H

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 Stop Control Bits

1	0	1	0	0	1	0	1	Enable Stop instruction
Other values								Disable Stop instruction

NOTE: Before executing the STOP instruction, the STPCON register must be set to "10100101B". Otherwise the STOP instruction will not execute.

SYM — System Mode Register 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 S3C9228/P9228

.3 Global Interrupt Enable Bit

0	Global interrupt processing disable (DI instruction)
1	Global interrupt processing enable (EI instruction)

.2-.0 Page Selection Bits

0	0	0	Page 0
0	0	1	Page 1
Other values			Not available

TACON — Timer 1/A Control Register BBH

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 Timer 1 Mode Selection Bit

0	Two 8-bit timers mode (Timer A/B)
1	One 16-bit timer mode (Timer 1)

.6-.4 Timer 1/A Clock Selection Bits

0	0	0	fxx/512
0	0	1	fxx/256
0	1	0	fxx/64
0	1	1	fxx/8
1	0	0	fxx (system clock)
1	0	1	fxt (sub clock)
1	1	0	T1CLK (external clock)
1	1	1	Not available

.3 Timer 1/A Counter Clear Bit

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

.2 Timer 1/A Counter Enable Bit

0	Disable counting operation
1	Enable counting operation

.1 Timer 1/A Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.0 Bit 0

Not used for S3C9228/P9228

TBCON — Timer B Control Register BAH

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 Not used for S3C9228/P9228

.6-.4 Timer B Clock Selection Bits

0	0	0	fxx/512
0	0	1	fxx/256
0	1	0	fxx/64
0	1	1	fxx/8
1	0	0	fxx (system clock)
1	0	1	fxt (sub clock)

.3 Timer B Counter Clear Bit

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

.2 Timer B Counter Enable Bit

0	Disable counting operation
1	Enable counting operation

.1 Timer B Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

.0 Not used for S3C9228/P9228

WTCON — Watch Timer Control Register DAH

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 Watch Timer Clock Selection Bit

0	Select main clock divided by 2^7 (fx/128)
1	Select sub clock (fxt)

.6 Watch Timer Interrupt Enable Bit

0	Disable watch timer interrupt
1	Enable watch timer interrupt

.5-.4 Buzzer Signal Selection Bits

0	0	0.5 kHz
0	1	1 kHz
1	0	2 kHz
1	1	4 kHz

.3-.2 Watch Timer Speed Selection Bits

0	0	Set watch timer interrupt to 1s
0	1	Set watch timer interrupt to 0.5s
1	0	Set watch timer interrupt to 0.25s
1	1	Set watch timer interrupt to 3.91ms

.1 Watch Timer Enable Bit

0	Disable watch timer; Clear frequency dividing circuits
1	Enable watch timer

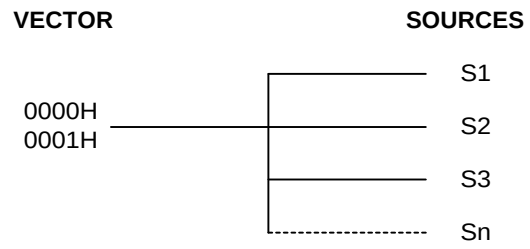
.0

Not used for S3C9228/P9228

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.



NOTES:
1. The SAM88RCRI interrupt has only one vector address (0000H-0001H).
2. The number of Sn value is expandable.

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 SAM87RCRI, the order of service is determined by a sequence of source which is executed in interrupt service routine.

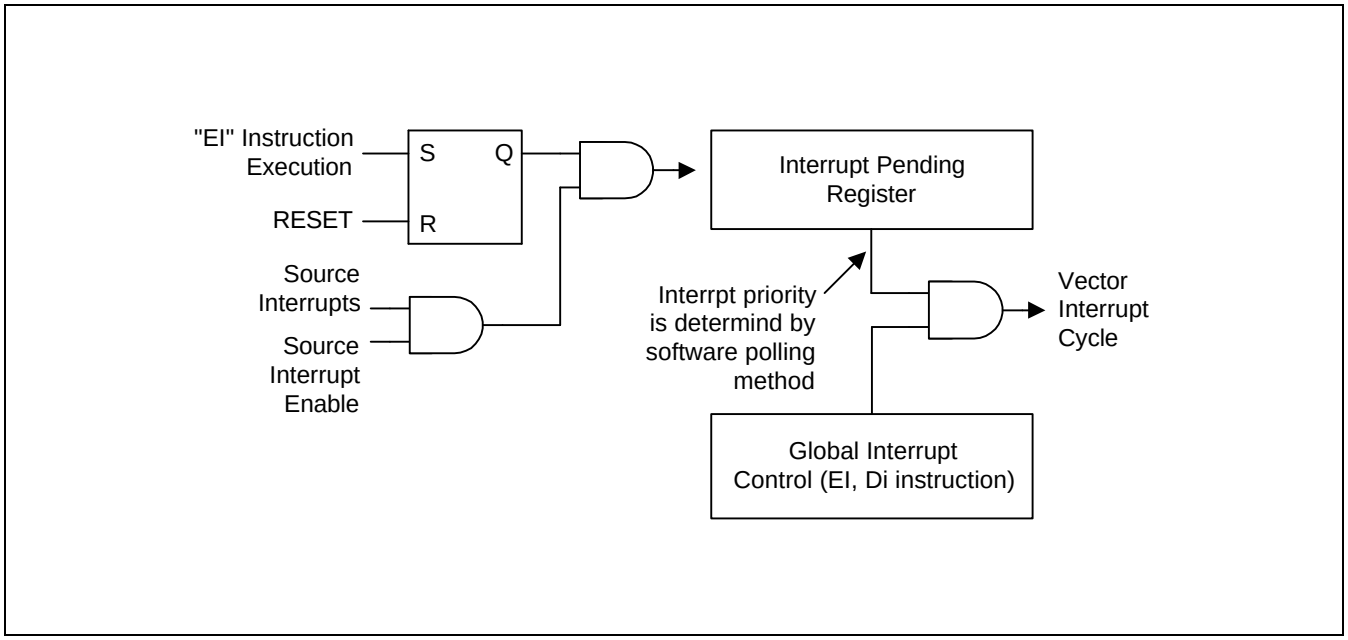


Figure 5-2. Interrupt Function Diagram

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.

S3C9228/P9228 INTERRUPT STRUCTURE

The S3C9228/P9228 microcontroller has fourteen peripheral interrupt sources:

— Timer 1/A interrupt
— Timer B interrupt
— SIO interrupt
— Watch Timer interrupt
— Four external interrupts for port 0
— Four external interrupts for port 1
— Two external interrupts for port 3

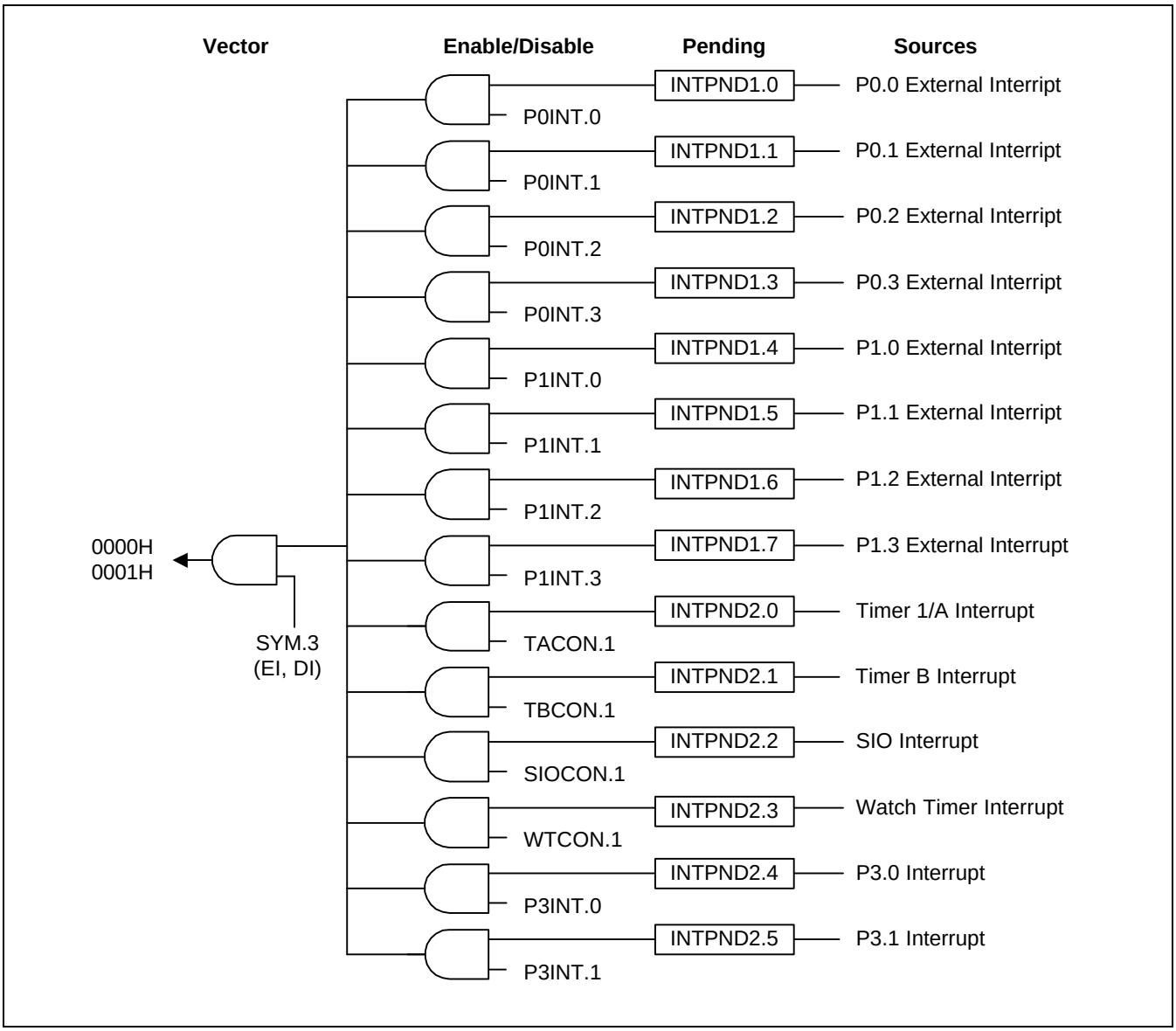


Figure 5-3. S3C9228/P9228 Interrupt Structure

Programming Tip — How to clear an interrupt pending bit

As the following examples are shown, a load instruction should be used to clear an interrupt pending bit.

Examples:

```
1.    LD      INTPND1, #11111011B     ; Clear P0.2's interrupt pending bit
      •
      •
      •
      IRET

2.    LD      INTPND2, #11110111B     ; Clear watch timer interrupt pending bit
      •
      •
      •
      IRET
```

6 SAM88RCRI INSTRUCTION SET

OVERVIEW

The SAM88RCRI instruction set is designed to support the large register file. It includes a full complement of 8-bit arithmetic and logic operations. There are 41 instructions. No special I/O instructions are necessary because I/O control and data registers are mapped directly into the register file. Flexible instructions for bit addressing, rotate, and shift operations complete the powerful data manipulation capabilities of the SAM88RCRI instruction set.

REGISTER ADDRESSING

To access an individual register, an 8-bit address in the range 0-255 or the 4-bit address of a working register is specified. Paired registers can be used to construct 16-bit program memory or data memory addresses. For detailed information about register addressing, please refer to Section 2, "Address Spaces".

ADDRESSING MODES

There are six addressing modes: Register (R), Indirect Register (IR), Indexed (X), Direct (DA), Relative (RA), and Immediate (IM). For detailed descriptions of these addressing modes, please refer to Section 3, "Addressing Modes".

Table 6-1. Instruction Group Summary

Mnemonic	Operands	Instruction
Load Instructions		
CLR	dst	Clear
LD	dst,src	Load
LDC	dst,src	Load program memory
LDE	dst,src	Load external data memory
LDCD	dst,src	Load program memory and decrement
LDED	dst,src	Load external data memory and decrement
LDCI	dst,src	Load program memory and increment
LDEI	dst,src	Load external data memory and increment
POP	dst	Pop from stack
PUSH	src	Push to stack
Arithmetic Instructions		
ADC	dst,src	Add with carry
ADD	dst,src	Add
CP	dst,src	Compare
DEC	dst	Decrement
INC	dst	Increment
SBC	dst,src	Subtract with carry
SUB	dst,src	Subtract
Logic Instructions		
AND	dst,src	Logical AND
COM	dst	Complement
OR	dst,src	Logical OR
XOR	dst,src	Logical exclusive OR

Table 6-1. Instruction Group Summary (Continued)

Mnemonic	Operands	Instruction
Program Control Instructions		
CALL	dst	Call procedure
IRET		Interrupt return
JP	cc,dst	Jump on condition code
JP	dst	Jump unconditional
JR	cc,dst	Jump relative on condition code
RET		Return
Bit Manipulation Instructions		
TCM	dst,src	Test complement under mask
TM	dst,src	Test under mask
Rotate and Shift Instructions		
RL	dst	Rotate left
RLC	dst	Rotate left through carry
RR	dst	Rotate right
RRC	dst	Rotate right through carry
SRA	dst	Shift right arithmetic
CPU Control Instructions		
CCF		Complement carry flag
DI		Disable interrupts
EI		Enable interrupts
IDLE		Enter Idle mode
NOP		No operation
RCF		Reset carry flag
SCF		Set carry flag
STOP		Enter Stop mode

FLAGS REGISTER (FLAGS)

The FLAGS register contains eight bits that describe the current status of CPU operations. Four of these bits, FLAGS.4 – FLAGS.7, can be tested and used with conditional jump instructions;

FLAGS register can be set or reset by instructions as long as its outcome does not affect the flags, such as, Load instruction. Logical and Arithmetic instructions such as, AND, OR, XOR, ADD, and SUB can affect the Flags register. For example, the AND instruction updates the Zero, Sign and Overflow flags based on the outcome of the AND instruction. If the AND instruction uses the Flags register as the destination, then simultaneously, two write will occur to the Flags register producing an unpredictable result.

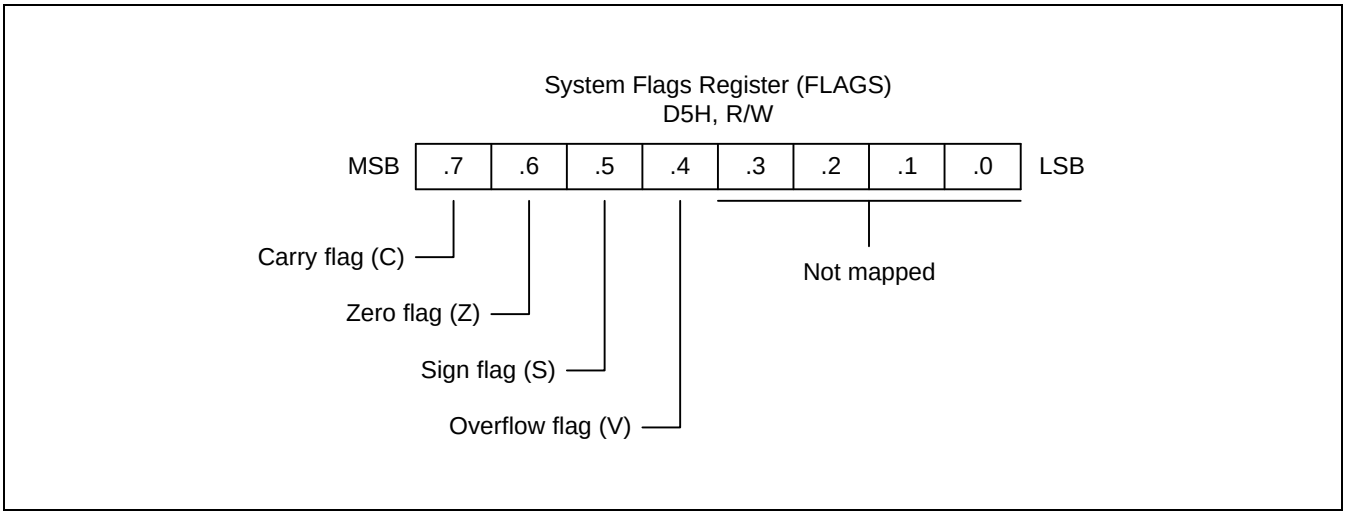


Figure 6-1. System Flags Register (FLAGS)

FLAG DESCRIPTIONS

Overflow Flag (FLAGS.4, V)

The V flag is set to "1" when the result of a two's-complement operation is greater than + 127 or less than – 128. It is also cleared to "0" following logic operations.

Sign Flag (FLAGS.5, S)

Following arithmetic, logic, rotate, or shift operations, the sign bit identifies the state of the MSB of the result. A logic zero indicates a positive number and a logic one indicates a negative number.

Zero Flag (FLAGS.6, Z)

For arithmetic and logic operations, the Z flag is set to "1" if the result of the operation is zero. For operations that test register bits, and for shift and rotate operations, the Z flag is set to "1" if the result is logic zero.

Carry Flag (FLAGS.7, C)

The C flag is set to "1" if the result from an arithmetic operation generates a carry-out from or a borrow to the bit 7 position (MSB). After rotate and shift operations, it contains the last value shifted out of the specified register. Program instructions can set, clear, or complement the carry flag.

INSTRUCTION SET NOTATION

Table 6-2. Flag Notation Conventions

Flag	Description
C	Carry flag
Z	Zero flag
S	Sign flag
V	Overflow flag
0	Cleared to logic zero
1	Set to logic one
*	Set or cleared according to operation
–	Value is unaffected
x	Value is undefined

Table 6-3. Instruction Set Symbols

Symbol	Description
dst	Destination operand
src	Source operand
@	Indirect register address prefix
PC	Program counter
FLAGS	Flags register (D5H)
#	Immediate operand or register address prefix
H	Hexadecimal number suffix
D	Decimal number suffix
B	Binary number suffix
opc	Opcode

Table 6-4. Instruction Notation Conventions

Notation	Description	Actual Operand Range
cc	Condition code	See list of condition codes in Table 6-6.
r	Working register only	Rn (n = 0–15)
rr	Working register pair	RRp (p = 0, 2, 4, ..., 14)
R	Register or working register	reg or Rn (reg = 0–255, n = 0–15)
RR	Register pair or working register pair	reg or RRp (reg = 0–254, even number only, where p = 0, 2, ..., 14)
Ir	Indirect working register only	@Rn (n = 0–15)
IR	Indirect register or indirect working register	@Rn or @reg (reg = 0–255, n = 0–15)
Irr	Indirect working register pair only	@RRp (p = 0, 2, ..., 14)
IRR	Indirect register pair or indirect working register pair	@RRp or @reg (reg = 0–254, even only, where p = 0, 2, ..., 14)
X	Indexed addressing mode	#reg[Rn] (reg = 0–255, n = 0–15)
XS	Indexed (short offset) addressing mode	#addr[RRp] (addr = range –128 to +127, where p = 0, 2, ..., 14)
xl	Indexed (long offset) addressing mode	#addr [RRp] (addr = range 0–8191, where p = 0, 2, ..., 14)
da	Direct addressing mode	addr (addr = range 0–8191)
ra	Relative addressing mode	addr (addr = number in the range +127 to –128 that is an offset relative to the address of the next instruction)
im	Immediate addressing mode	#data (data = 0–255)

Table 6-5. Opcode Quick Reference

OPCODE MAP									
	LOWER NIBBLE (HEX)								
	–	0	1	2	3	4	5	6	7
U	0	DEC R1	DEC IR1	ADD r1,r2	ADD r1,Ir2	ADD R2,R1	ADD IR2,R1	ADD R1,IM	
P	1	RLC R1	RLC IR1	ADC r1,r2	ADC r1,Ir2	ADC R2,R1	ADC IR2,R1	ADC R1,IM	
P	2	INC R1	INC IR1	SUB r1,r2	SUB r1,Ir2	SUB R2,R1	SUB IR2,R1	SUB R1,IM	
E	3	JP IRR1		SBC r1,r2	SBC r1,Ir2	SBC R2,R1	SBC IR2,R1	SBC R1,IM	
R	4			OR r1,r2	OR r1,Ir2	OR R2,R1	OR IR2,R1	OR R1,IM	
	5	POP R1	POP IR1	AND r1,r2	AND r1,Ir2	AND R2,R1	AND IR2,R1	AND R1,IM	
N	6	COM R1	COM IR1	TCM r1,r2	TCM r1,Ir2	TCM R2,R1	TCM IR2,R1	TCM R1,IM	
I	7	PUSH R2	PUSH IR2	TM r1,r2	TM r1,Ir2	TM R2,R1	TM IR2,R1	TM R1,IM	
B	8								LD r1, x, r2
B	9	RL R1	RL IR1						LD r2, x, r1
L	A			CP r1,r2	CP r1,Ir2	CP R2,R1	CP IR2,R1	CP R1,IM	LDC r1, Irr2, xL
E	B	CLR R1	CLR IR1	XOR r1,r2	XOR r1,Ir2	XOR R2,R1	XOR IR2,R1	XOR R1,IM	LDC r2, Irr2, xL
	C	RRC R1	RRC IR1		LDC r1,Irr2				LD r1, Ir2
H	D	SRA R1	SRA IR1		LDC r2,Irr1			LD IR1,IM	LD Ir1, r2
E	E	RR R1	RR IR1	LDCD r1,Irr2	LDCI r1,Irr2	LD R2,R1	LD R2,IR1	LD R1,IM	LDC r1, Irr2, xs
X	F					CALL IRR1	LD IR2,R1	CALL DA1	LDC r2, Irr1, xs

Table 6-5. Opcode Quick Reference (Continued)

OPCODE MAP									
	LOWER NIBBLE (HEX)								
	–	8	9	A	B	C	D	E	F
U	0	LD r1,R2	LD r2,R1		JR cc,RA	LD r1,IM	JP cc,DA	INC r1	
P	1	↓	↓		↓	↓	↓	↓	
P	2								
E	3								
R	4								
	5								
N	6								IDLE
I	7	↓	↓		↓	↓	↓	↓	STOP
B	8								DI
B	9								EI
L	A								RET
E	B								IRET
	C								RCF
H	D	↓	↓		↓	↓	↓	↓	SCF
E	E								CCF
X	F	LD r1,R2	LD r2,R1		JR cc,RA	LD r1,IM	JP cc,DA	INC r1	NOP

CONDITION CODES

The opcode of a conditional jump always contains a 4-bit field called the condition code (cc). This specifies under which conditions it is to execute the jump. For example, a conditional jump with the condition code for "equal" after a compare operation only jumps if the two operands are equal. Condition codes are listed in Table 6-6.

The carry (C), zero (Z), sign (S), and overflow (V) flags are used to control the operation of conditional jump instructions.

Table 6-6. Condition Codes

Binary	Mnemonic	Description	Flags Set
0000	F	Always false	–
1000	T	Always true	–
0111 (1)	C	Carry	C = 1
1111 (1)	NC	No carry	C = 0
0110 (1)	Z	Zero	Z = 1
1110 (1)	NZ	Not zero	Z = 0
1101	PL	Plus	S = 0
0101	MI	Minus	S = 1
0100	OV	Overflow	V = 1
1100	NOV	No overflow	V = 0
0110 (1)	EQ	Equal	Z = 1
1110 (1)	NE	Not equal	Z = 0
1001	GE	Greater than or equal	(S XOR V) = 0
0001	LT	Less than	(S XOR V) = 1
1010	GT	Greater than	(Z OR (S XOR V)) = 0
0010	LE	Less than or equal	(Z OR (S XOR V)) = 1
1111 (1)	UGE	Unsigned greater than or equal	C = 0
0111 (1)	ULT	Unsigned less than	C = 1
1011	UGT	Unsigned greater than	(C = 0 AND Z = 0) = 1
0011	ULE	Unsigned less than or equal	(C OR Z) = 1

NOTES:

1. Indicate condition codes that are related to two different mnemonics but which test the same flag. For example, Z and EQ are both true if the zero flag (Z) is set, but after an ADD instruction, Z would probably be used; after a CP instruction, however, EQ would probably be used.
2. For operations involving unsigned numbers, the special condition codes UGE, ULT, UGT, and ULE must be used.

INSTRUCTION DESCRIPTIONS

This section contains detailed information and programming examples for each instruction in the SAM88RCRI instruction set. Information is arranged in a consistent format for improved readability and for fast referencing. The following information is included in each instruction description:

— Instruction name (mnemonic)
— Full instruction name
— Source/destination format of the instruction operand
— Shorthand notation of the instruction's operation
— Textual description of the instruction's effect
— Specific flag settings affected by the instruction
— Detailed description of the instruction's format, execution time, and addressing mode(s)
— Programming example(s) explaining how to use the instruction

ADC — Add With Carry

ADC dst,src

Operation: dst ¨ dst + src + c

The source operand, along with the setting of the carry flag, is added to the destination operand and the sum is stored in the destination. The contents of the source are unaffected. Two's-complement addition is performed. In multiple precision arithmetic, this instruction permits the carry from the addition of low-order operands to be carried into the addition of high-order operands.

Flags:

C: Set if there is a carry from the most significant bit of the result; cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurs, that is, if both operands are of the same sign and the result is of the opposite sign; cleared otherwise.
D: Always cleared to "0".
H: Set if there is a carry from the most significant bit of the low-order four bits of the result; cleared otherwise.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	dst | src		2	4	12	r	r
				6	13	r	lr
opc	src	dst	3	6	14	R	R
				6	15	R	IR
opc	dst	src	3	6	16	R	IM

Examples: Given: R1 = 10H, R2 = 03H, C flag = "1", register 01H = 20H, register 02H = 03H, and register 03H = 0AH:

ADC	R1,R2	→	R1 = 14H, R2 = 03H
ADC	R1,@R2	→	R1 = 1BH, R2 = 03H
ADC	01H,02H	→	Register 01H = 24H, register 02H = 03H
ADC	01H,@02H	→	Register 01H = 2BH, register 02H = 03H
ADC	01H,#11H	→	Register 01H = 32H

In the first example, destination register R1 contains the value 10H, the carry flag is set to "1", and the source working register R2 contains the value 03H. The statement "ADC R1,R2" adds 03H and the carry flag value ("1") to the destination value 10H, leaving 14H in register R1.

ADD — Add

ADD dst,src

Operation: dst ¨ dst + src

The source operand is added to the destination operand and the sum is stored in the destination. The contents of the source are unaffected. Two's-complement addition is performed.

Flags:
C: Set if there is a carry from the most significant bit of the result; cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if both operands are of the same sign and the result is of the opposite sign; cleared otherwise.
D: Always cleared to "0".
H: Set if a carry from the low-order nibble occurred.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc | dst | src	2	4	02	r	r
		6	03	r	lr
opc | src | dst	3	6	04	R	R
		6	05	R	IR
opc | dst | src	3	6	06	R	IM

Examples: Given: R1 = 12H, R2 = 03H, register 01H = 21H, register 02H = 03H, register 03H = 0AH:

ADD	R1,R2	→	R1 = 15H, R2 = 03H
ADD	R1,@R2	→	R1 = 1CH, R2 = 03H
ADD	01H,02H	→	Register 01H = 24H, register 02H = 03H
ADD	01H,@02H	→	Register 01H = 2BH, register 02H = 03H
ADD	01H,#25H	→	Register 01H = 46H

In the first example, destination working register R1 contains 12H and the source working register R2 contains 03H. The statement "ADD R1,R2" adds 03H to 12H, leaving the value 15H in register R1.

AND — Logical AND

AND dst,src

Operation: dst ¨ dst AND src

The source operand is logically ANDed with the destination operand. The result is stored in the destination. The AND operation results in a "1" bit being stored whenever the corresponding bits in the two operands are both logic ones; otherwise a "0" bit value is stored. The contents of the source are unaffected.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always cleared to "0".
D: Unaffected.
H: Unaffected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc	dst | src		2	4	52	r	r
				6	53	r	lr
opc	src	dst	3	6	54	R	R
				6	55	R	IR
opc	dst	src	3	6	56	R	IM

Examples: Given: R1 = 12H, R2 = 03H, register 01H = 21H, register 02H = 03H, register 03H = 0AH:

AND	R1,R2	→	R1 = 02H, R2 = 03H
AND	R1,@R2	→	R1 = 02H, R2 = 03H
AND	01H,02H	→	Register 01H = 01H, register 02H = 03H
AND	01H,@02H	→	Register 01H = 00H, register 02H = 03H
AND	01H,#25H	→	Register 01H = 21H

In the first example, destination working register R1 contains the value 12H and the source working register R2 contains 03H. The statement "AND R1,R2" logically ANDs the source operand 03H with the destination operand value 12H, leaving the value 02H in register R1.

CALL — Call Procedure

CALL dst

Operation:

SP ¨ SP – 1
@SP ¨ PCL
SP ¨ SP –1
@SP ¨ PCH
PC ¨ dst

The current contents of the program counter are pushed onto the top of the stack. The program counter value used is the address of the first instruction following the CALL instruction. The specified destination address is then loaded into the program counter and points to the first instruction of a procedure. At the end of the procedure the return instruction (RET) can be used to return to the original program flow. RET pops the top of the stack back into the program counter.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	3	14	F6	DA
opc dst	2	12	F4	IRR

Examples: Given: R0 = 15H, R1 = 21H, PC = 1A47H, and SP = 0B2H:

CALL 1521H → SP = 0B0H
(Memory locations 00H = 1AH, 01H = 4AH, where 4AH is the address that follows the instruction.)

CALL @RR0 → SP = 0B0H (00H = 1AH, 01H = 49H)

In the first example, if the program counter value is 1A47H and the stack pointer contains the value 0B2H, the statement "CALL 1521H" pushes the current PC value onto the top of the stack. The stack pointer now points to memory location 00H. The PC is then loaded with the value 1521H, the address of the first instruction in the program sequence to be executed.

If the contents of the program counter and stack pointer are the same as in the first example, the statement "CALL @RR0" produces the same result except that the 49H is stored in stack location 01H (because the two-byte instruction format was used). The PC is then loaded with the value 1521H, the address of the first instruction in the program sequence to be executed.

CCF — Complement Carry Flag

CCF

Operation: C ¨ NOT C

The carry flag (C) is complemented. If C = "1", the value of the carry flag is changed to logic zero; if C = "0", the value of the carry flag is changed to logic one.

Flags: **C:** Complemented.

No other flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	EF

Example: Given: The carry flag = "0":

CCF

If the carry flag = "0", the CCF instruction complements it in the FLAGS register (0D5H), changing its value from logic zero to logic one.

CLR — Clear

CLR dst

Operation: dst ¨ "0"

The destination location is cleared to "0".

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc | dst	2	4	B0	R
		4	B1	IR

Examples: Given: Register 00H = 4FH, register 01H = 02H, and register 02H = 5EH:

```
CLR   00H    →   Register 00H = 00H
CLR   @01H   →   Register 01H = 02H, register 02H = 00H
```

In Register (R) addressing mode, the statement "CLR 00H" clears the destination register 00H value to 00H. In the second example, the statement "CLR @01H" uses Indirect Register (IR) addressing mode to clear the 02H register value to 00H.

COM — Complement

COM dst

Operation: dst ¨ NOT dst

The contents of the destination location are complemented (one's complement); all "1s" are changed to "0s", and vice-versa.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always reset to "0".
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc | dst	2	4	60	R
		4	61	IR

Examples: Given: R1 = 07H and register 07H = 0F1H:

```
COM     R1      →     R1 = 0F8H
COM     @R1     →     R1 = 07H, register 07H = 0EH
```

In the first example, destination working register R1 contains the value 07H (00000111B). The statement "COM R1" complements all the bits in R1: all logic ones are changed to logic zeros, and vice-versa, leaving the value 0F8H (11111000B).

In the second example, Indirect Register (IR) addressing mode is used to complement the value of destination register 07H (11110001B), leaving the new value 0EH (00001110B).

CP — Compare

CP dst,src

Operation: dst – src

The source operand is compared to (subtracted from) the destination operand, and the appropriate flags are set accordingly. The contents of both operands are unaffected by the comparison.

Flags:
C: Set if a "borrow" occurred (src > dst); cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the operands were of opposite signs and the sign of the result is of the same as the sign of the source operand; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	dst | src		2	4	A2	r	r
				6	A3	r	lr
opc	src	dst	3	6	A4	R	R
				6	A5	R	IR
opc	dst	src	3	6	A6	R	IM

Examples: 1. Given: R1 = 02H and R2 = 03H:

```
CP      R1,R2      →      Set the C and S flags
```

Destination working register R1 contains the value 02H and source register R2 contains the value 03H. The statement "CP R1,R2" subtracts the R2 value (source/subtrahend) from the R1 value (destination/minuend). Because a "borrow" occurs and the difference is negative, C and S are "1".

2. Given: R1 = 05H and R2 = 0AH:

```
        CP      R1,R2
        JP      UGE,SKIP
        INC     R1
SKIP    LD      R3,R1
```

In this example, destination working register R1 contains the value 05H which is less than the contents of the source working register R2 (0AH). The statement "CP R1,R2" generates C = "1" and the JP instruction does not jump to the SKIP location. After the statement "LD R3,R1" executes, the value 06H remains in working register R3.

DEC — Decrement

DEC dst

Operation: dst ¨ dst – 1

The contents of the destination operand are decremented by one.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, dst value is –128(80H) and result value is +127(7FH); cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	00	R
		4	01	IR

Examples: Given: R1 = 03H and register 03H = 10H:

```
DEC   R1    →   R1 = 02H
DEC   @R1   →   Register 03H = 0FH
```

In the first example, if working register R1 contains the value 03H, the statement "DEC R1" decrements the hexadecimal value by one, leaving the value 02H. In the second example, the statement "DEC @R1" decrements the value 10H contained in the destination register 03H by one, leaving the value 0FH.

DI — Disable Interrupts

DI

Operation: SYM (2) ¨ 0

Bit zero of the system mode register, SYM.2, is cleared to "0", globally disabling all interrupt processing. Interrupt requests will continue to set their respective interrupt pending bits, but the CPU will not service them while interrupt processing is disabled.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	8F

Example: Given: SYM = 04H:

DI

If the value of the SYM register is 04H, the statement "DI" leaves the new value 00H in the register and clears SYM.2 to "0", disabling interrupt processing.

EI — Enable Interrupts

EI

Operation: SYM (2) ¨ 1

An EI instruction sets bit 2 of the system mode register, SYM.2 to "1". This allows interrupts to be serviced as they occur. If an interrupt's pending bit was set while interrupt processing was disabled (by executing a DI instruction), it will be serviced when you execute the EI instruction.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	9F

Example: Given: SYM = 00H:

EI

If the SYM register contains the value 00H, that is, if interrupts are currently disabled, the statement "EI" sets the SYM register to 04H, enabling all interrupts (SYM.2 is the enable bit for global interrupt processing).

IDLE — Idle Operation

IDLE

Operation:

The IDLE instruction stops the CPU clock while allowing system clock oscillation to continue. Idle mode can be released by an interrupt request (IRQ) or an external reset operation.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	1	4	6F	–	–

Example: The instruction

IDLE

stops the CPU clock but not the system clock.

INC — Increment

INC dst

Operation: dst ¨ dst + 1

The contents of the destination operand are incremented by one.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is dst value is +127(7FH) and result is –128(80H); cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
dst | opc	1	4	rE r = 0 to F	r
opc | dst	2	4	20	R
		4	21	IR

Examples: Given: R0 = 1BH, register 00H = 0CH, and register 1BH = 0FH:

```
INC   R0    →   R0 = 1CH
INC   00H   →   Register 00H = 0DH
INC   @R0   →   R0 = 1BH, register 01H = 10H
```

In the first example, if destination working register R0 contains the value 1BH, the statement "INC R0" leaves the value 1CH in that same register.

The next example shows the effect an INC instruction has on register 00H, assuming that it contains the value 0CH.

In the third example, INC is used in Indirect Register (IR) addressing mode to increment the value of register 1BH from 0FH to 10H.

IRET — Interrupt Return

IRET IRET

Operation:
FLAGS ¨ @SP
SP ¨ SP + 1
PC ¨ @SP
SP ¨ SP + 2
SYM(2) ¨ 1

This instruction is used at the end of an interrupt service routine. It restores the flag register and the program counter. It also re-enables global interrupts.

Flags: All flags are restored to their original settings (that is, the settings before the interrupt occurred).

Format:

IRET (Normal)	Bytes	Cycles	Opcode (Hex)
opc	1	6	BF

JP — Jump

JP cc,dst (Conditional)

JP dst (Unconditional)

Operation: If cc is true, PC ¨ dst

The conditional JUMP instruction transfers program control to the destination address if the condition specified by the condition code (cc) is true; otherwise, the instruction following the JP instruction is executed. The unconditional JP simply replaces the contents of the PC with the contents of the specified register pair. Control then passes to the statement addressed by the PC.

Flags: No flags are affected.

Format: (1)

(2)	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
cc | opc / dst	3	8 (3)	ccD cc = 0 to F	DA
opc / dst	2	8	30	IRR

NOTES:

1. The 3-byte format is used for a conditional jump and the 2-byte format for an unconditional jump.
2. In the first byte of the three-byte instruction format (conditional jump), the condition code and the opcode are both four bits.

Examples: Given: The carry flag (C) = "1", register 00 = 01H, and register 01 = 20H:

```
JP   C,LABEL_W  →   LABEL_W = 1000H, PC = 1000H
JP   @00H       →   PC = 0120H
```

The first example shows a conditional JP. Assuming that the carry flag is set to "1", the statement "JP C,LABEL_W" replaces the contents of the PC with the value 1000H and transfers control to that location. Had the carry flag not been set, control would then have passed to the statement immediately following the JP instruction.

The second example shows an unconditional JP. The statement "JP @00" replaces the contents of the PC with the contents of the register pair 00H and 01H, leaving the value 0120H.

JR — Jump Relative

JR cc,dst

Operation: If cc is true, PC ¨ PC + dst

If the condition specified by the condition code (cc) is true, the relative address is added to the program counter and control passes to the statement whose address is now in the program counter; otherwise, the instruction following the JR instruction is executed (See list of condition codes).

The range of the relative address is +127, –128, and the original value of the program counter is taken to be the address of the first instruction byte following the JR statement.

Flags: No flags are affected.

Format:

(1)

cc | opc	dst

Bytes	Cycles	Opcode (Hex)	Addr Mode dst
2	6 (2)	ccB	RA
		cc = 0 to F	

NOTE: In the first byte of the two-byte instruction format, the condition code and the opcode are each four bits.

Example: Given: The carry flag = "1" and LABEL_X = 1FF7H:

JR C,LABEL_X → PC = 1FF7H

If the carry flag is set (that is, if the condition code is true), the statement "JR C,LABEL_X" will pass control to the statement whose address is now in the PC. Otherwise, the program instruction following the JR would be executed.

LD — Load

LD dst,src

Operation: dst ¨ src

The contents of the source are loaded into the destination. The source's contents are unaffected.

Flags: No flags are affected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
dst | opc	src		2	4	rC	r	IM
				4	r8	r	R
src | opc	dst		2	4	r9	R	r
					r = 0 to F		
opc	dst | src		2	4	C7	r	lr
				4	D7	lr	r
opc	src	dst	3	6	E4	R	R
				6	E5	R	IR
opc	dst	src	3	6	E6	R	IM
				6	D6	IR	IM
opc	src	dst	3	6	F5	IR	R
opc	dst | src	x	3	6	87	r	x [r]
opc	src | dst	x	3	6	97	x [r]	r

LD — Load

LD (Continued)

Examples: Given: R0 = 01H, R1 = 0AH, register 00H = 01H, register 01H = 20H, register 02H = 02H, LOOP = 30H, and register 3AH = 0FFH:

LD	R0,#10H	→	R0 = 10H
LD	R0,01H	→	R0 = 20H, register 01H = 20H
LD	01H,R0	→	Register 01H = 01H, R0 = 01H
LD	R1,@R0	→	R1 = 20H, R0 = 01H
LD	@R0,R1	→	R0 = 01H, R1 = 0AH, register 01H = 0AH
LD	00H,01H	→	Register 00H = 20H, register 01H = 20H
LD	02H,@00H	→	Register 02H = 20H, register 00H = 01H
LD	00H,#0AH	→	Register 00H = 0AH
LD	@00H,#10H	→	Register 00H = 01H, register 01H = 10H
LD	@00H,02H	→	Register 00H = 01H, register 01H = 02, register 02H = 02H
LD	R0,#LOOP[R1]	→	R0 = 0FFH, R1 = 0AH
LD	#LOOP[R0],R1	→	Register 31H = 0AH, R0 = 01H, R1 = 0AH

LDC/LDE — Load Memory

LDC/LDE dst,src

Operation: dst ¨ src

This instruction loads a byte from program or data memory into a working register or vice-versa. The source values are unaffected. LDC refers to program memory and LDE to data memory. The assembler makes 'Irr' or 'rr' values an even number for program memory and an odd number for data memory.

Flags: No flags are affected.

Format:

					Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
1.	opc	dst | src			2	10	C3	r	Irr
2.	opc	src | dst			2	10	D3	Irr	r
3.	opc	dst | src	XS		3	12	E7	r	XS [rr]
4.	opc	src | dst	XS		3	12	F7	XS [rr]	r
5.	opc	dst | src	XL_L	XL_H	4	14	A7	r	XL [rr]
6.	opc	src | dst	XL_L	XL_H	4	14	B7	XL [rr]	r
7.	opc	dst | 0000	DA_L	DA_H	4	14	A7	r	DA
8.	opc	src | 0000	DA_L	DA_H	4	14	B7	DA	r
9.	opc	dst | 0001	DA_L	DA_H	4	14	A7	r	DA
10.	opc	src | 0001	DA_L	DA_H	4	14	B7	DA	r

NOTES:

1. The source (src) or working register pair [rr] for formats 5 and 6 cannot use register pair 0–1.
2. For formats 3 and 4, the destination address 'XS [rr]' and the source address 'XS [rr]' are each one byte.
3. For formats 5 and 6, the destination address 'XL [rr] and the source address 'XL [rr]' are each two bytes.
4. The DA and r source values for formats 7 and 8 are used to address program memory; the second set of values, used in formats 9 and 10, are used to address data memory.

LDC/LDE — Load Memory

LDC/LDE (Continued)

Examples: Given: R0 = 11H, R1 = 34H, R2 = 01H, R3 = 04H, R4 = 00H, R5 = 60H; Program memory locations 0061 = AAH, 0103H = 4FH, 0104H = 1A, 0105H = 6DH, and 1104H = 88H. External data memory locations 0061H = BBH, 0103H = 5FH, 0104H = 2AH, 0105H = 7DH, and 1104H = 98H:

```
LDC     R0,@RR2          ; R0 ¨ contents of program memory location 0104H
                         ; R0 = 1AH, R2 = 01H, R3 = 04H
LDE     R0,@RR2          ; R0 ¨ contents of external data memory location 0104H
                         ; R0 = 2AH, R2 = 01H, R3 = 04H
LDC *   @RR2,R0          ; 11H (contents of R0) is loaded into program memory
                         ; location 0104H (RR2),
                         ; working registers R0, R2, R3 Æ no change
LDE     @RR2,R0          ; 11H (contents of R0) is loaded into external data memory
                         ; location 0104H (RR2),
                         ; working registers R0, R2, R3 Æ no change
LDC     R0,#01H[RR4]     ; R0 ¨ contents of program memory location 0061H
                         ; (01H + RR4),
                         ; R0 = AAH, R2 = 00H, R3 = 60H
LDE     R0,#01H[RR4]     ; R0 ¨ contents of external data memory location 0061H
                         ; (01H + RR4), R0 = BBH, R4 = 00H, R5 = 60H
LDC (note)  #01H[RR4],R0 ; 11H (contents of R0) is loaded into program memory
                         ; location 0061H (01H + 0060H)
LDE     #01H[RR4],R0     ; 11H (contents of R0) is loaded into external data memory
                         ; location 0061H (01H + 0060H)
LDC     R0,#1000H[RR2]   ; R0 ¨ contents of program memory location 1104H
                         ; (1000H + 0104H), R0 = 88H, R2 = 01H, R3 = 04H
LDE     R0,#1000H[RR2]   ; R0 ¨ contents of external data memory location 1104H
                         ; (1000H + 0104H), R0 = 98H, R2 = 01H, R3 = 04H
LDC     R0,1104H         ; R0 ¨ contents of program memory location 1104H,
                         ; R0 = 88H
LDE     R0,1104H         ; R0 ¨ contents of external data memory location 1104H,
                         ; R0 = 98H
LDC (note)  1105H,R0     ; 11H (contents of R0) is loaded into program memory
                         ; location 1105H, (1105H) ¨ 11H
LDE     1105H,R0         ; 11H (contents of R0) is loaded into external data memory
                         ; location 1105H, (1105H) ¨ 11H
```

NOTE: These instructions are not supported by masked ROM type devices.

LDCD/LDED — Load Memory and Decrement

LDCD/LDED dst,src

Operation: dst ¨ src

rr ¨ rr – 1

These instructions are used for user stacks or block transfers of data from program or data memory to the register file. The address of the memory location is specified by a working register pair. The contents of the source location are loaded into the destination location. The memory address is then decremented. The contents of the source are unaffected.

LDCD references program memory and LDED references external data memory. The assembler makes 'Irr' an even number for program memory and an odd number for data memory.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc | dst | src	2	10	E2	r	Irr

Examples: Given: R6 = 10H, R7 = 33H, R8 = 12H, program memory location 1033H = 0CDH, and external data memory location 1033H = 0DDH:

```
LDCD    R8,@RR6     ; 0CDH (contents of program memory location 1033H) is
                    ; loaded into R8 and RR6 is decremented by one
                    ; R8 = 0CDH, R6 = 10H, R7 = 32H (RR6 ← RR6 - 1)
LDED    R8,@RR6     ; 0DDH (contents of data memory location 1033H) is
                    ; loaded into R8 and RR6 is decremented by one
                    ; (RR6 ← RR6 - 1) R8 = 0DDH, R6 = 10H, R7 = 32H
```

LDCI/LDEI — Load Memory and Increment

LDCI/LDEI dst,src

Operation: dst ¨ src

rr ¨ rr + 1

These instructions are used for user stacks or block transfers of data from program or data memory to the register file. The address of the memory location is specified by a working register pair. The contents of the source location are loaded into the destination location. The memory address is then incremented automatically. The contents of the source are unaffected.

LDCI refers to program memory and LDEI refers to external data memory. The assembler makes 'Irr' even for program memory and odd for data memory.

Flags: No flags are affected.

Format:

		Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	dst | src	2	10	E3	r	Irr

Examples: Given: R6 = 10H, R7 = 33H, R8 = 12H, program memory locations 1033H = 0CDH and 1034H = 0C5H; external data memory locations 1033H = 0DDH and 1034H = 0D5H:

```
LDCI    R8,@RR6     ; 0CDH (contents of program memory location 1033H) is
                    ; loaded into R8 and RR6 is incremented by one
                    ; (RR6 ¨ RR6 + 1) R8 = 0CDH, R6 = 10H, R7 = 34H
LDEI    R8,@RR6     ; 0DDH (contents of data memory location 1033H) is
                    ; loaded into R8 and RR6 is incremented by one
                    ; (RR6 ¨ RR6 + 1) R8 = 0DDH, R6 = 10H, R7 = 34H
```

NOP — No Operation

NOP

Operation: No action is performed when the CPU executes this instruction. Typically, one or more NOPs are executed in sequence in order to effect a timing delay of variable duration.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	FF

Example: When the instruction

NOP

is encountered in a program, no operation occurs. Instead, there is a delay in instruction execution time.

OR — Logical OR

OR dst,src

Operation: dst ¨ dst OR src

The source operand is logically ORed with the destination operand and the result is stored in the destination. The contents of the source are unaffected. The OR operation results in a "1" being stored whenever either of the corresponding bits in the two operands is a "1"; otherwise a "0" is stored.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always cleared to "0".
D: Unaffected.
H: Unaffected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc	dst | src		2	4	42	r	r
				6	43	r	lr
opc	src	dst	3	6	44	R	R
				6	45	R	IR
opc	dst	src	3	6	46	R	IM

Examples: Given: R0 = 15H, R1 = 2AH, R2 = 01H, register 00H = 08H, register 01H = 37H, and register 08H = 8AH:

OR	R0,R1	→	R0 = 3FH, R1 = 2AH
OR	R0,@R2	→	R0 = 37H, R2 = 01H, register 01H = 37H
OR	00H,01H	→	Register 00H = 3FH, register 01H = 37H
OR	01H,@00H	→	Register 00H = 08H, register 01H = 0BFH
OR	00H,#02H	→	Register 00H = 0AH

In the first example, if working register R0 contains the value 15H and register R1 the value 2AH, the statement "OR R0,R1" logical-ORs the R0 and R1 register contents and stores the result (3FH) in destination register R0.

The other examples show the use of the logical OR instruction with the various addressing modes and formats.

POP — Pop From Stack

POP dst

Operation: dst ¨ @SP

SP ¨ SP + 1

The contents of the location addressed by the stack pointer are loaded into the destination. The stack pointer is then incremented by one.

Flags: No flags affected.

Format:

		Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc	dst	2	8	50	R
			8	51	IR

Examples: Given: Register 00H = 01H, register 01H = 1BH, SP (0D9H) = 0BBH, and stack register 0BBH = 55H:

```
POP   00H    →   Register 00H = 55H, SP = 0BCH
POP   @00H   →   Register 00H = 01H, register 01H = 55H, SP = 0BCH
```

In the first example, general register 00H contains the value 01H. The statement "POP 00H" loads the contents of location 0BBH (55H) into destination register 00H and then increments the stack pointer by one. Register 00H then contains the value 55H and the SP points to location 0BCH.

PUSH — Push To Stack

PUSH src

Operation: SP ¨ SP – 1

@SP ¨ src

A PUSH instruction decrements the stack pointer value and loads the contents of the source (src) into the location addressed by the decremented stack pointer. The operation then adds the new value to the top of the stack.

Flags: No flags are affected.

Format:

		Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc	src	2	8	70	R
			8	71	IR

Examples: Given: Register 40H = 4FH, register 4FH = 0AAH, SP = 0C0H:

PUSH 40H → Register 40H = 4FH, stack register 0BFH = 4FH, SP = 0BFH

PUSH @40H → Register 40H = 4FH, register 4FH = 0AAH, stack register 0BFH = 0AAH, SP = 0BFH

In the first example, if the stack pointer contains the value 0C0H, and general register 40H the value 4FH, the statement "PUSH 40H" decrements the stack pointer from 0C0 to 0BFH. It then loads the contents of register 40H into location 0BFH. Register 0BFH then contains the value 4FH and SP points to location 0BFH.

RCF — Reset Carry Flag

RCF RCF

Operation: C ¨ 0

The carry flag is cleared to logic zero, regardless of its previous value.

Flags: **C:** Cleared to "0".

No other flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	CF

Example: Given: C = "1" or "0":

The instruction RCF clears the carry flag (C) to logic zero.

RET — Return

RET

Operation: PC ¨ @SP

SP ¨ SP + 2

The RET instruction is normally used to return to the previously executing procedure at the end of a procedure entered by a CALL instruction. The contents of the location addressed by the stack pointer are popped into the program counter. The next statement that is executed is the one that is addressed by the new program counter value.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	8	AF

Example: Given: SP = 0BCH, (SP) = 101AH, and PC = 1234:

RET → PC = 101AH, SP = 0BEH

The statement "RET" pops the contents of stack pointer location 0BCH (10H) into the high byte of the program counter. The stack pointer then pops the value in location 0BDH (1AH) into the PC's low byte and the instruction at location 101AH is executed. The stack pointer now points to memory location 0BEH.

RL — Rotate Left

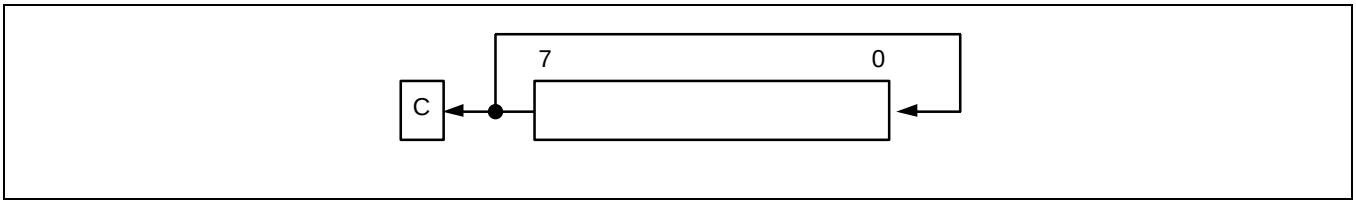
RL dst

Operation: C ¨ dst (7)

dst (0) ¨ dst (7)

dst (n + 1) ¨ dst (n), n = 0–6

The contents of the destination operand are rotated left one bit position. The initial value of bit 7 is moved to the bit zero (LSB) position and also replaces the carry flag.



Flags:
C: Set if the bit rotated from the most significant bit position (bit 7) was "1".
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the sign of the destination changed during rotation; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	90	R
		4	91	IR

Examples: Given: Register 00H = 0AAH, register 01H = 02H and register 02H = 17H:

```
RL    00H    →    Register 00H = 55H, C = "1"
RL    @01H   →    Register 01H = 02H, register 02H = 2EH, C = "0"
```

In the first example, if general register 00H contains the value 0AAH (10101010B), the statement "RL 00H" rotates the 0AAH value left one bit position, leaving the new value 55H (01010101B) and setting the carry and overflow flags.

RLC — Rotate Left Through Carry

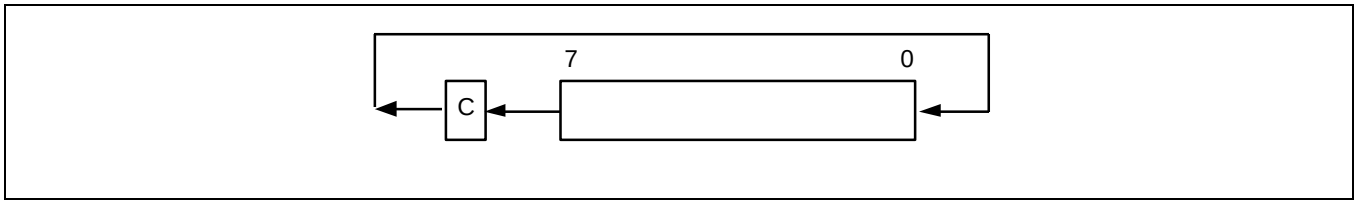
RLC dst

Operation: dst (0) ¨ C

C ¨ dst (7)

dst (n + 1) ¨ dst (n), n = 0–6

The contents of the destination operand with the carry flag are rotated left one bit position. The initial value of bit 7 replaces the carry flag (C); the initial value of the carry flag replaces bit zero.



Flags:
C: Set if the bit rotated from the most significant bit position (bit 7) was "1".
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the sign of the destination changed during rotation; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	10	R
		4	11	IR

Examples: Given: Register 00H = 0AAH, register 01H = 02H, and register 02H = 17H, C = "0":

```
RLC   00H    →   Register 00H = 54H, C = "1"
RLC   @01H   →   Register 01H = 02H, register 02H = 2EH, C = "0"
```

In the first example, if general register 00H has the value 0AAH (10101010B), the statement "RLC 00H" rotates 0AAH one bit position to the left. The initial value of bit 7 sets the carry flag and the initial value of the C flag replaces bit zero of register 00H, leaving the value 55H (01010101B). The MSB of register 00H resets the carry flag to "1" and sets the overflow flag.

RR — Rotate Right

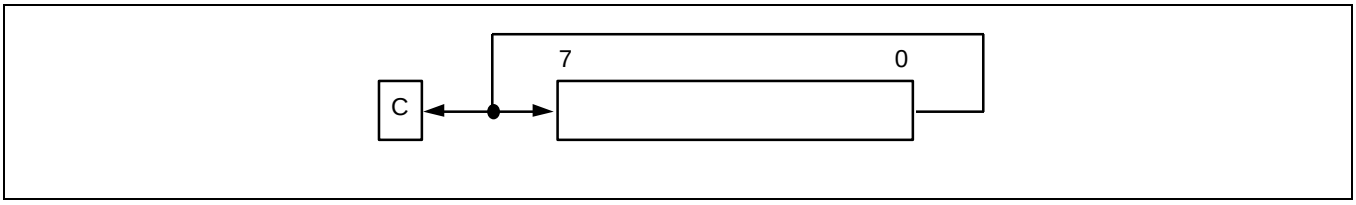
RR dst

Operation: C ¨ dst (0)

dst (7) ¨ dst (0)

dst (n) ¨ dst (n + 1), n = 0–6

The contents of the destination operand are rotated right one bit position. The initial value of bit zero (LSB) is moved to bit 7 (MSB) and also replaces the carry flag (C).



Flags:

C: Set if the bit rotated from the least significant bit position (bit zero) was "1".
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the sign of the destination changed during rotation; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	E0	R
		4	E1	IR

Examples: Given: Register 00H = 31H, register 01H = 02H, and register 02H = 17H:

```
RR    00H    →    Register 00H = 98H, C = "1"
RR    @01H   →    Register 01H = 02H, register 02H = 8BH, C = "1"
```

In the first example, if general register 00H contains the value 31H (00110001B), the statement "RR 00H" rotates this value one bit position to the right. The initial value of bit zero is moved to bit 7, leaving the new value 98H (10011000B) in the destination register. The initial bit zero also resets the C flag to "1" and the sign flag and overflow flag are also set to "1".

RRC — Rotate Right Through Carry

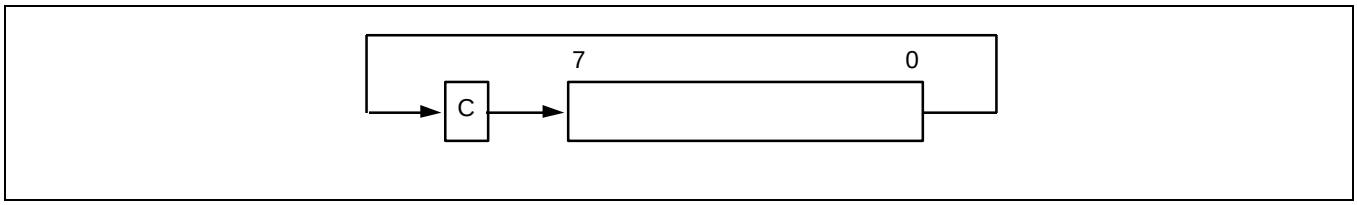
RRC dst

Operation: dst (7) ¨ C

C ¨ dst (0)

dst (n) ¨ dst (n + 1), n = 0–6

The contents of the destination operand and the carry flag are rotated right one bit position. The initial value of bit zero (LSB) replaces the carry flag; the initial value of the carry flag replaces bit 7 (MSB).



Flags:
C: Set if the bit rotated from the least significant bit position (bit zero) was "1".
Z: Set if the result is "0" cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the sign of the destination changed during rotation; cleared otherwise.
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	C0	R
		4	C1	IR

Examples: Given: Register 00H = 55H, register 01H = 02H, register 02H = 17H, and C = "0":

```
RRC   00H    →   Register 00H = 2AH, C = "1"
RRC   @01H   →   Register 01H = 02H, register 02H = 0BH, C = "1"
```

In the first example, if general register 00H contains the value 55H (01010101B), the statement "RRC 00H" rotates this value one bit position to the right. The initial value of bit zero ("1") replaces the carry flag and the initial value of the C flag ("1") replaces bit 7. This leaves the new value 2AH (00101010B) in destination register 00H. The sign flag and overflow flag are both cleared to "0".

SBC — Subtract With Carry

SBC dst,src

Operation: dst ¨ dst – src – c

The source operand, along with the current value of the carry flag, is subtracted from the destination operand and the result is stored in the destination. The contents of the source are unaffected. Subtraction is performed by adding the two's-complement of the source operand to the destination operand. In multiple precision arithmetic, this instruction permits the carry ("borrow") from the subtraction of the low-order operands to be subtracted from the subtraction of high-order operands.

Flags:

C: Set if a borrow occurred (src > dst); cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the operands were of opposite sign and the sign

f the result is the same as the sign of the source; cleared otherwise.
D: Always set to "1".
H: Cleared if there is a carry from the most significant bit of the low-order four bits of the result; set otherwise, indicating a "borrow".

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc	dst | src		2	4	32	r	r
				6	33	r	lr
opc	src	dst	3	6	34	R	R
				6	35	R	IR
opc	dst	src	3	6	36	R	IM

Examples: Given: R1 = 10H, R2 = 03H, C = "1", register 01H = 20H, register 02H = 03H, and register 03H = 0AH:

SBC	R1,R2	→	R1 = 0CH, R2 = 03H
SBC	R1,@R2	→	R1 = 05H, R2 = 03H, register 03H = 0AH
SBC	01H,02H	→	Register 01H = 1CH, register 02H = 03H
SBC	01H,@02H	→	Register 01H = 15H,register 02H = 03H, register 03H = 0AH
SBC	01H,#8AH	→	Register 01H = 95H; C, S, and V = "1"

In the first example, if working register R1 contains the value 10H and register R2 the value 03H, the statement "SBC R1,R2" subtracts the source value (03H) and the C flag value ("1") from the destination (10H) and then stores the result (0CH) in register R1.

SCF — Set Carry Flag

SCF

Operation: C ¨ 1

The carry flag (C) is set to logic one, regardless of its previous value.

Flags: **C:** Set to "1".

No other flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4	DF

Example: The statement

SCF

sets the carry flag to logic one.

SRA — Shift Right Arithmetic

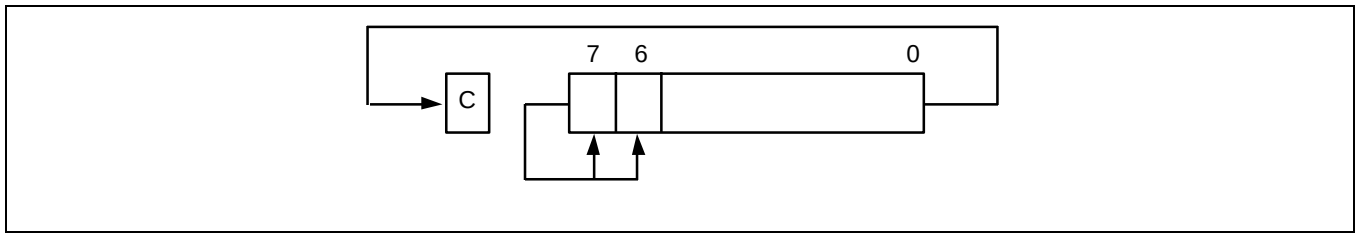
SRA dst

Operation: dst (7) ¨ dst (7)

C ¨ dst (0)

dst (n) ¨ dst (n + 1), n = 0–6

An arithmetic shift-right of one bit position is performed on the destination operand. Bit zero (the LSB) replaces the carry flag. The value of bit 7 (the sign bit) is unchanged and is shifted into bit position 6.



Flags:
C: Set if the bit shifted from the LSB position (bit zero) was "1".
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Always cleared to "0".
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst
opc dst	2	4	D0	R
		4	D1	IR

Examples: Given: Register 00H = 9AH, register 02H = 03H, register 03H = 0BCH, and C = "1":

```
SRA   00H    →   Register 00H = 0CD, C = "0"
SRA   @02H   →   Register 02H = 03H, register 03H = 0DEH, C = "0"
```

In the first example, if general register 00H contains the value 9AH (10011010B), the statement "SRA 00H" shifts the bit values in register 00H right one bit position. Bit zero ("0") clears the C flag and bit 7 ("1") is then shifted into the bit 6 position (bit 7 remains unchanged). This leaves the value 0CDH (11001101B) in destination register 00H.

STOP — Stop Operation

STOP

Operation:

The STOP instruction stops both the CPU clock and system clock and causes the microcontroller to enter Stop mode. During Stop mode, the contents of on-chip CPU registers, peripheral registers, and I/O port control and data registers are retained. Stop mode can be released by an external reset operation or External interrupt input. For the reset operation, the RESET pin must be held to Low level until the required oscillation stabilization interval has elapsed.

Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	1	4	7F	–	–

Example: The statement

STOP

halts all microcontroller operations.

SUB — Subtract

SUB dst,src

Operation: dst ¨ dst – src

The source operand is subtracted from the destination operand and the result is stored in the destination. The contents of the source are unaffected. Subtraction is performed by adding the two's complement of the source operand to the destination operand.

Flags:
C: Set if a "borrow" occurred; cleared otherwise.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result is negative; cleared otherwise.
V: Set if arithmetic overflow occurred, that is, if the operands were of opposite signs and the sign of the result is of the same as the sign of the source operand; cleared otherwise.
D: Always set to "1".
H: Cleared if there is a carry from the most significant bit of the low-order four bits of the result; set otherwise indicating a "borrow".

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc	dst | src		2	4	22	r	r
				6	23	r	lr
opc	src	dst	3	6	24	R	R
				6	25	R	IR
opc	dst	src	3	6	26	R	IM

Examples: Given: R1 = 12H, R2 = 03H, register 01H = 21H, register 02H = 03H, register 03H = 0AH:

SUB	R1,R2	→	R1 = 0FH, R2 = 03H
SUB	R1,@R2	→	R1 = 08H, R2 = 03H
SUB	01H,02H	→	Register 01H = 1EH, register 02H = 03H
SUB	01H,@02H	→	Register 01H = 17H, register 02H = 03H
SUB	01H,#90H	→	Register 01H = 91H; C, S, and V = "1"
SUB	01H,#65H	→	Register 01H = 0BCH; C and S = "1", V = "0"

In the first example, if working register R1 contains the value 12H and if register R2 contains the value 03H, the statement "SUB R1,R2" subtracts the source value (03H) from the destination value (12H) and stores the result (0FH) in destination register R1.

TCM — Test Complement Under Mask

TCM dst,src

Operation: (NOT dst) AND src

This instruction tests selected bits in the destination operand for a logic one value. The bits to be tested are specified by setting a "1" bit in the corresponding position of the source operand (mask). The TCM statement complements the destination operand, which is then ANDed with the source mask. The zero (Z) flag can then be checked to determine the result. The destination and source operands are unaffected.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always cleared to "0".
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	src
opc | dst | src	2	4	62	r	r
		6	63	r	lr
opc | src | dst	3	6	64	R	R
		6	65	R	IR
opc | dst | src	3	6	66	R	IM

Examples: Given: R0 = 0C7H, R1 = 02H, R2 = 12H, register 00H = 2BH, register 01H = 02H, and register 02H = 23H:

TCM	R0,R1	→	R0 = 0C7H, R1 = 02H, Z = "1"
TCM	R0,@R1	→	R0 = 0C7H, R1 = 02H, register 02H = 23H, Z = "0"
TCM	00H,01H	→	Register 00H = 2BH, register 01H = 02H, Z = "1"
TCM	00H,@01H	→	Register 00H = 2BH, register 01H = 02H, register 02H = 23H, Z = "1"
TCM	00H,#34	→	Register 00H = 2BH, Z = "0"

In the first example, if working register R0 contains the value 0C7H (11000111B) and register R1 the value 02H (00000010B), the statement "TCM R0,R1" tests bit one in the destination register for a "1" value. Because the mask value corresponds to the test bit, the Z flag is set to logic one and can be tested to determine the result of the TCM operation.

TM — Test Under Mask

TM dst,src

Operation: dst AND src

This instruction tests selected bits in the destination operand for a logic zero value. The bits to be tested are specified by setting a "1" bit in the corresponding position of the source operand (mask), which is ANDed with the destination operand. The zero (Z) flag can then be checked to determine the result. The destination and source operands are unaffected.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always reset to "0".
D: Unaffected.
H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc | dst | src	2	4	72	r	r
		6	73	r	lr
opc | src | dst	3	6	74	R	R
		6	75	R	IR
opc | dst | src	3	6	76	R	IM

Examples: Given: R0 = 0C7H, R1 = 02H, R2 = 18H, register 00H = 2BH, register 01H = 02H, and register 02H = 23H:

```
TM    R0,R1      →   R0 = 0C7H, R1 = 02H, Z = "0"
TM    R0,@R1     →   R0 = 0C7H, R1 = 02H, register 02H = 23H, Z = "0"
TM    00H,01H    →   Register 00H = 2BH, register 01H = 02H, Z = "0"
TM    00H,@01H   →   Register 00H = 2BH, register 01H = 02H,
                     register 02H = 23H, Z = "0"
TM    00H,#54H   →   Register 00H = 2BH, Z = "1"
```

In the first example, if working register R0 contains the value 0C7H (11000111B) and register R1 the value 02H (00000010B), the statement "TM R0,R1" tests bit one in the destination register for a "0" value. Because the mask value does not match the test bit, the Z flag is cleared to logic zero and can be tested to determine the result of the TM operation.

XOR — Logical Exclusive OR

XOR dst,src

Operation: dst ¨ dst XOR src

The source operand is logically exclusive-ORed with the destination operand and the result is stored in the destination. The exclusive-OR operation results in a "1" bit being stored whenever the corresponding bits in the operands are different; otherwise, a "0" bit is stored.

Flags:
C: Unaffected.
Z: Set if the result is "0"; cleared otherwise.
S: Set if the result bit 7 is set; cleared otherwise.
V: Always reset to "0".
D: Unaffected.
H: Unaffected.

Format:

			Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc	dst | src		2	4	B2	r	r
				6	B3	r	lr
opc	src	dst	3	6	B4	R	R
				6	B5	R	IR
opc	dst	src	3	6	B6	R	IM

Examples: Given: R0 = 0C7H, R1 = 02H, R2 = 18H, register 00H = 2BH, register 01H = 02H, and register 02H = 23H:

XOR	R0,R1	→	R0 = 0C5H, R1 = 02H
XOR	R0,@R1	→	R0 = 0E4H, R1 = 02H, register 02H = 23H
XOR	00H,01H	→	Register 00H = 29H, register 01H = 02H
XOR	00H,@01H	→	Register 00H = 08H, register 01H = 02H, register 02H = 23H
XOR	00H,#54H	→	Register 00H = 7FH

In the first example, if working register R0 contains the value 0C7H and if register R1 contains the value 02H, the statement "XOR R0,R1" logically exclusive-ORs the R1 value with the R0 value and stores the result (0C5H) in the destination register R0.

7 CLOCK CIRCUITS

OVERVIEW

The S3C9228 microcontroller has two oscillator circuits: a main clock, and a sub clock circuit. The CPU and peripheral hardware operate on the system clock frequency supplied through these circuits. The maximum CPU clock frequency, is determined by CLKCON register settings.

SYSTEM CLOCK CIRCUIT

The system clock circuit has the following components:

— Crystal, ceramic resonator, RC oscillation source (main clock only), or an external clock
— Oscillator stop and wake-up functions
— Programmable frequency divider for the CPU clock (fxx divided by 1, 2, 8, or 16)
— Clock circuit control register, CLKCON
— Oscillator control register, OSCCON

CPU CLOCK NOTATION

In this document, the following notation is used for descriptions of the CPU clock:

fx main clock
fxt sub clock
fxx selected system clock

MAIN OSCILLATOR CIRCUITS

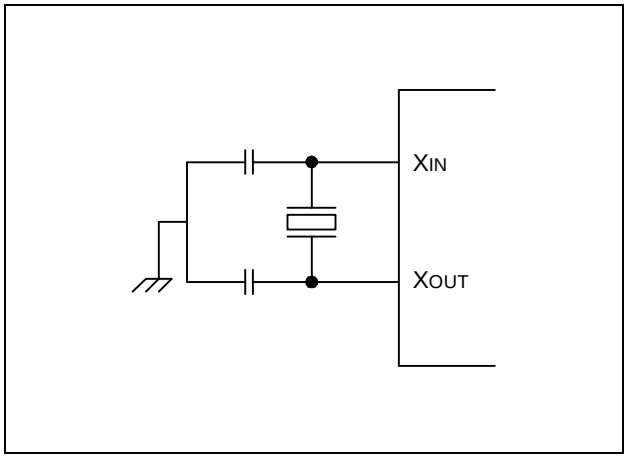


Figure 7-1. Crystal/Ceramic Oscillator

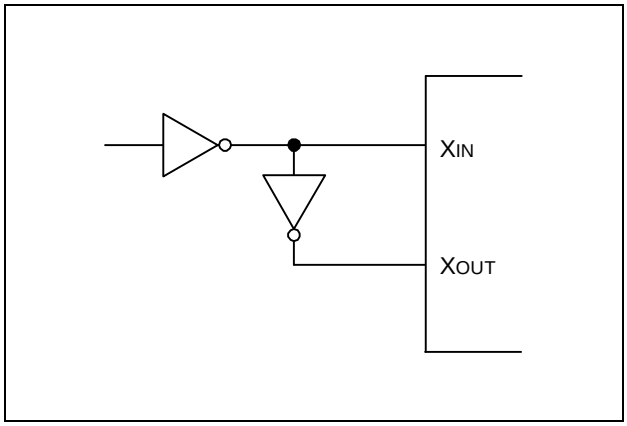


Figure 7-2. External Oscillator

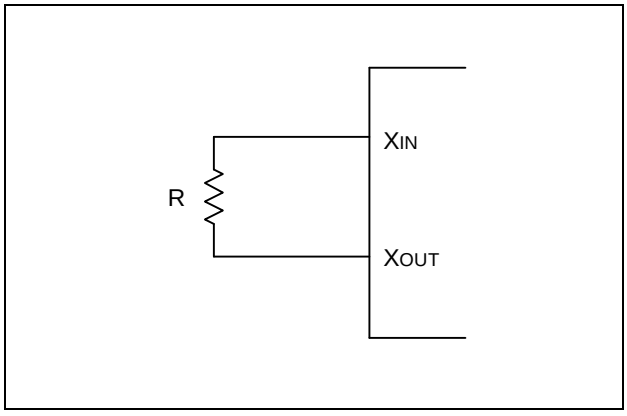


Figure 7-3. RC Oscillator

SUB OSCILLATOR CIRCUITS

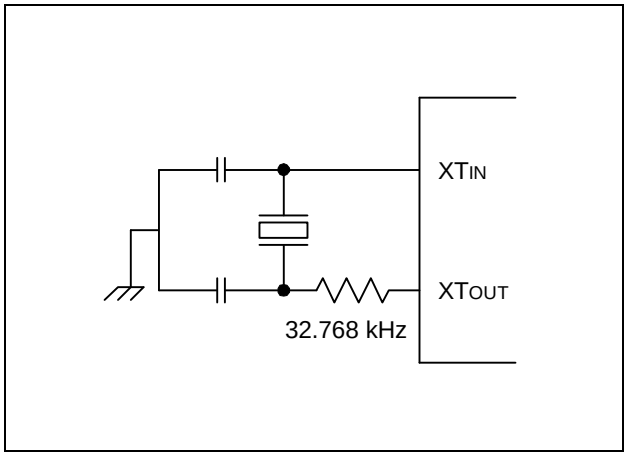


Figure 7-4. Crystal/Ceramic Oscillator

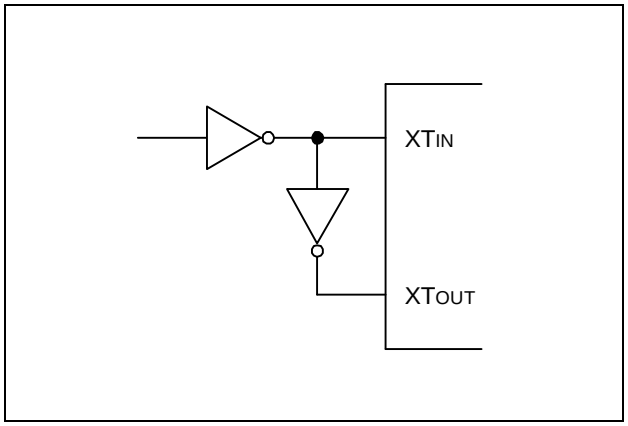


Figure 7-5. External Oscillator

CLOCK STATUS DURING POWER-DOWN MODES

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

— In Stop mode, the main oscillator is halted. Stop mode is released, and the oscillator started, by a reset operation, by an external interrupt, or by an internal interrupt if sub clock is selected as the clock source (When the fx is selected as system clock).

— In Idle mode, the internal clock signal is gated away from the CPU, but continues to be supplied to the interrupt structure, timer A/B, and watch timer. Idle mode is released by a reset or by an external or internal interrupts.

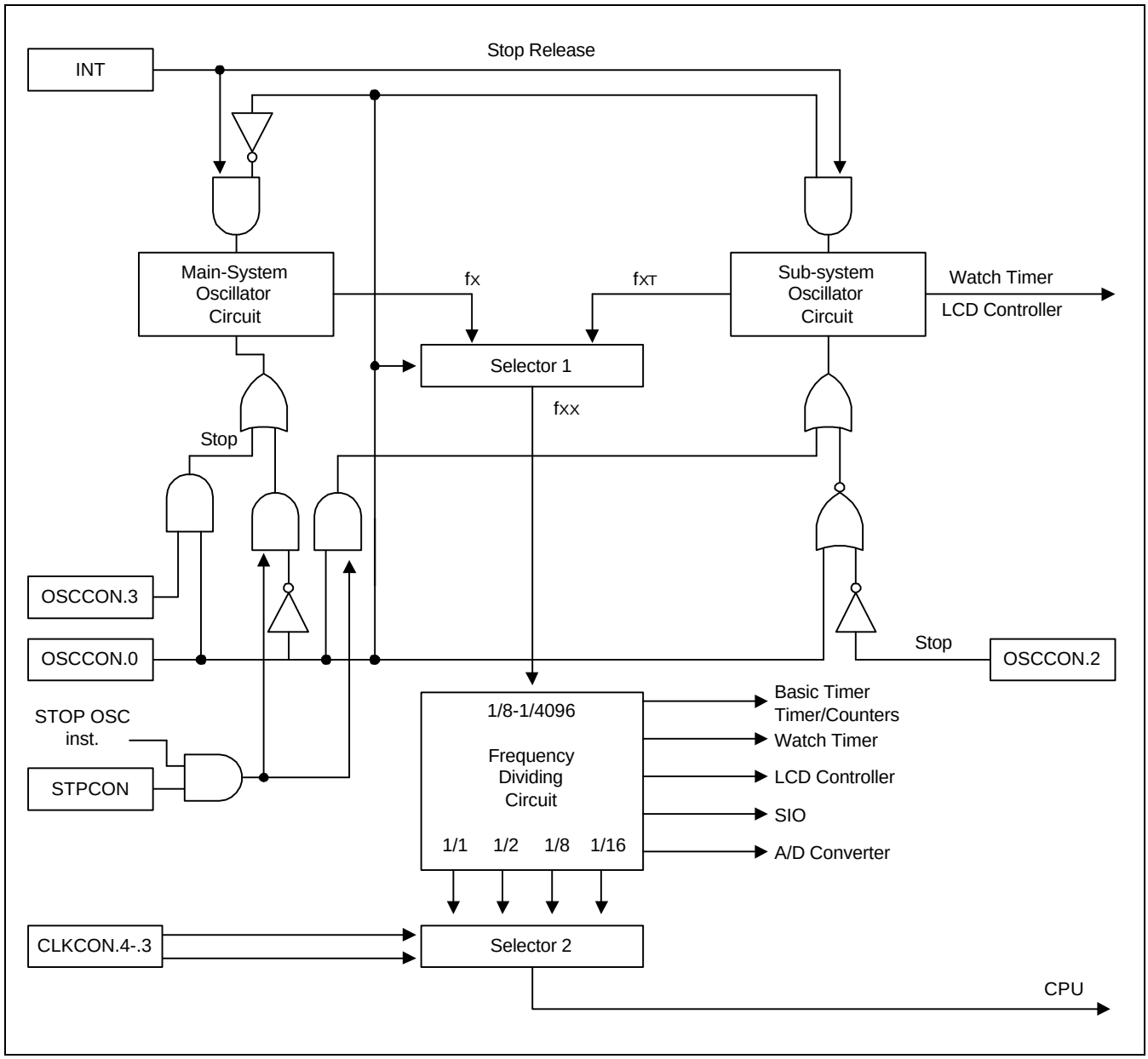


Figure 7-6. System Clock Circuit Diagram

SYSTEM CLOCK CONTROL REGISTER (CLKCON)

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

— Oscillator IRQ wake-up function enable/disable

— Oscillator frequency divide-by value

CLKCON register settings control 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 fx/16 (the slowest clock speed) is selected as the CPU clock. If necessary, you can then increase the CPU clock speed to fx, fx/2, or fx/8 by setting the CLKCON, and you can change system clock from main clock to sub clock by setting the OSCCON.

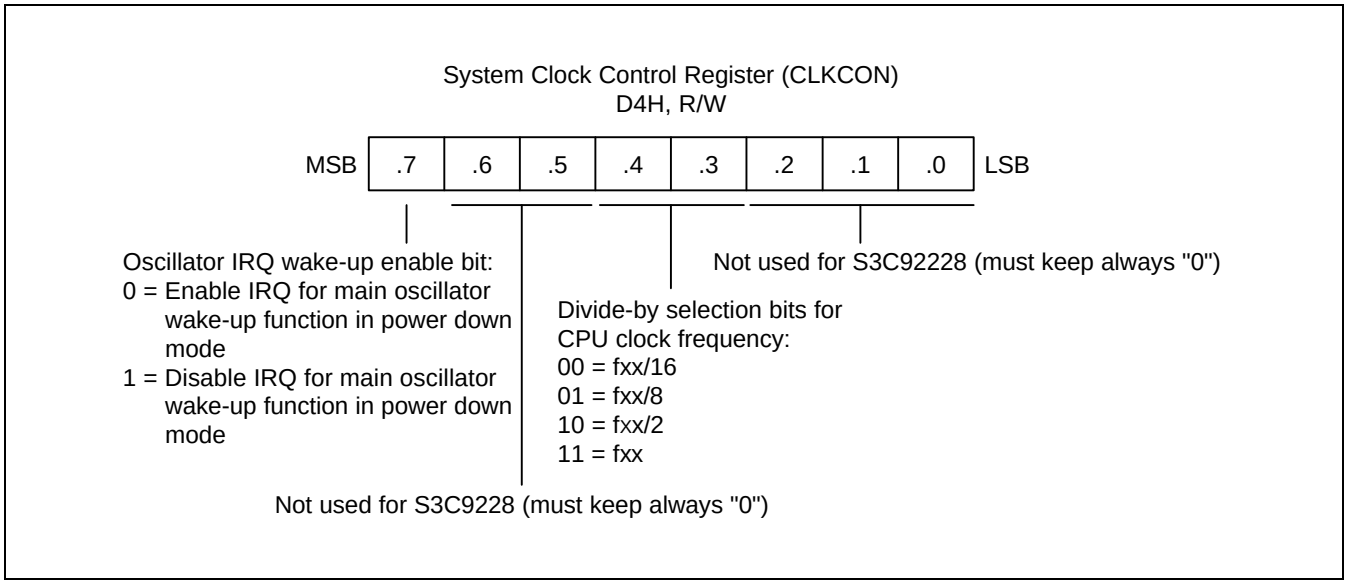


Figure 7-7. System Clock Control Register (CLKCON)

OSCILLATOR CONTROL REGISTER (OSCCON)

The oscillator control register, OSCCON, is located in address D3H. It is read/write addressable and has the following functions:

— System clock selection
— Main oscillator control
— Sub oscillator control

OSCCON.0 register settings select Main clock or Sub clock as system clock.
After a reset, Main clock is selected for system clock because the reset value of OSCCON.0 is "0".

The main oscillator can be stopped or run by setting OSCCON.3.

The sub oscillator can be stopped or run by setting OSCCON.2.

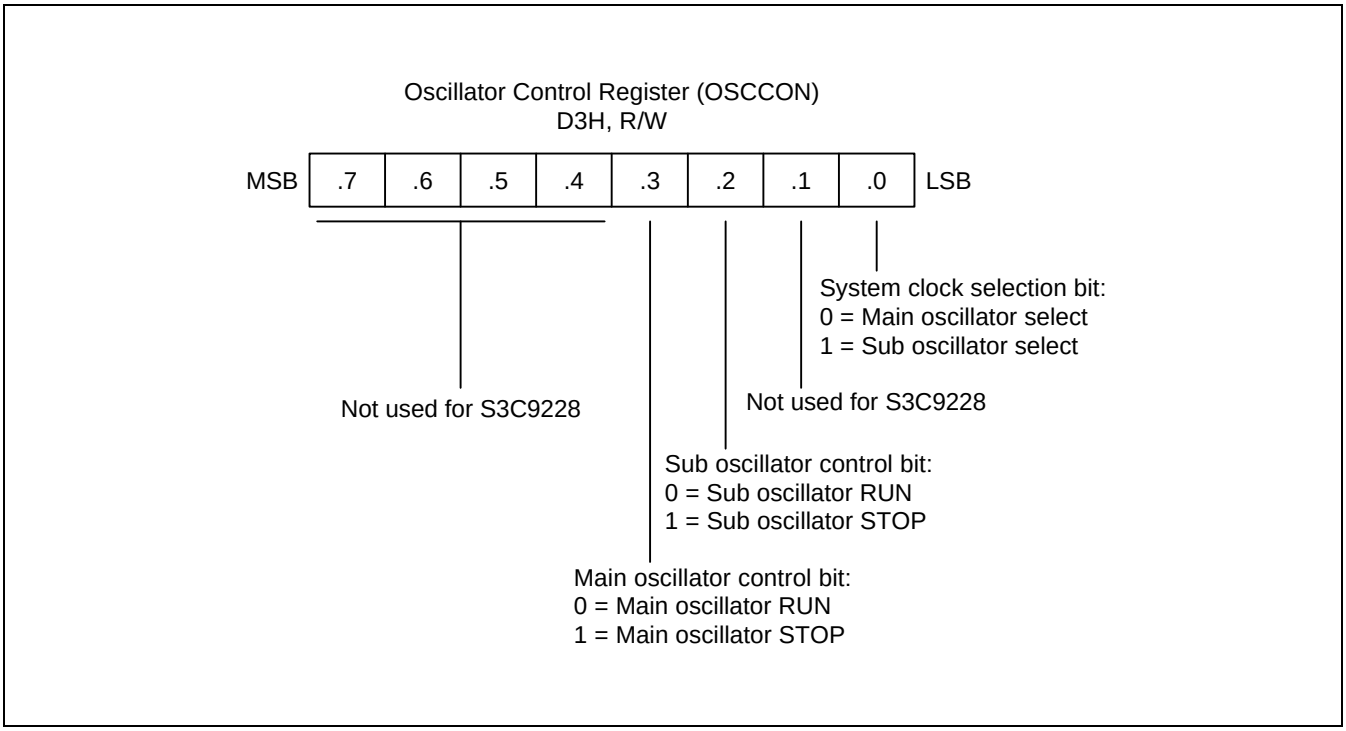


Figure 7-8. Oscillator Control Register (OSCCON)

SWITCHING THE CPU CLOCK

Data loadings in the oscillator control register, OSCCON, determine whether a main or a sub clock is selected as the CPU clock, and also how this frequency is to be divided by setting CLKCON. This makes it possible to switch dynamically between main and sub clocks and to modify operating frequencies.

OSCCON.0 select the main clock (fx) or the sub clock (fxt) for the system clock. OSCCON .3 start or stop main clock oscillation, and OSCCON.2 start or stop sub clock oscillation. CLKCON.4–.3 control the frequency divider circuit, and divide the selected fxx clock by 1, 2, 8, or 16.

For example, you are using the default system clock (normal operating mode and a main clock of fx/16) and you want to switch from the fx clock to a sub clock and to stop the main clock. To do this, you need to set OSCCON.0 to "1", take a delay, and OSCCON.3 to "1" sequently. This switches the clock from fx to fxt and stops main clock oscillation.

The following steps must be taken to switch from a sub clock to the main clock: first, set OSCCON.3 to "0" to enable main system clock oscillation. Then, after a certain number of machine cycles has elapsed, select the main clock by setting OSCCON.0 to "0".

☞ PROGRAMMING TIP — Switching the CPU clock

1. This example shows how to change from the main clock to the sub clock:

```
MA2SUB  OR      OSCCON,#01H     ;  Switches to the sub clock
        CALL    DLY16           ;  Delay 16ms
        OR      OSCCON,#08H     ;  Stop the main clock oscillation
        RET
```

2. This example shows how to change from sub clock to main clock:

```
SUB2MA  AND     OSCCON,#0F7H    ;  Start the main clock oscillation
        CALL    DLY16           ;  Delay 16 ms
        AND     OSCCON,#0FEH    ;  Switch to the main clock
        RET
DLY16   LD      R0,#20H
DEL     NOP
        DEC     R0
        JR      NZ,DEL
        RET
```

STOP CONTROL REGISTER (STPCON)

The STOP control register, STPCON, is located in address E0H. It is read/write addressable and has the following functions:

— Enable/Disable STOP instruction

After a reset, the STOP instruction is disabled, because the value of STPCON is "other values".
If necessary, you can use the STOP instruction by setting the value of STPCON to "10100101B".

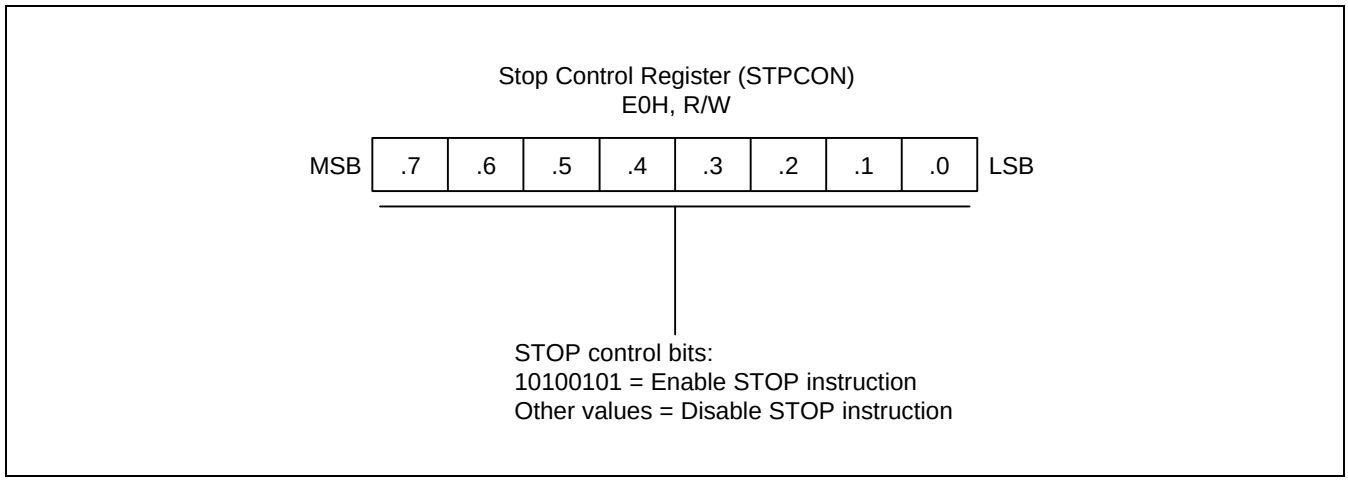


Figure 7-9. STOP Control Register (STPCON)

☞ PROGRAMMING TIP — How to Use Stop Instruction

This example shows how to go STOP mode when a main clock is selected as the system clock.

```
LD      STOPCON,#1010010B   ;  Enable STOP instruction
STOP                        ;  Enter STOP mode
NOP
NOP
NOP                         ;  Release STOP mode
LD      STOPCON,#00000000B  ;  Disable STOP instruction
```

NOTES

8 RESET and POWER-DOWN

SYSTEM RESET

OVERVIEW

During a power-on reset, the voltage at V_{DD} goes to High level and the RESET pin is forced to Low level. The RESET signal is input through a schmitt trigger circuit where it is then synchronized with the CPU clock. This procedure brings S3C9228/P9228 into a known operating status.

To allow time for internal CPU clock oscillation to stabilize, the RESET pin must be held to Low level for a minimum time interval after the power supply comes within tolerance. The minimum required oscillation stabilization time for a reset operation is 1 millisecond.

Whenever a reset occurs during normal operation (that is, when both V_{DD} and RESET are High level), the RESET pin is forced Low and the reset operation starts. All system and peripheral control registers are then reset to their default hardware values (see Table 8-1).

In summary, the following sequence of events occurs during a reset operation:

— All interrupts are disabled.

— The watchdog function (basic timer) is enabled.

— The P0.0–P0.3, P1, and P2.2–P2.3 are set to schmitt trigger input mode and all pull-up resistors are disabled for the I/O port pin circuits.

— Peripheral control and data registers are disabled and reset to their default hardware values.

— The program counter (PC) is loaded with the program reset address in the ROM, 0100H.

— When the programmed oscillation stabilization time interval has elapsed, the instruction stored in ROM location 0100H (and 0101H) is fetched and executed.

NOTE

To program the duration of the oscillation stabilization interval, you 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. In Stop mode, the operation of the CPU and main oscillator is halted. All peripherals which the main oscillator is selected as a clock source stop also because main oscillator stops. But the watch timer and LCD controller will not halted in stop mode if the sub clock is selected as watch timer clock source. The data stored in the internal register file are retained in stop mode. Stop mode can be released in one of three ways: by a system reset, by an internal watch timer interrupt (when sub clock is selected as clock source of watch timer), or by an external interrupt.

Example:

```
LD      STOPCON,#10100101B
STOP
NOP
NOP
NOP
LD      STOPCON,#00000000B
```

NOTES

1. Do not use stop mode if you are using an external clock source because X_{IN} input must be restricted internally to V_{SS} to reduce current leakage.
2. In application programs, a STOP instruction must be immediately followed by at least three NOP instructions. This ensures an adequate time interval for the clock to stabilize before the next instruction is executed. If three or more NOP instructions are not used after STOP instruction, leakage current could be flown because of the floating state in the internal bus.
3. To enable/disable STOP instruction, the STOPCON register should be written with 10100101B/other values before/after stop instruction.

Using RESET to Release Stop Mode

Stop mode is released when the RESET signal goes active (Low level): all system and peripheral control registers are reset to their default hardware values and the contents of all data registers are retained. When the programmed oscillation stabilization interval has elapsed, the CPU starts the system initialization routine by fetching the program instruction stored in ROM location 0100H.

Using an External Interrupt to Release Stop Mode

External interrupts can be used to release stop mode. For the S3C9228 microcontroller, we recommend using the INT interrupt, P0, P1, and P3.

Using an Internal Interrupt to Release Stop Mode

An internal interrupt, watch timer, can be used to release stop mode because the watch timer operates in stop mode if the clock source of watch timer is sub clock. If system clock is sub clock, you can't use any interrupts to release stop mode. That is, you had better use the idle instruction instead of stop one when sub clock is selected as the system clock.

Please note the following conditions for Stop mode release:

— If you release stop mode using an internal or external interrupt, the current values in system and peripheral control registers are unchanged.

— If you use an internal or 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.

— If you use an interrupt to release stop mode, the bit-pair setting for CLKCON.4/CLKCON.3 remains unchanged and the currently selected clock value is used.

— The internal or 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.

IDLE MODE

Idle mode is invoked by the instruction IDLE (opcode 6FH). In Idle mode, CPU operations are halted while some peripherals remain active. During Idle mode, the internal clock signal is gated away from the CPU and from all but the following peripherals, which remain active:

— Interrupt logic

— Basic timer

— Timer 1 (Timer A and B)

— Watch timer

— LCD controller

I/O port pins retain the mode (input or output) they had at the time Idle mode was entered.

Idle Mode Release

You can release Idle mode in one of two ways:

1. Execute a 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 the *slowest clock (1/16)* because of the hardware reset value for the CLKCON register. If all external interrupts are masked in the IMR register, a reset is the only way you can release Idle mode.
2. Activate any enabled interrupt — internal or external. When you use an interrupt to release Idle mode, the 2-bit CLKCON.4/CLKCON.3 value remains unchanged, and the *currently selected clock* value is used. The interrupt is then serviced. When the return-from-interrupt condition (IRET) occurs, the instruction immediately following the one which initiated Idle mode is executed.

HARDWARE RESET VALUES

Table 8-1 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 table 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 Values after RESET

Register Name	Mnemonic	Address		Bit Values after RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
Locations B8H–B9H are not mapped.											
Timer B Control Register	TBCON	202	BAH	–	0	0	0	0	0	0	–
Timer 1/A Control Register	TACON	203	BBH	0	0	0	0	0	0	0	–
Timer B Data Register	TBDATA	204	BCH	1	1	1	1	1	1	1	1
Timer A Data Register	TADATA	205	BDH	1	1	1	1	1	1	1	1
Timer B Counter	TBCNT	206	BEH	0	0	0	0	0	0	0	0
Timer A Counter	TACNT	207	BFH	0	0	0	0	0	0	0	0
A/D Converter Control Register	ADCON	208	D0H	–	–	0	0	0	0	0	0
A/D Converter Data Register (high byte)	ADDATAH	209	D1H	X	X	X	X	X	X	X	X
A/D Converter Data Register (low byte)	ADDATAL	210	D2H	–	–	–	–	–	–	X	X
Oscillator Control Register	OSCCON	211	D3H	–	–	–	–	0	0	–	0
System Clock Control Register	CLKCON	212	D4H	0	0	0	0	0	0	0	0
System Flags Register	FLAGS	213	D5H	X	X	X	X	–	–	–	–
Interrupt Pending Register 1	INTPND1	214	D6H	0	0	0	0	0	0	0	0
Interrupt Pending Register 2	INTPND2	215	D7H	–	–	0	0	0	0	0	0
LCD Port Control Register	LOPT	216	D8H	–	0	0	0	0	0	0	0
Stack Pointer	SP	217	D9H	X	X	X	X	X	X	X	X
Watch Timer Control Register	WTCON	218	DAH	0	0	0	0	0	0	0	–
Locations DBH is not mapped.											
Basic Timer Control Register	BTCON	220	DCH	0	0	0	0	0	0	0	0
Basic Timer Counter	BTCNT	221	DDH	0	0	0	0	0	0	0	0
Locations DEH is not mapped.											

Table 8-1. Register Values after RESET (Continued)

Register Name	Mnemonic	Address		Bit Values after RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
System Mode Register	SYM	223	DFH	–	–	–	–	0	0	0	0
STOP Control Register	STPCON	224	E0H	0	0	0	0	0	0	0	0
SIO Control Register	SIOCON	225	E1H	0	0	0	0	0	0	0	–
SIO Data Register	SIODATA	226	E2H	0	0	0	0	0	0	0	0
SIO Prescaler Register	SIOPS	227	E3H	0	0	0	0	0	0	0	0
Port 0 Data Register	P0	228	E4H	0	0	0	0	0	0	0	0
Port 1 Data Register	P1	229	E5H	0	0	0	0	0	0	0	0
Port 2 Data Register	P2	230	E6H	0	0	0	0	0	0	0	0
Port 3 Data Register	P3	231	E7H	0	0	0	0	0	0	0	0
Port 4 Data Register	P4	232	E8H	0	0	0	0	0	0	0	0
Port 5 Data Register	P5	233	E9H	0	0	0	0	0	0	0	0
Port 6 Data Register	P6	234	EAH	0	0	0	0	0	0	0	0
Port 0 Control Register	P0CON	235	EBH	0	0	0	0	0	0	0	0
Port 0 Pull-up Resistors Enable Register	P0PUR	236	ECH	–	–	–	–	0	0	0	0
Port 0 Interrupt Control Register	P0INT	237	EDH	–	–	–	–	0	0	0	0
Port 0 Interrupt Edge Selection Register	P0EDGE	238	EEH	–	–	–	–	0	0	0	0
Port 1 Control Register	P1CON	239	EFH	0	0	0	0	0	0	0	0
Port 1 Pull-up Resistors Enable Register	P1PUR	240	F0H	–	–	–	–	0	0	0	0
Port 1 Interrupt Control Register	P1INT	241	F1H	–	–	–	–	0	0	0	0
Port 1 Interrupt Edge Selection Register	P1EDGE	242	F2H	–	–	–	–	0	0	0	0
Port 2 Control Register	P2CON	243	F3H	0	0	0	0	0	0	0	0
Port 2 Pull-up Resistors Enable Register	P2PUR	244	F4H	–	–	–	–	0	0	0	0
Port 3 Control Register	P3CON	245	F5H	–	–	–	–	0	0	0	0
Port 3 Pull-up Resistors Enable Register	P3PUR	246	F6H	–	–	–	–	–	–	0	0
Port 3 Interrupt Control Register	P3INT	247	F7H	–	–	–	–	–	–	0	0
Port 3 Interrupt Edge Selection Register	P3EDGE	248	F8H	–	–	–	–	–	–	0	0
Port 4 Control Register (High Byte)	P4CONH	249	F9H	0	0	0	0	0	0	0	0
Port 4 Control Register (High Byte)	P4CONL	250	FAH	0	0	0	0	0	0	0	0
Port 5 Control Register (High Byte)	P5CONH	251	FBH	0	0	0	0	0	0	0	0
Port 5 Control Register (High Byte)	P5CONL	252	FCH	0	0	0	0	0	0	0	0
Port 6 Control Register	P6CON	253	FDH	0	0	0	0	0	0	0	0
LCD Mode Register	LMOD	254	FEH	–	0	0	0	0	0	0	0
Location FFH is not mapped.											

NOTES

9 I/O PORTS

OVERVIEW

The S3C9228/P9228 microcontroller has seven bit-programmable I/O ports, P0-P6. Port 0 is 6-bit port, port 1, port 2, and port 6 are 4-bit ports, port 3 is 2-bit port, and port 4 and port 5 are 8-bit ports. This gives a total of 36 I/O pins. Each port can be flexibly configured to meet application design requirements.

The CPU accesses ports by directly writing or reading port registers. No special I/O instructions are required. All ports of the S3C9228/P9228 except P0.4 and P0.5 can be configured to input or output mode. All LCD signal pins are shared with normal I/O ports.

Table 9-1 gives you a general overview of S3C9228 I/O port functions.

Table 9-1. S3C9228 Port Configuration Overview

Port	Configuration Options
0	1-bit programmable I/O port except P0.4 and P0.5. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups. The P0.4 and P0.5 are only push-pull output ports. Alternatively P0.0-P0.3 can be used as input for external interrupts INT and can be used as TAOUT, T1CLK, and BUZ.
1	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups. Alternatively P1 can be used as input for external interrupts INT and can be used as AD0-AD3.
2	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups. Alternatively P2.0 and P2.1 can be used as outputs for LCD segment signals and P2.0-P2.2 can be used as SCK, SO, and SI.
3	1-bit programmable I/O port. Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups. Alternatively P3 can be used as input for external interrupts INTP and can be used as outputs for LCD segment signals.
4	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups. Alternatively P4 can be used as outputs for LCD segment signals.
5	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups. Alternatively P5.0-P5.3 can be used as outputs for LCD segment signals and P5.4-P5.7 can be used as outputs for LCD common or segment signals.
6	1-bit programmable I/O port. Input or push-pull, open-drain output and software assignable pull-ups. Alternatively P6 can be used as outputs for LCD common signals.

PORT DATA REGISTERS

Table 9-2 gives you an overview of the register locations of all seven S3C9228 I/O port data registers. Data registers for ports 1, 2, 3, 4, 5, and 6 have the general format shown in Figure 9-1.

Table 9-2. Port Data Register Summary

Register Name	Mnemonic	Decimal	Hex	R/W
Port 0 data register	P0	228	E4H	R/W
Port 1 data register	P1	229	E5H	R/W
Port 2 data register	P2	230	E6H	R/W
Port 3 data register	P3	231	E7H	R/W
Port 4 data register	P4	232	E8H	R/W
Port 5 data register	P5	233	E9H	R/W
Port 6 data register	P6	234	EAH	R/W

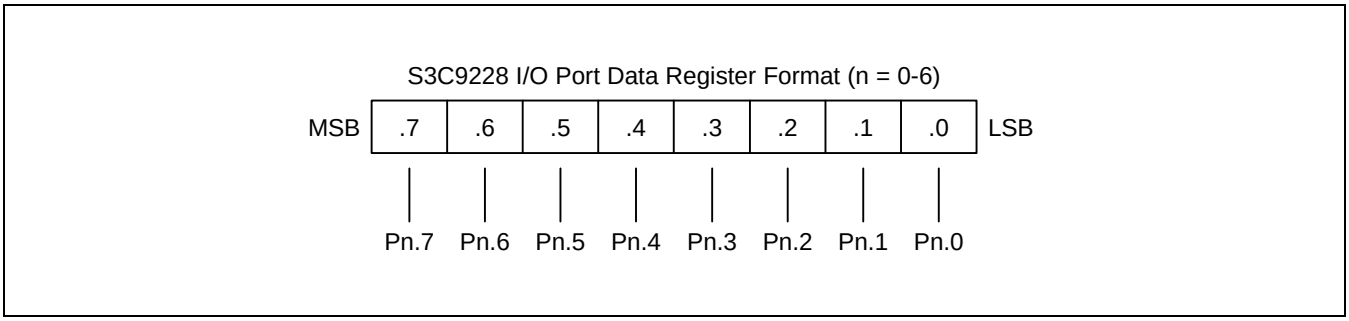


Figure 9-1. S3C9228 I/O Port Data Register Format

PORT 0

Port 0 is an 6-bit I/O port with individually configurable pins. Port 0 pins are accessed directly by writing or reading the port 0 data register, P0 at location E4H in page 0. P0.0-P0.3 can serve as inputs (with or without pull-up), as outputs (push-pull or open-drain) or you can be configured the following functions.

— Low-nibble pins (P0.0-P0.3): TAOUT,T1CLK, BUZ, INT

— High-nibble pins (P0.4-P0.5): push-pull output ports (only 44-QFP package)

Port 0 Control register (P0CON)

Port 0 has a 8-bit control register: P0CON for P0.0-P0.3. A reset clears the P0CON register to “00H”, configuring pins to input mode. You use control register setting to select input or output mode (push-pull or open-drain) and enable the alternative functions.

When programming this port, please remember that any alternative peripheral I/O function you configure using the port 0 control register must also be enabled in the associated peripheral module.

Port 0 Pull-up Resistor Control Register (P0PUR)

Using the port 0 pull-up resistor control register, P0PUR (ECH, page 0), you can configure pull-up resistors to individual port 0 pins.

Port 0 Interrupt Enable, Pending, and Edge Selection Registers (P0INT, INTPND1.3-.0, P0EDGE)

To process external interrupts at the port 0 pins, three additional control registers are provided: the port 0 interrupt enable register P0INT (EDH, page 0), the port 0 interrupt pending bits INTPND1.3-.0 (D6H, page 0), and the port 0 interrupt edge selection register P0EDGE (EEH, page 0).

The port 0 interrupt pending register bits lets you check for interrupt pending conditions and clear the pending condition when the interrupt service routine has been initiated. The application program detects interrupt requests by polling the INTPND1.3-.0 register at regular intervals.

When the interrupt enable bit of any port 0 pin is "1", a rising or falling edge at that pin will generate an interrupt request. The corresponding INTPND1 bit is then automatically set to "1" and the IRQ level goes low to signal the CPU that an interrupt request is waiting. When the CPU acknowledges the interrupt request, application software must the clear the pending condition by writing a "0" to the corresponding INTPND1 bit.

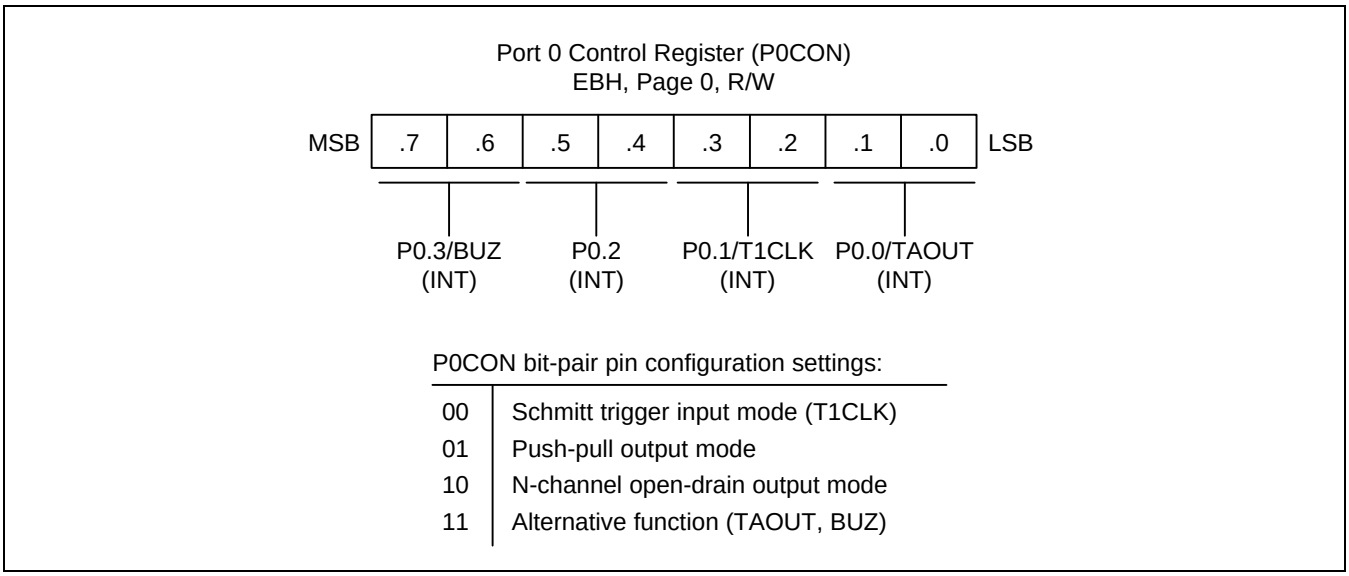


Figure 9-2. Port 0 Control Register (P0CON)

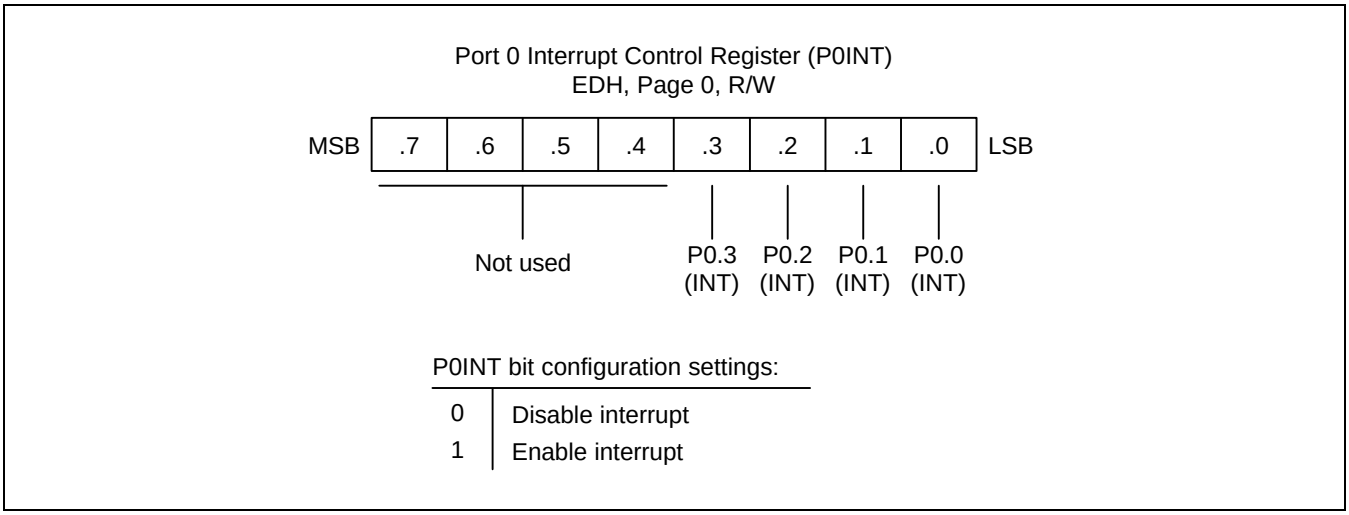


Figure 9-3. Port 0 Interrupt Control Register (P0INT)

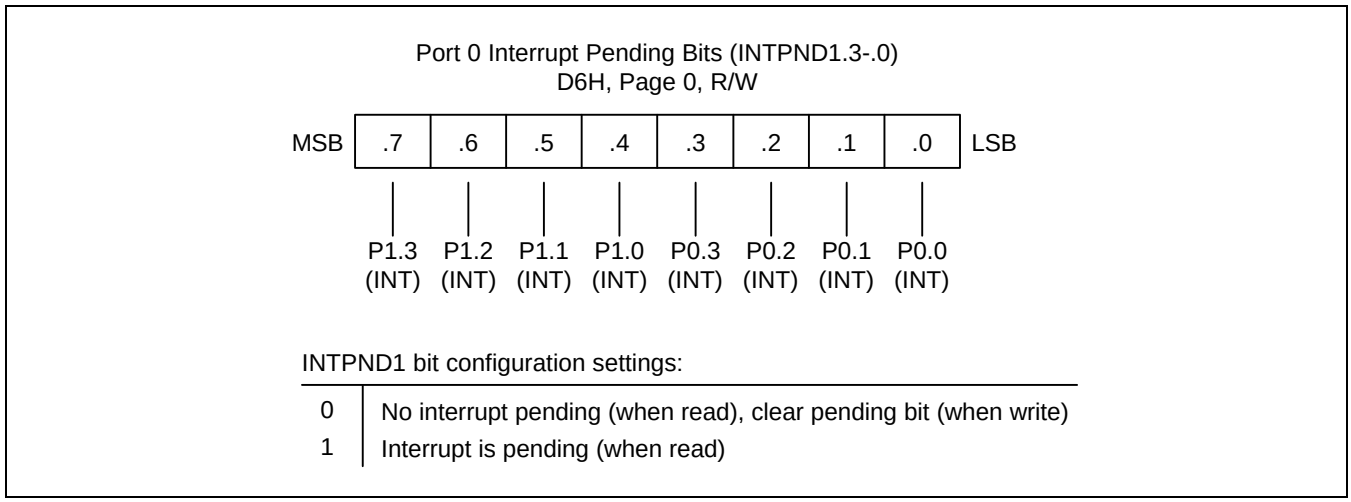


Figure 9-4. Port 0 Interrupt Pending Bits (INTPND1.3-.0)

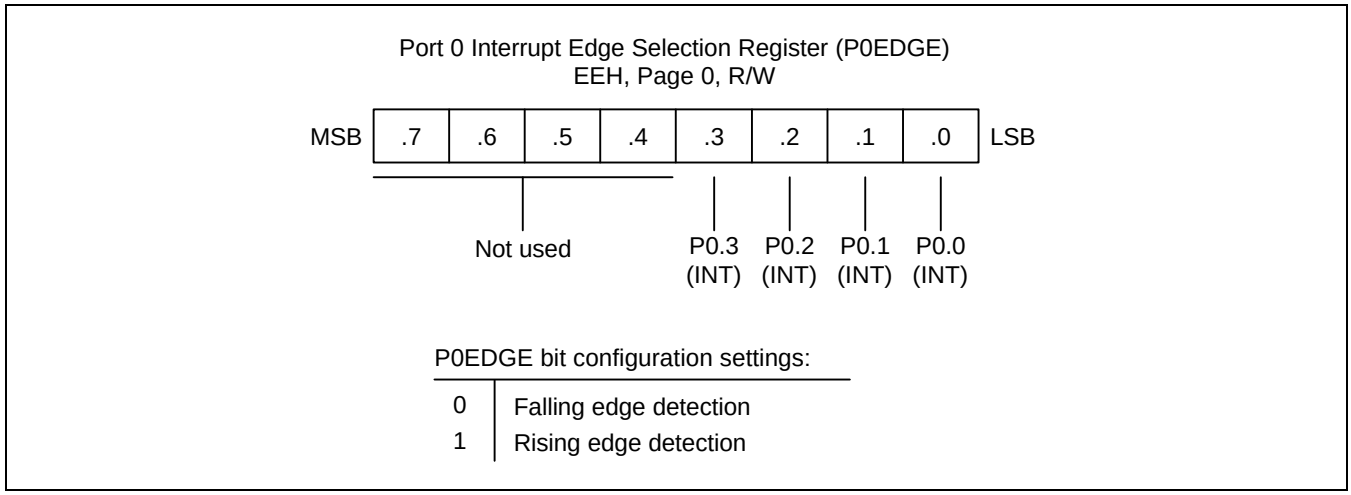


Figure 9-5. Port 0 Interrupt Edge Selection Register (P0EDGE)

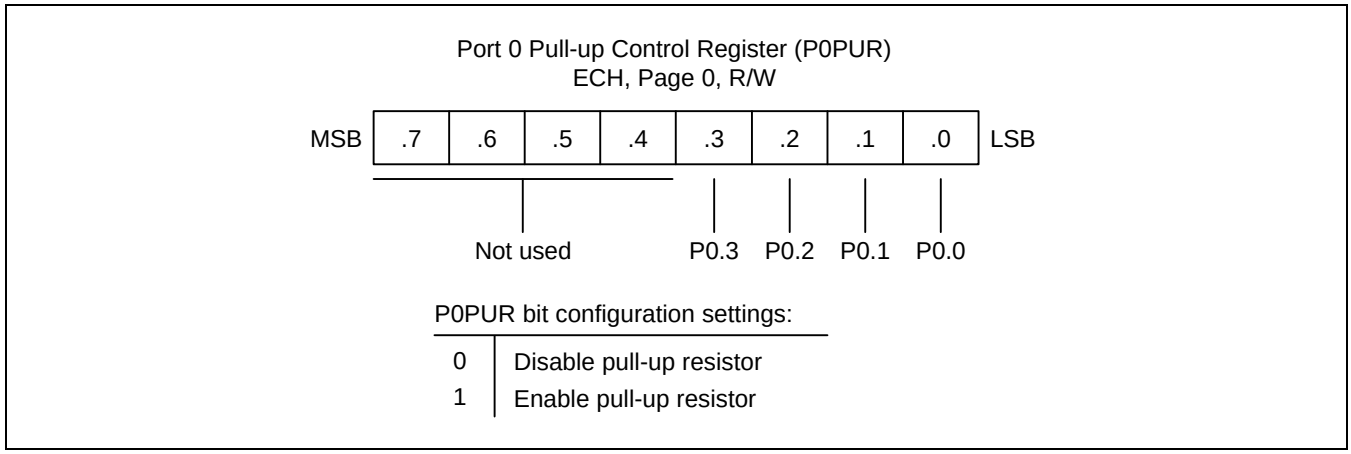


Figure 9-6. Port 0 Pull-up Control Register (P0PUR)

PORT 1

Port 1 is an 4-bit I/O port with individually configurable pins. Port 1 pins are accessed directly by writing or reading the port 1 data register, P1 at location E5H in page 0. P1.0-P1.3 can serve as inputs (with or without pull-up), as outputs (push-pull or open-drain) or you can be configured the following functions.

— Low-nibble pins (P1.0-P1.3): AD0-AD3, INT

Port 1 Control Register (P1CON)

Port 1 has a 8-bit control register: P1CON for P1.0-P1.3. A reset clears the P1CON register to "00H", configuring pins to input mode. You use control register setting to select input or output mode (push-pull or open-drain) and enable the alternative functions.

When programming this port, please remember that any alternative peripheral I/O function you configure using the port 1 control register must also be enabled in the associated peripheral module.

Port 1 Pull-up Resistor Control Register (P1PUR)

Using the port 1 pull-up resistor control register, P1PUR (F0H, page 0), you can configure pull-up resistors to individual port 1 pins.

Port 1 Interrupt Enable, Pending, and Edge Selection Registers (P1INT, INTPND1.7-.4, P1EDGE)

To process external interrupts at the port 1 pins, three additional control registers are provided: the port 1 interrupt enable register P1INT (F1H, page 0), the port 1 interrupt pending bits INTPND1.7-.4 (D6H, page 0), and the port 1 interrupt edge selection register P1EDGE (F2H, page 0).

The port 1 interrupt pending register bits lets you check for interrupt pending conditions and clear the pending condition when the interrupt service routine has been initiated. The application program detects interrupt requests by polling the INTPND1.7-.4 register at regular intervals.

When the interrupt enable bit of any port 1 pin is "1", a rising or falling edge at that pin will generate an interrupt request. The corresponding INTPND1 bit is then automatically set to "1" and the IRQ level goes low to signal the CPU that an interrupt request is waiting. When the CPU acknowledges the interrupt request, application software must the clear the pending condition by writing a "0" to the corresponding INTPND1 bit.

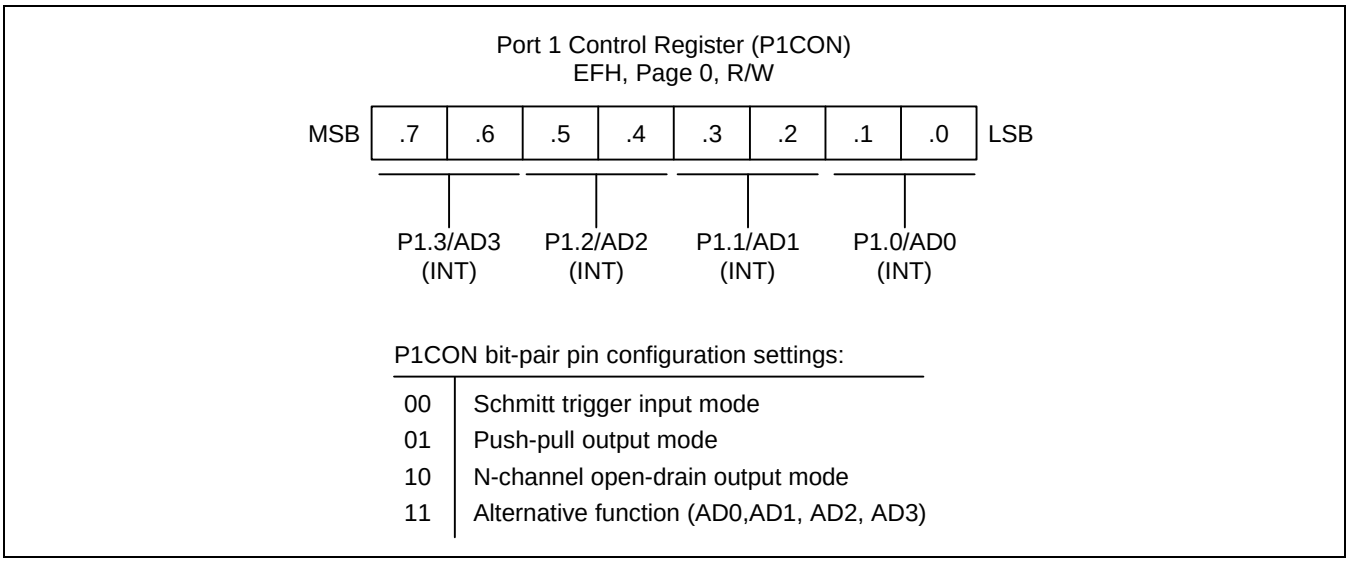


Figure 9-7. Port 1 Control Register (P1CON)

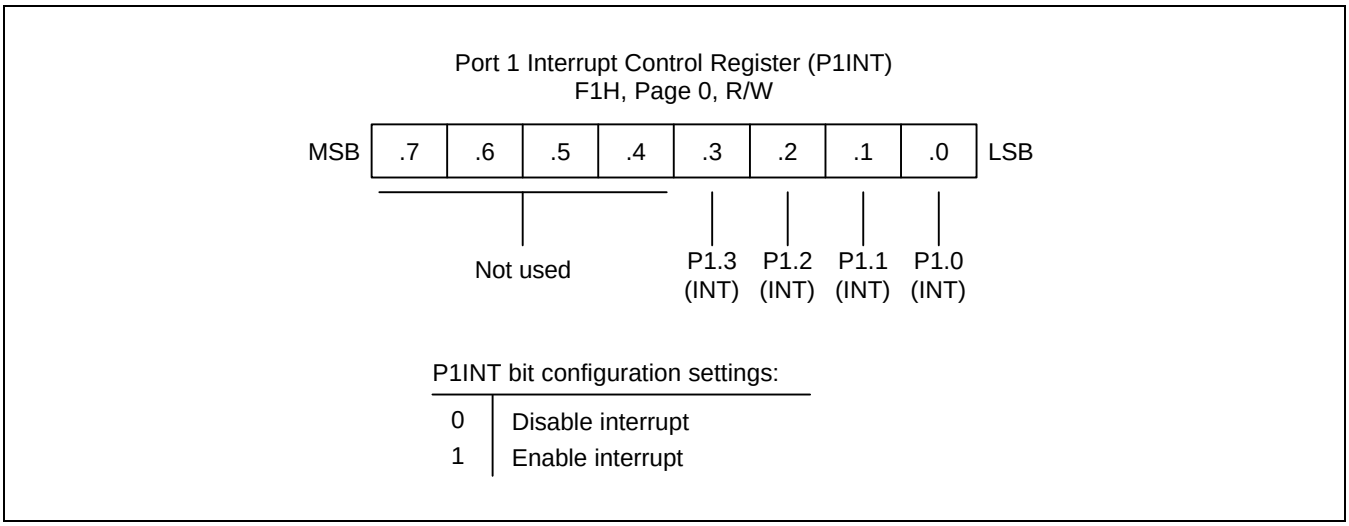


Figure 9-8. Port 1 Interrupt Control Register (P1INT)

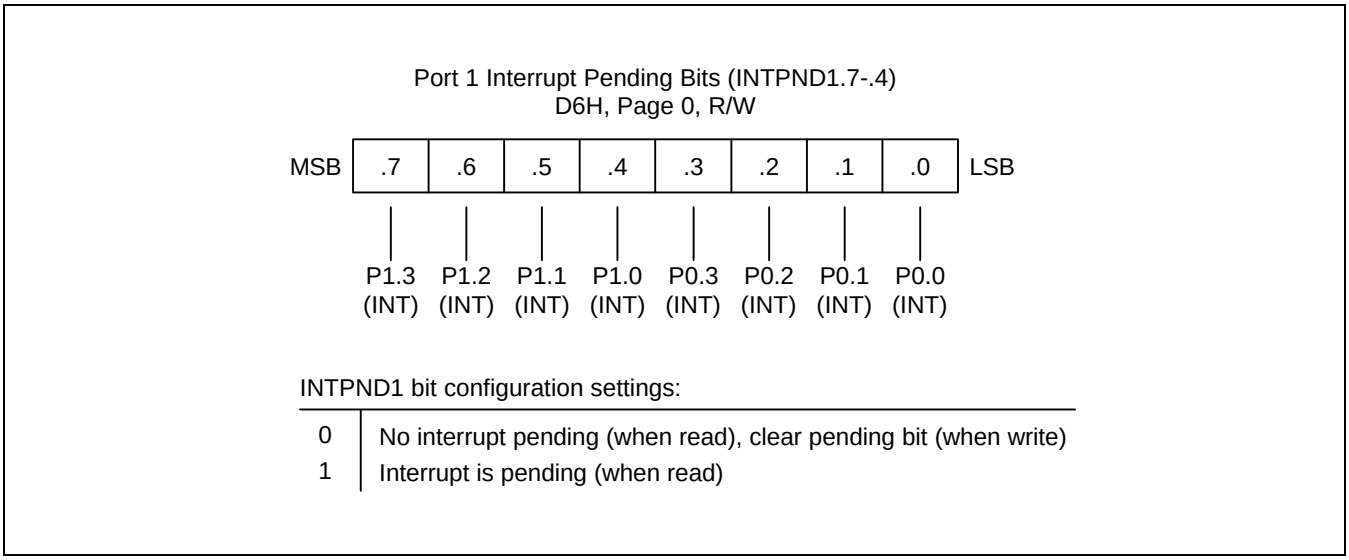


Figure 9-9. Port 1 Interrupt Pending Bits (INTPND1.7-.4)

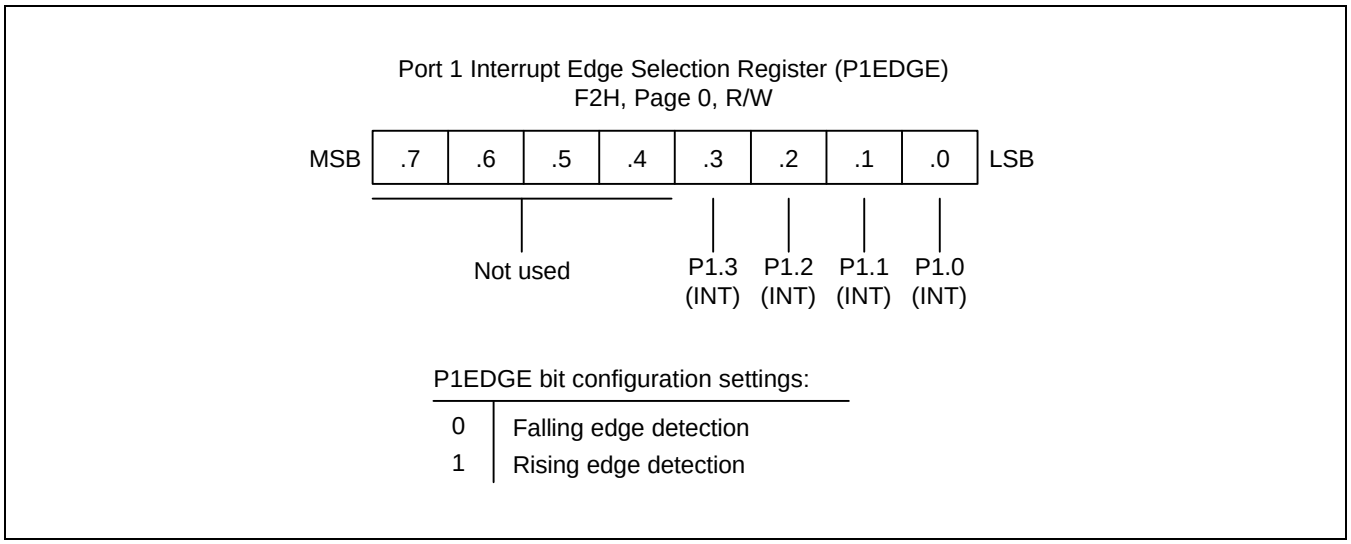


Figure 9-10. Port 1 Interrupt Edge Selection Register (P1EDGE)

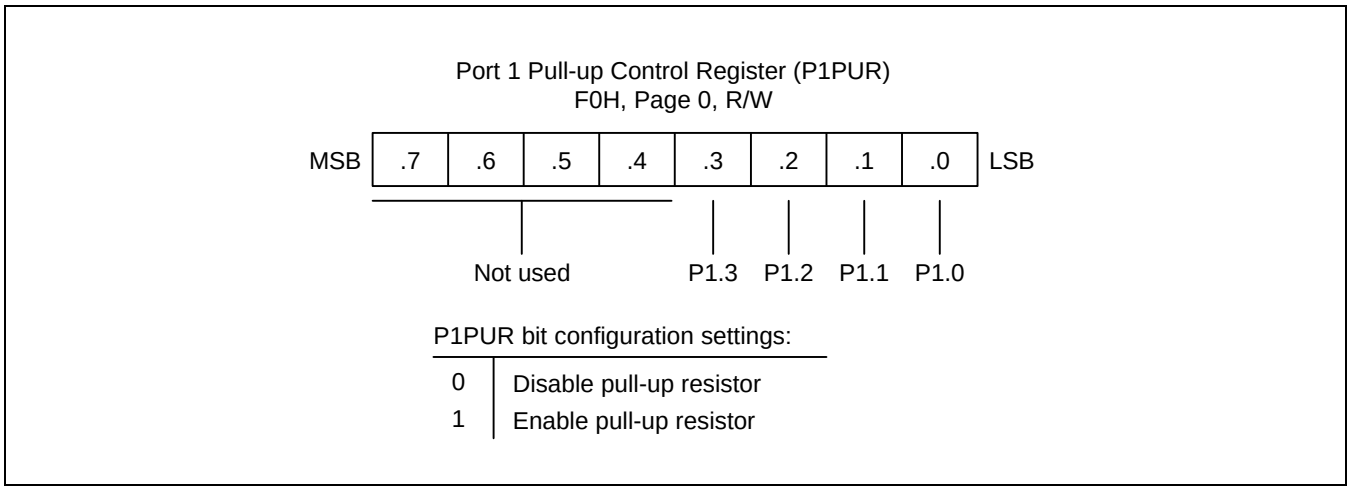


Figure 9-11. Port 1 Pull-up Control Register (P1PUR)

PORT 2

Port 2 is an 4-bit I/O port with individually configurable pins. Port 2 pins are accessed directly by writing or reading the port 2 data register, P2 at location E6H in page 0. P2.0-P2.3 can serve as inputs (with or without pull-up), as outputs (push-pull or open-drain) or you can be configured the following functions.

— Low-nibble pins (P2.0-P2.3): SCK, SO, SI, SEG0-SEG1

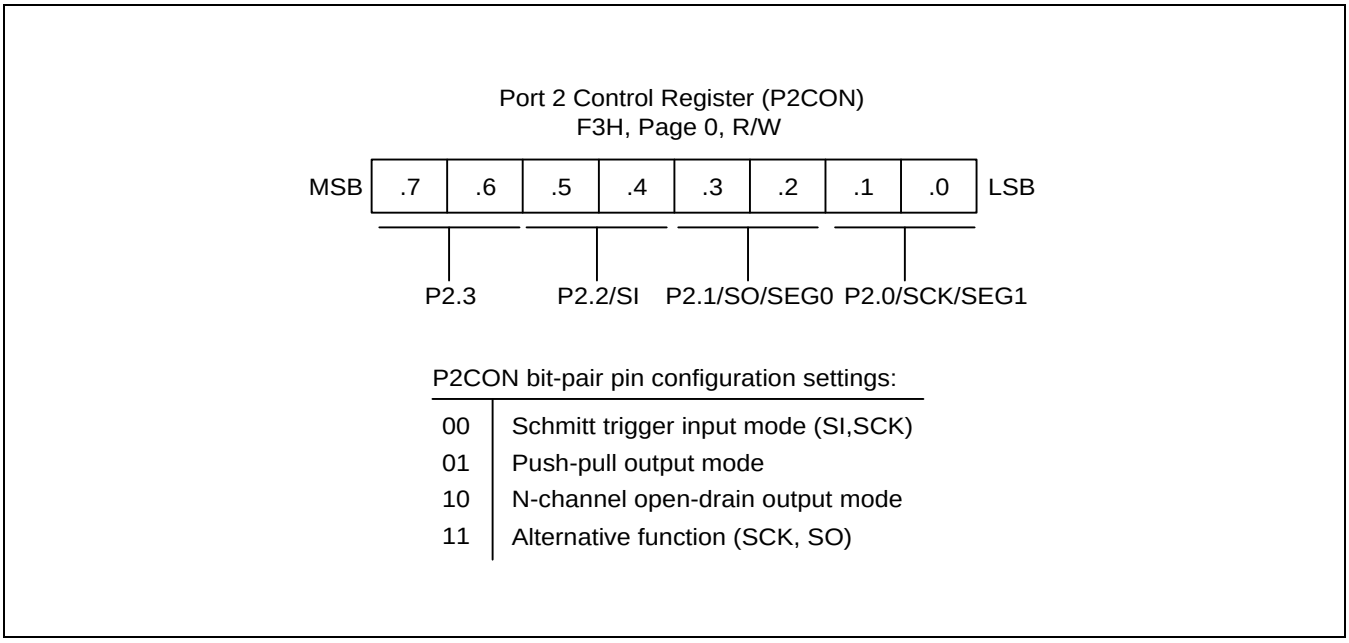
Port 2 Control Register (P2CON)

Port 2 has a 8-bit control register: P2CON for P2.0-P2.3. A reset clears the P2CON register to "00H", configuring pins to input mode. You use control register setting to select input or output mode (push-pull or open-drain) and enable the alternative functions.

When programming this port, please remember that any alternative peripheral I/O function you configure using the port 2 control register must also be enabled in the associated peripheral module.

Port 2 Pull-up Resistor Control Register (P2PUR)

Using the port 2 pull-up resistor control register, P2PUR (F4H, page 0), you can configure pull-up resistors to individual port 2 pins.



P2CON bit-pair pin configuration settings:

00	Schmitt trigger input mode (SI,SCK)
01	Push-pull output mode
10	N-channel open-drain output mode
11	Alternative function (SCK, SO)

Figure 9-12. Port 2 Control Register (P2CON)

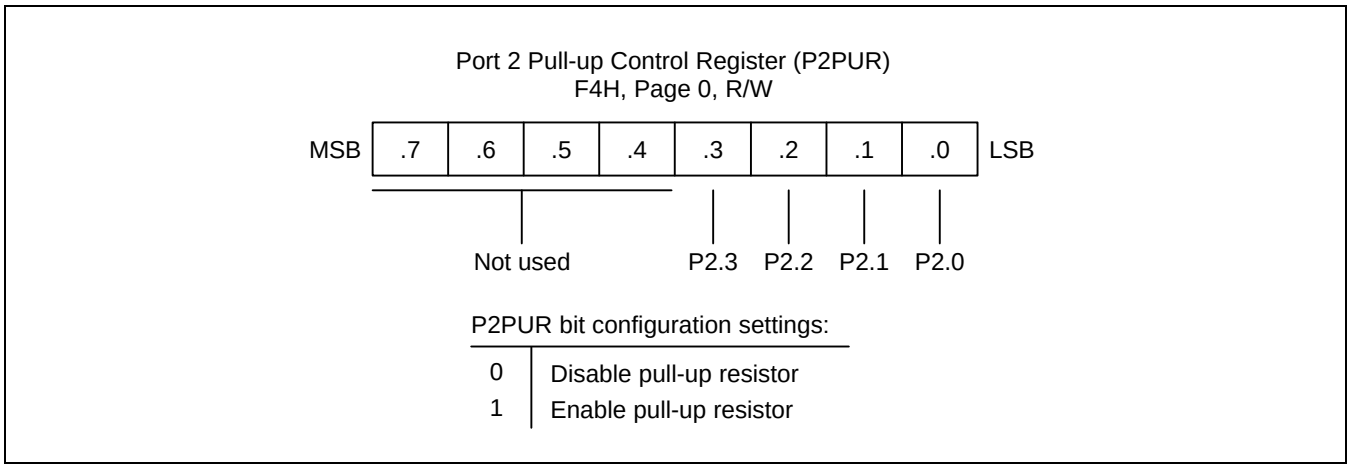


Figure 9-13. Port 2 Pull-up Control Register (P2PUR)

PORT 3

Port 3 is an 2-bit I/O port with individually configurable pins. Port 3 pins are accessed directly by writing or reading the port 3 data register, P3 at location E7H in page 0. P3.0-P3.1 can serve as inputs (with or without pull-up, and high impedance input), as outputs (push-pull or open-drain) or you can be configured the following functions.

— Low-nibble pins (P3.0-P3.1): SEG2-SEG3, INTP

Port 3 Control Register (P3CON)

Port 3 has a 8-bit control register: P3CON for P3.0-P3.1. A reset clears the P3CON register to "00H", configuring pins to input mode. You use control register setting to select input or output mode (push-pull or open-drain).

Port 3 Pull-up Resistor Control Register (P3PUR)

Using the port 3 pull-up resistor control register, P3PUR (F6H, page 0), you can configure pull-up resistors to individually port 3 pins.

Port 3 Interrupt Enable, Pending, and Edge Selection Registers(P3INT, INTPND2.5-.4, P3EDGE)

To process external interrupts at the port 3 pins, three additional control registers are provided: the port 3 interrupt enable register P3INT (F7H, page 0), the port 3 interrupt pending bits INTPND2.5-.4 (D7H, page 0), and the port 3 interrupt edge selection register P3EDGE (F8H, page 0).

The port 3 interrupt pending register bits lets you check for interrupt pending conditions and clear the pending condition when the interrupt service routine has been initiated. The application program detects interrupt requests by polling the INTPND2.5-.4 register at regular intervals.

When the interrupt enable bit of any port 3 pin is "1", a rising or falling edge at that pin will generate an interrupt request. The corresponding INTPND2 bit is then automatically set to "1" and the IRQ level goes low to signal the CPU that an interrupt request is waiting. When the CPU acknowledges the interrupt request, application software must the clear the pending condition by writing a "0" to the corresponding INTPND2 bit.

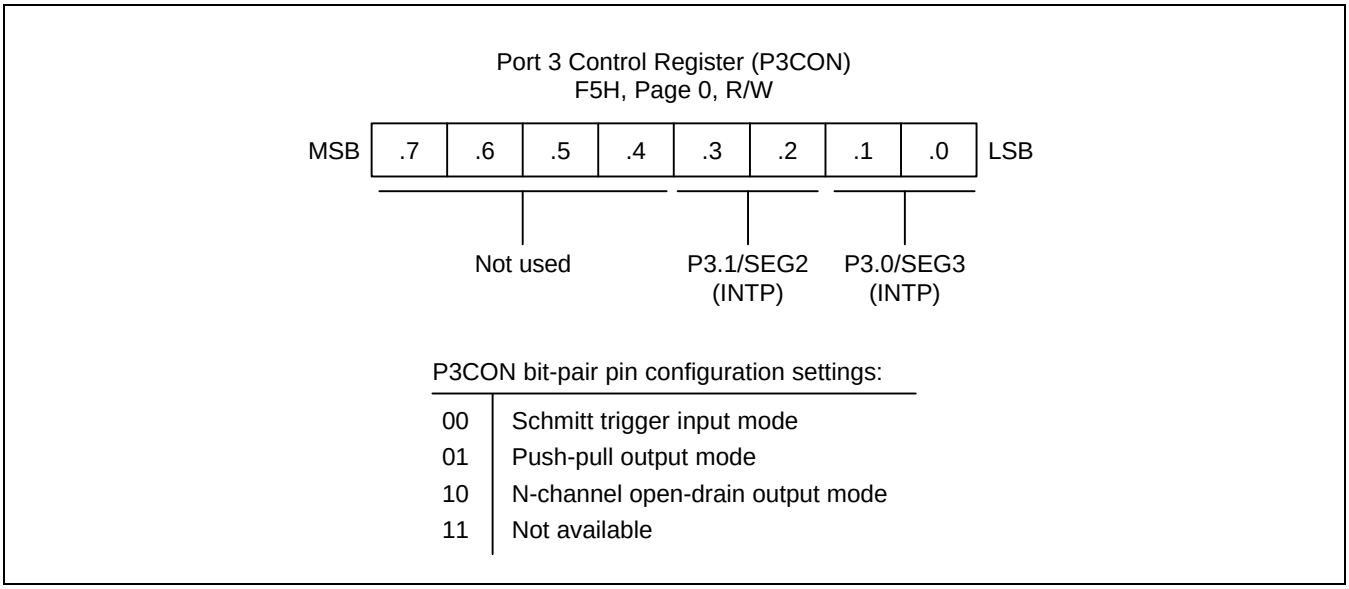


Figure 9-14. Port 3 Control Register (P3CON)

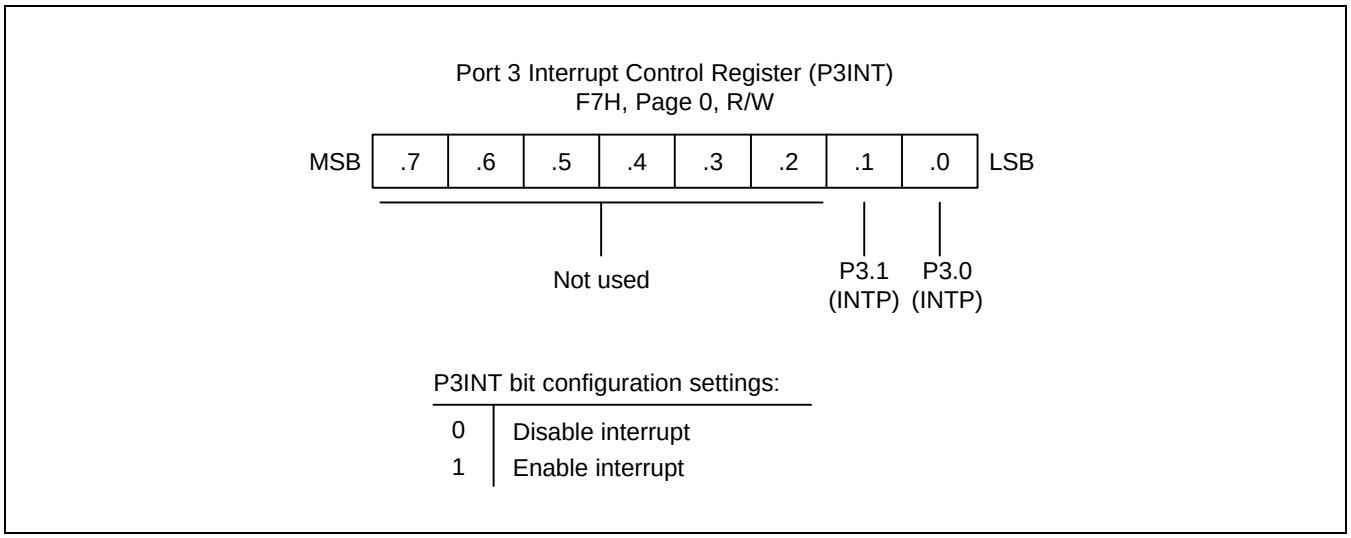


Figure 9-15. Port 3 Interrupt Control Register (P3INT)

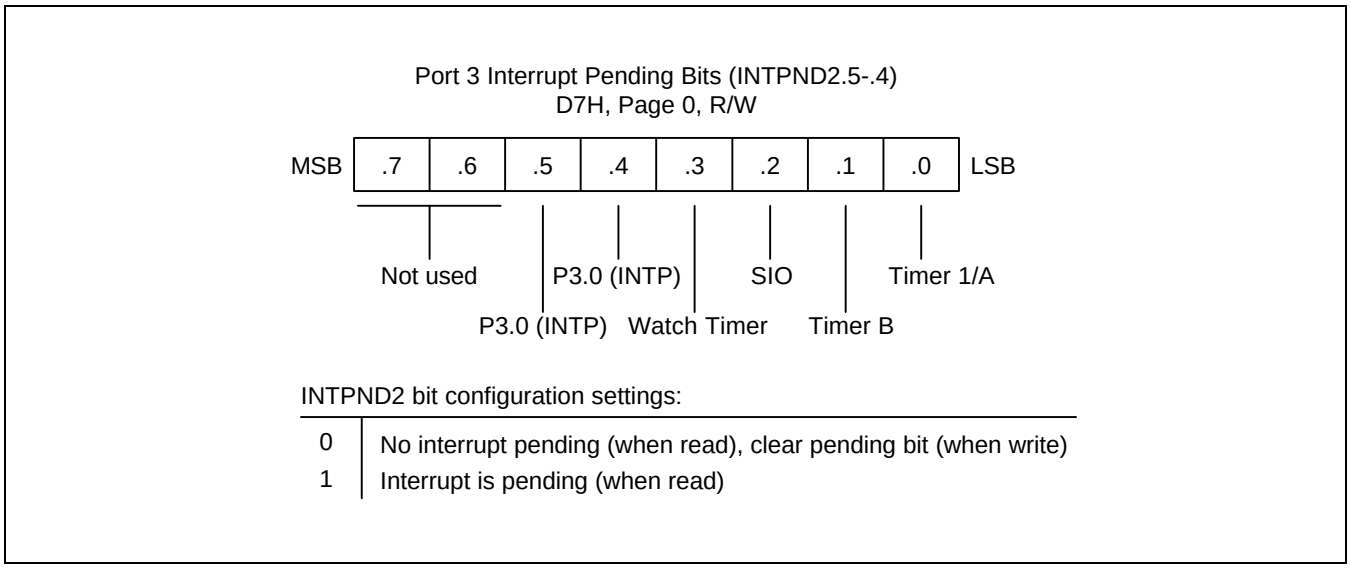


Figure 9-16. Port 3 Interrupt Pending Bits (INTPND2.5-.4)

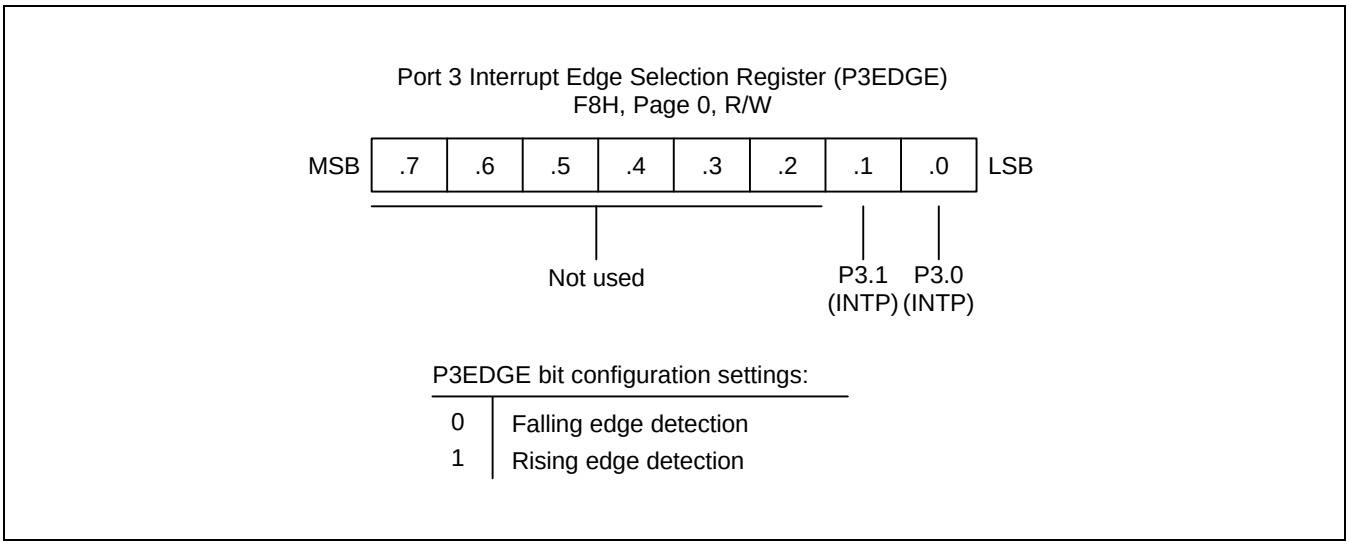


Figure 9-17. Port 3 Interrupt Edge Selection Register (P3EDGE)

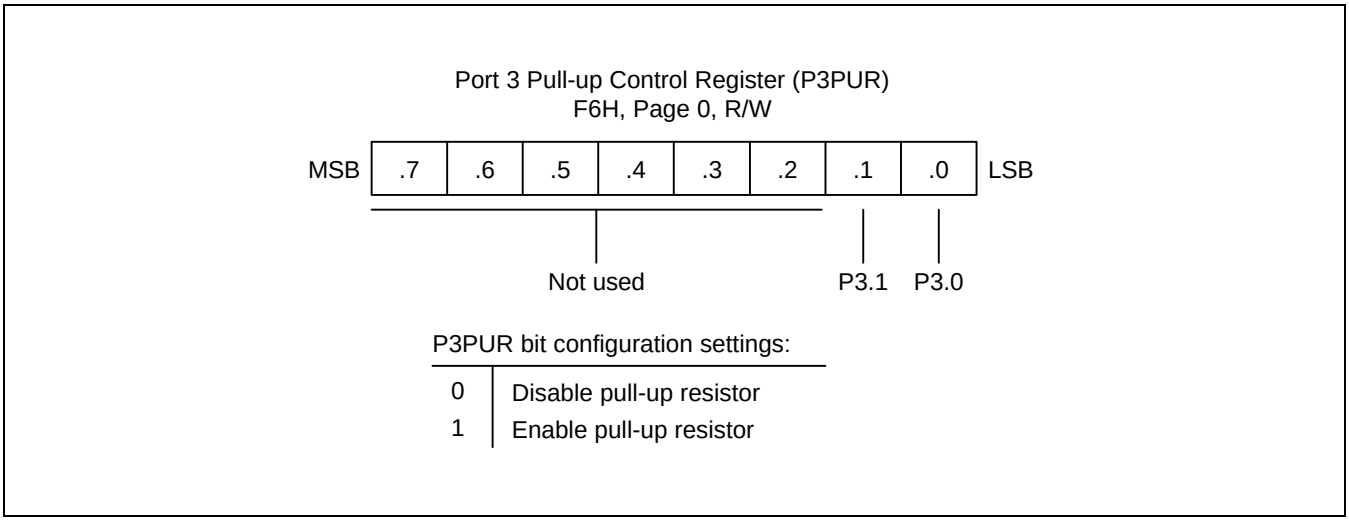


Figure 9-18. Port 3 Pull-up Control Register (P3PUR)

PORT 4

Port 4 is an 8-bit I/O port with individually configurable pins. Port 4 pins are accessed directly by writing or reading the port 4 data register, P4 at location E8H in page 0. P4.0-P4.7 can serve as inputs or as push-pull, open-drain outputs. You can configure the following alternative functions with LCD port control register, LPOT:

— Low-nibble pins (P4.0-P4.3): SEG4-SEG7

— High-nibble pins (P4.4-P4.7): SEG8-SEG11

Port 4 Control Registers (P4CONH, P4CONL)

Port 4 has two 8-bit control registers: P4CONH for P4.4-P4.7 and P4CONL for P4.0-P4.3. A reset clears the P4CONH and P4CONL registers to "00H", configuring all pins to input mode. You use control registers setting to select input or output mode.

Port 4 Control Register, High Byte (P4CONH)
F9H, Page 0, R/W

MSB	.7	.6	.5	.4	.3	.2	.1	.0	LSB

P4.7/SEG11 P4.6/SEG10 P4.5/SEG9 P4.4/SEG8

P4CONH bit-pair pin configuration settings:

00	Input mode
01	Push-pull output mode
10	N-channel open-drain output mode
11	Input mode with pull-up

Figure 9-19. Port 4 High-Byte Control Register (P4CONH)

Port 4 Control Register, Low Byte (P4CONL)
FAH, Page 0, R/W

MSB	.7	.6	.5	.4	.3	.2	.1	.0	LSB

P4.3/SEG7 P4.2/SEG6 P4.1/SEG5 P4.0/SEG4

P4CONL bit-pair pin configuration settings:

00	Input mode
01	Push-pull output mode
10	N-channel open-drain output mode
11	Input mode with pull-up

Figure 9-20. Port 4 Low-Byte Control Register (P4CONL)

PORT 5

Port 5 is an 8-bit I/O port with individually configurable pins. Port 5 pins are accessed directly by writing or reading the port 5 data register, P5 at location E9H in page 0. P5.0-P5.7 can serve as inputs or as push-pull, open-drain outputs. You can configure the following alternative functions with LCD port control register, LPOT:

— Low-nibble pins (P5.0-P5.3): SEG12-SEG15

— High-nibble pins (P5.4-P5.7): SEG16-SEG19, COM4-COM7

Port 5 Control Registers (P5CONH, P5CONL)

Port 5 has two 8-bit control registers: P5CONH for P5.4-P5.7 and P4CONL for P5.0-P5.3. A reset clears the P5CONH and P5CONL registers to "00H", configuring all pins to input mode. You use control registers setting to select input or output mode.

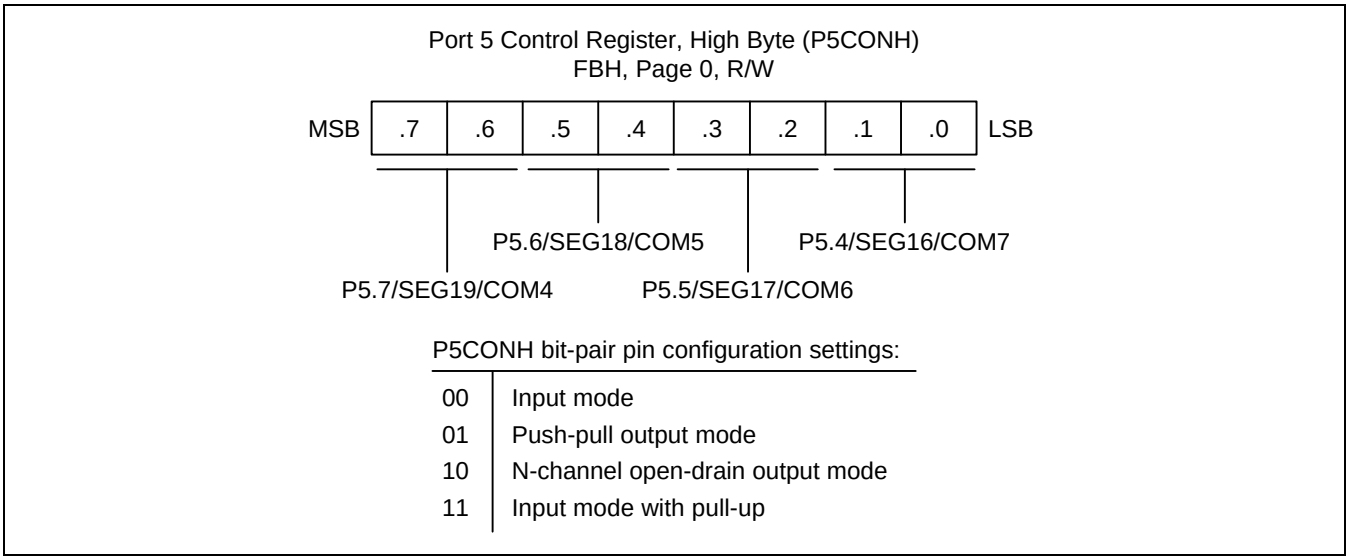


Figure 9-21. Port 5 High-Byte Control Register (P5CONH)

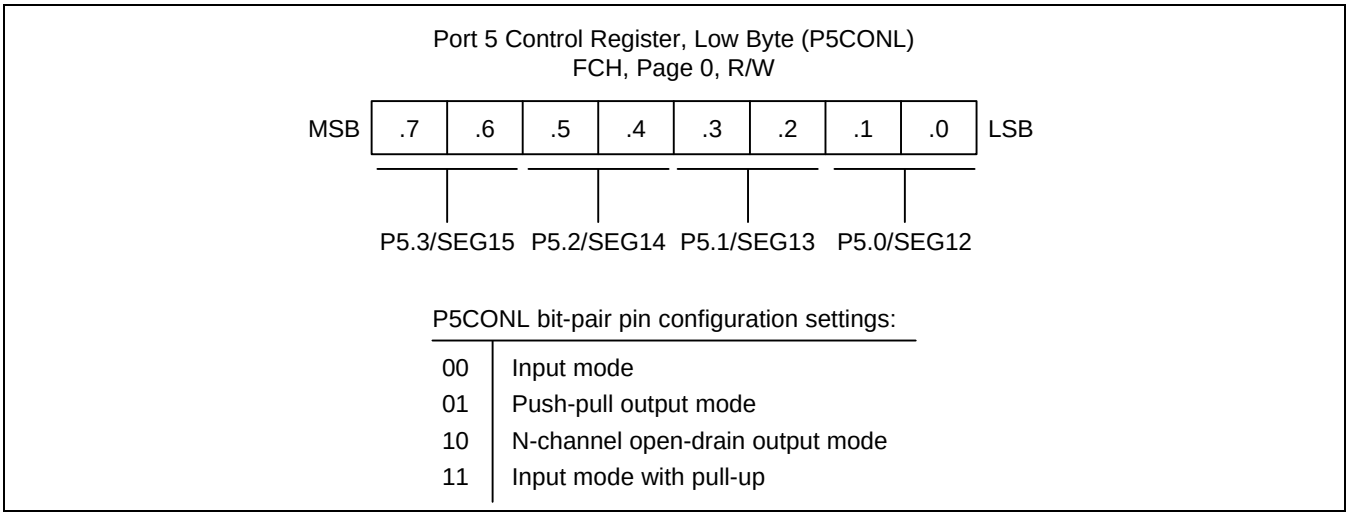


Figure 9-22. Port 5 Low-Byte Control Register (P5CONL)

PORT 6

Port 6 is an 4-bit I/O port with individually configurable pins. Port 6 pins are accessed directly by writing or reading the port 6 data register, P6 at location EAH in page 0. P6.0-P6.3 can serve as inputs or as push-pull, open-drain outputs. You can configure the following alternative functions with LCD port control register, LPOT:

— Low-nibble pins (P6.0-P6.3): COM0-COM3

Port 6 Control Register (P6CON)

Port 6 has a 8-bit control register: P6CONH for P6.0-P6.3. A reset clears the P6CON registers to "00H", configuring all pins to input mode. You use control registers setting to select input or output mode.

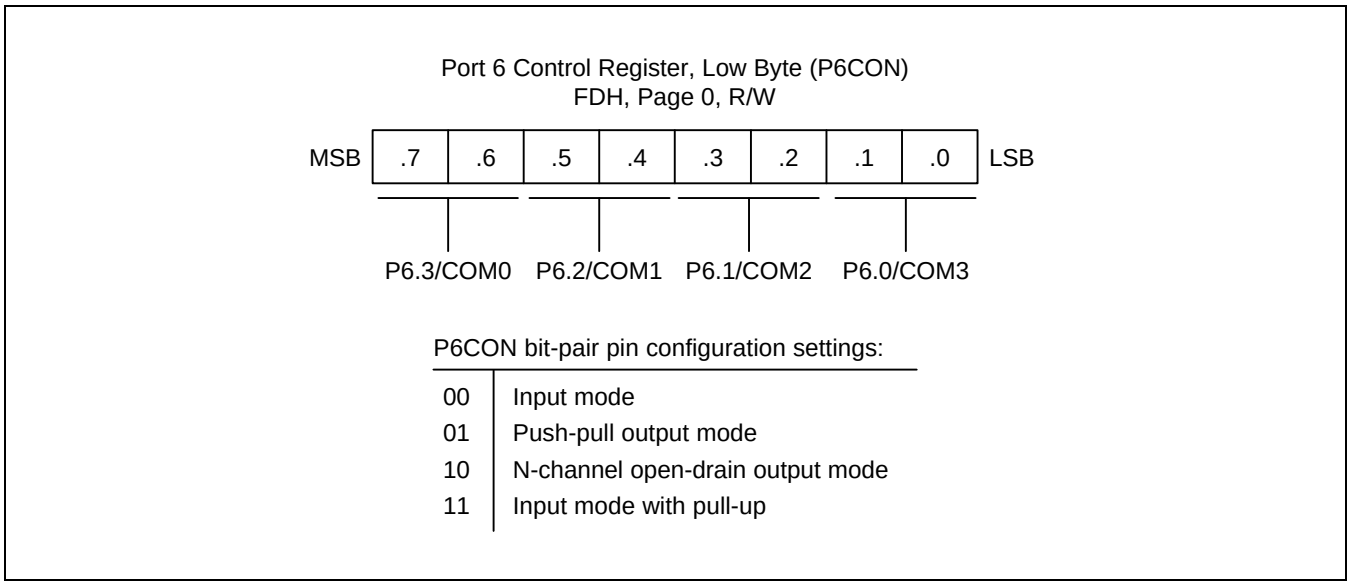


Figure 9-23. Port 6 Control Register (P6CON)

10 BASIC TIMER

OVERVIEW

Basic timer (BT) can be used 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 (fxx divided by 4096, 1024, 128, or 16) with multiplexer
— 8-bit basic timer counter, BTCNT (DDH, read-only)
— Basic timer control register, BTCON (DCH, read/write)

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 page 0, address DCH, 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 fxx/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 (page 0, DDH), can be cleared at any time during normal operation by writing a "1" to BTCON.1. To clear the frequency dividers for the basic timer input clock and timer counters, you write a "1" to BTCON.0.

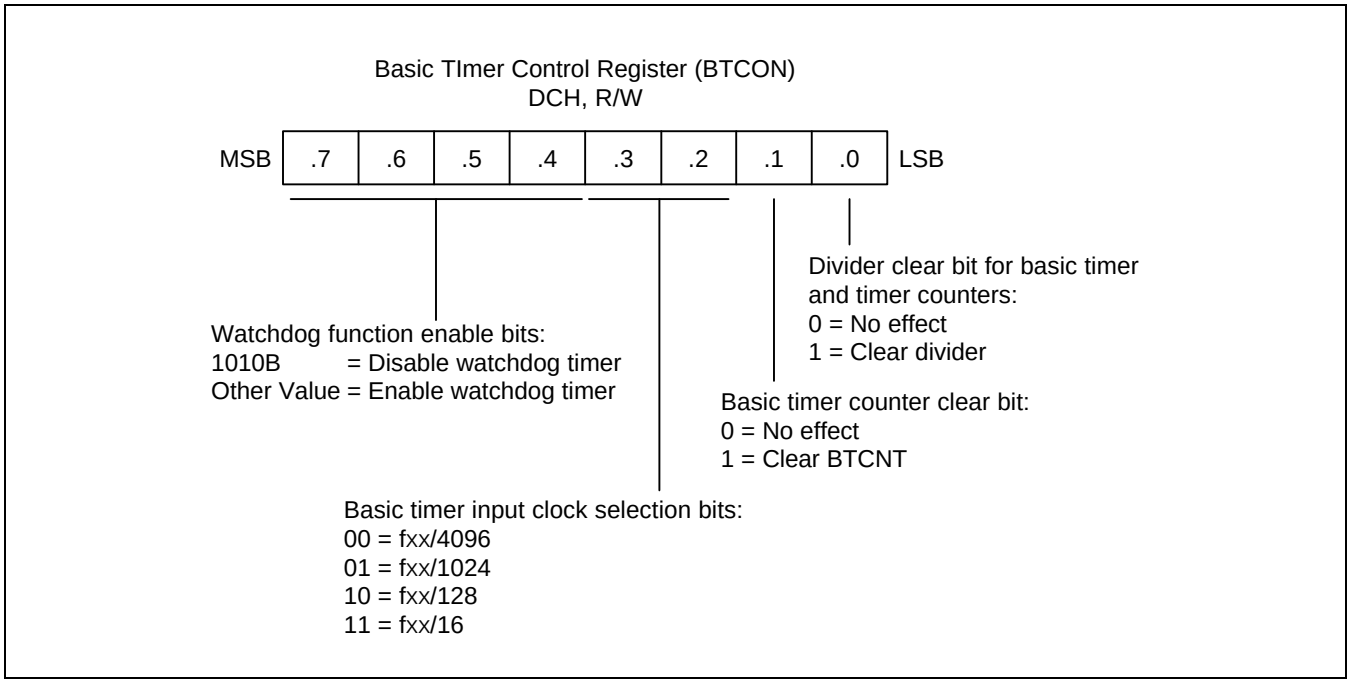


Figure 10-1. Basic Timer Control Register (BTCON)

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 BT clock.

A reset whenever a basic timer counter overflow occurs. During 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 BT counter clear operation will not be executed and a basic timer overflow will occur, initiating a reset. In other words, during 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.

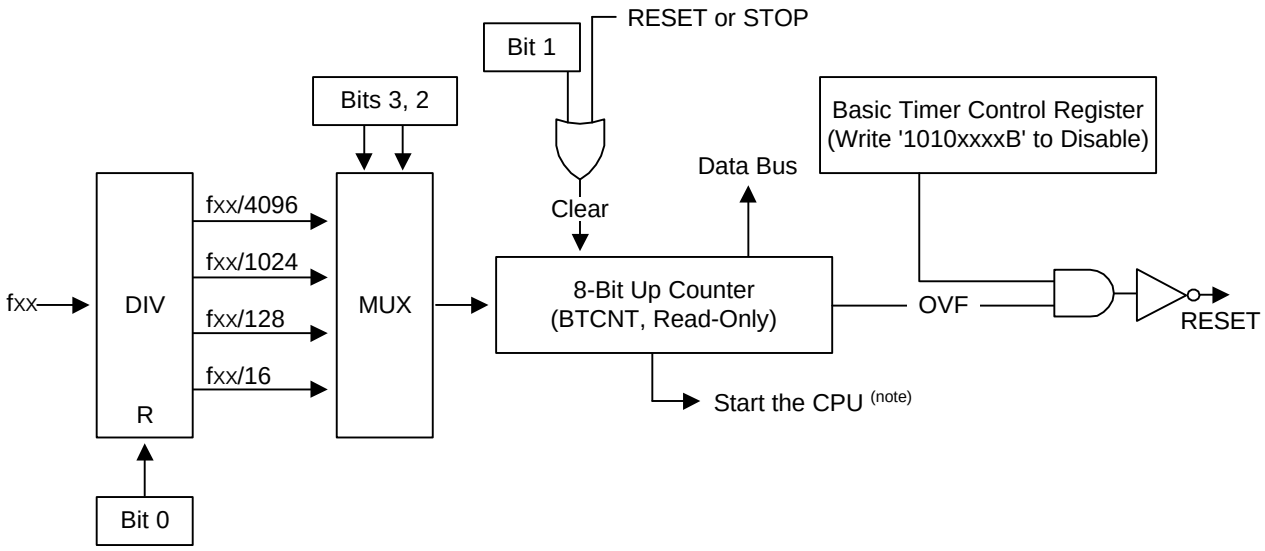
Oscillation Stabilization Interval Timer Function

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

In stop mode, whenever a reset or an internal and an external interrupt occurs, the oscillator starts. The BTCNT value then starts increasing at the rate of fxx/4096 (for reset), or at the rate of the preset clock source (for an internal and an external interrupt). When BTCNT.3 overflows, 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 internal and 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 fxx/4096. If an internal and 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 3 of the basic timer counter overflows.
4. When a BTCNT.3 overflow occurs, normal CPU operation resumes.



NOTE: During a power-on reset operation, the CPU is idle during the required oscillation stabilization interval (until bit 4 of the basic timer counter overflows).

Figure 10-2. Basic Timer Block Diagram

11 TIMER 1

ONE 16-BIT TIMER MODE (TIMER 1)

The 16-bit timer 1 is used in one 16-bit timer or two 8-bit timers mode. If TACON.7 is set to "1", timer 1 is used as a 16-bit timer. If TACON.7 is set to "0", timer 1 is used as two 8-bit timers.

— One 16-bit timer mode (Timer 1)

— Two 8-bit timers mode (Timer A and B)

OVERVIEW

The 16-bit timer 1 is an 16-bit general-purpose timer. Timer 1 has the interval timer mode by using the appropriate TACON setting.

Timer 1 has the following functional components:

— Clock frequency divider (fxx divided by 512, 256, 64, 8, or 1, fxt, and T1CLK: External clock) with multiplexer

— 16-bit counter (TACNT, TBCNT), 16-bit comparator, and 16-bit reference data register (TADATA, TBDATA)

— Timer 1 match interrupt generation

— Timer 1 control register, TACON (page 0, BBH, read/write)

FUNCTION DESCRIPTION

Interval Timer Function

The timer 1 module can generate an interrupt: the timer 1 match interrupt (T1INT).

The T1INT pending condition should be cleared by software when it has been serviced. Even though T1INT is disabled, the application's service routine can detect a pending condition of T1INT by the software and execute it's sub-routine. When this case is used, the T1INT pending bit must be cleared by the application sub-routine by writing a "0" to the INTPND2.0 pending bit.

In interval timer mode, a match signal is generated when the counter value is identical to the values written to the timer 1 reference data registers, TADATA and TBDATA. The match signal generates a timer 1 match interrupt and clears the counter.

If, for example, you write the value 32H and 10H to TADATA and TBDATA, respectively, and 8EH to TACON, the counter will increment until it reaches 3210H. At this point, the timer 1 interrupt request is generated, the counter value is reset, and counting resumes.

Timer 1 Control Register (TACON)

You use the timer 1 control register, TACON, to

— Enable the timer 1 operating (interval timer)

— Select the timer 1 input clock frequency

— Clear the timer 1 counter, TACNT and TBCNT

— Enable the timer 1 interrupt

TACON is located in page 0, at address BBH, and is read/write addressable using register addressing mode.

A reset clears TACON to "00H". This sets timer 1 to disable interval timer mode, selects an input clock frequency of fxx/512, and disables timer 1 interrupt. You can clear the timer 1 counter at any time during normal operation by writing a "1" to TACON.3.

To enable the timer 1 interrupt, you must write TACON.7, TACON.2, and TACON.1 to "1".
To generate the exact time interval, you should write TACON.3 and INTPND2.0, which cleared counter and interrupt pending bit. To detect an interrupt pending condition when T1INT is disabled, the application program polls pending bit, INTPND.2.0. When a "1" is detected, a timer 1 interrupt is pending. When the T1INT sub-routine has been serviced, the pending condition must be cleared by software by writing a "0" to the timer 1 interrupt pending bit, INTPND2.0.

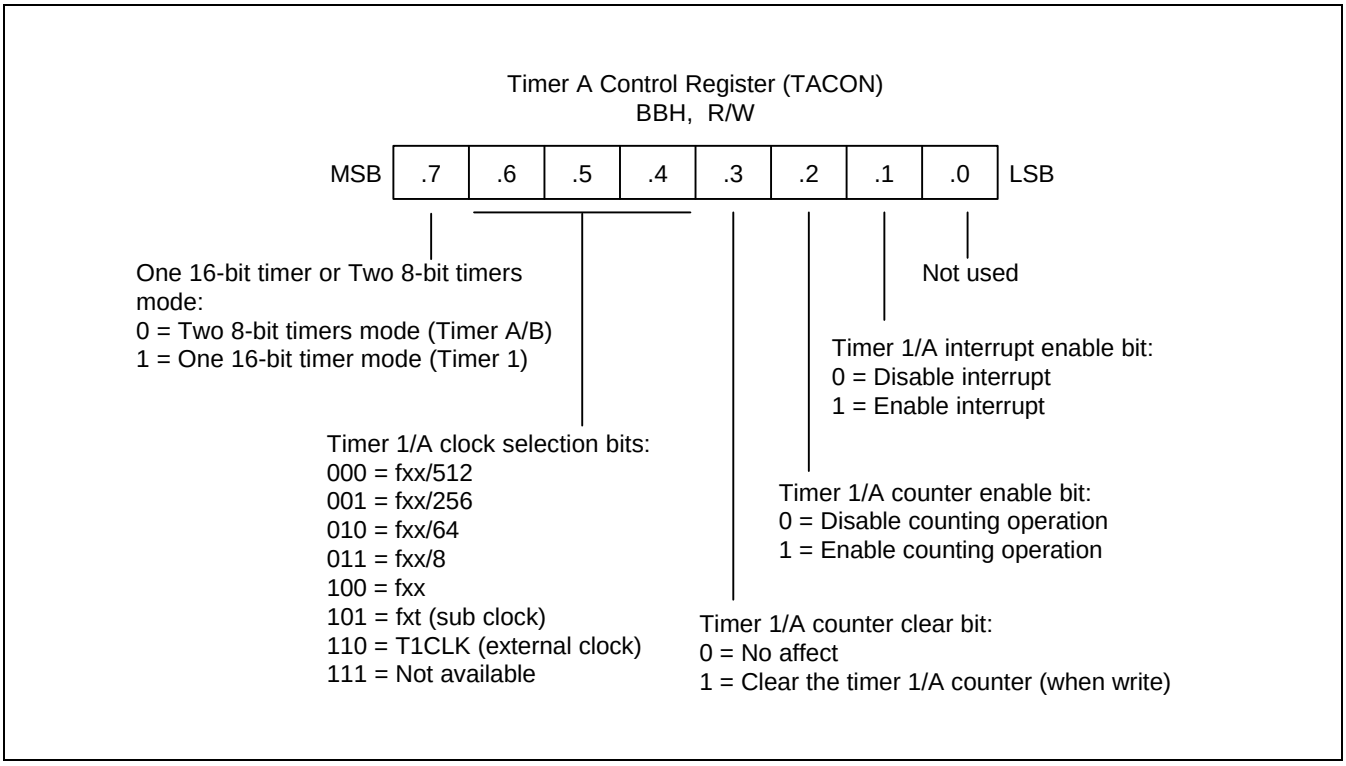


Figure 11-1. Timer 1 Control Register (TACON)

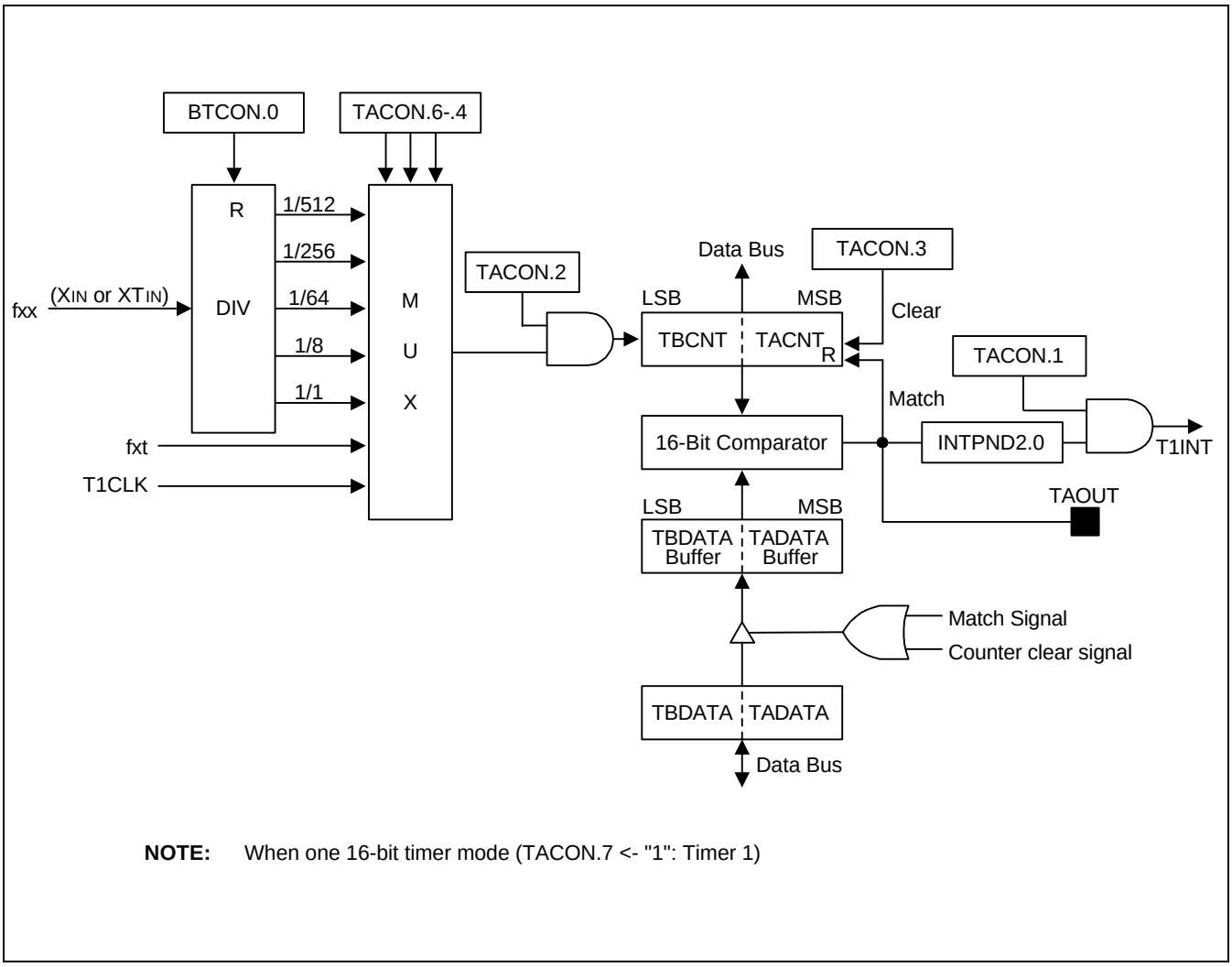


Figure 11-2. Timer 1 Block Diagram (One 16-bit Mode)

TWO 8-BIT TIMERS MODE (TIMER A and B)

OVERVIEW

The 8-bit timer A and B are the 8-bit general-purpose timers. Timer A and B have the interval timer mode by using the appropriate TACON and TBCON setting, respectively.

Timer A and B have the following functional components:

— Clock frequency divider with multiplexer
  – fxx divided by 512, 256, 64, 8 or 1, fxt, and T1CLK (External clock) for timer A
  – fxx divided by 512, 256, 64, 8 or 1, and fxt for timer B

— 8-bit counter (TACNT, TBCNT), 8-bit comparator, and 8-bit reference data register (TADATA, TBDATA)

— Timer A have I/O pin for match output (TAOUT)

— Timer A match interrupt generation

— Timer A control register, TACON (page 0, BBH, read/write)

— Timer B match interrupt generation

— Timer B control register, TBCON (page 0, BAH, read/write)

Timer A and B Control Register (TACON, TBCON)

You use the timer A and B control register, TACON and TBCON, to

— Enable the timer A (interval timer mode) and B operating (interval timer mode)

— Select the timer A and B input clock frequency

— Clear the timer A and B counter, TACNT and TBCNT

— Enable the timer A and B interrupt

TACON and TBCON are located in page 0, at address BBH and BAH, and is read/write addressable using register addressing mode.

A reset clears TACON to "00H". This sets timer A to disable interval timer mode, selects an input clock frequency of fxx/512, and disables timer A interrupt. You can clear the timer A counter at any time during normal operation by writing a "1" to TACON.3.

A reset clears TBCON to "00H". This sets timer B to disable interval timer mode, selects an input clock frequency of fxx/512, and disables timer A interrupt. You can clear the timer B counter at any time during normal operation by writing a "1" to TBCON.3.

To enable the timer A interrupt (TAINT) and timer B interrupt (TBINT), you must write TACON.7 to "0", TACON.2 (TBCON.2) and TACON.1 (TBCON.1) to "1". To generate the exact time interval, you should write TACON.3 (TBCON.3) and INTPND2.0 (INTPND2.1), which cleared counter and interrupt pending bit. To detect an interrupt pending condition when TAINT and TBINT is disabled, the application program polls pending bit, INTPND2.0 and INTPND2.1. When a "1" is detected, a timer A interrupt (TAINT) and timer B interrupt (TBINT) is pending. When the TAINT and TBINT sub-routine has been serviced, the pending condition must be cleared by software by writing a "0" to the timer A and B interrupt pending bit, INTPND2.0 and INTPND2.1.

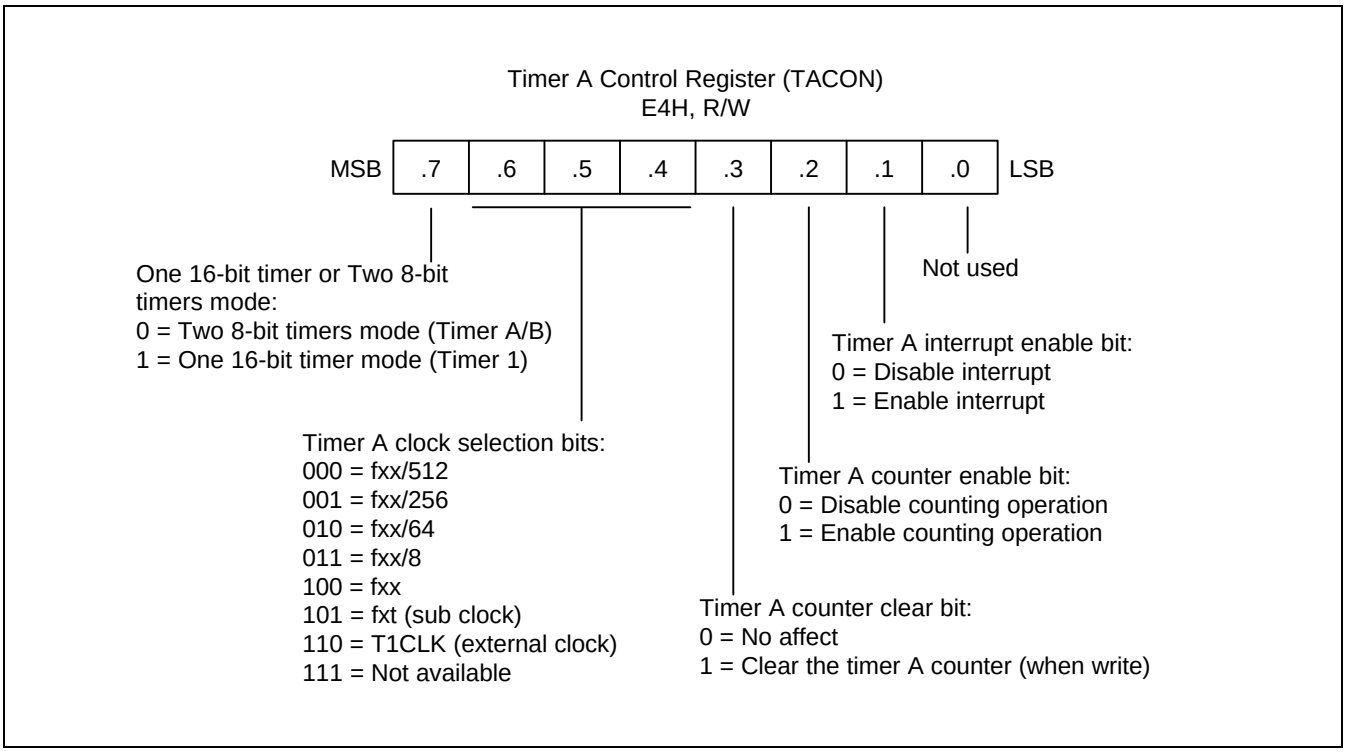


Figure 11-3. Timer A Control Register (TACON)

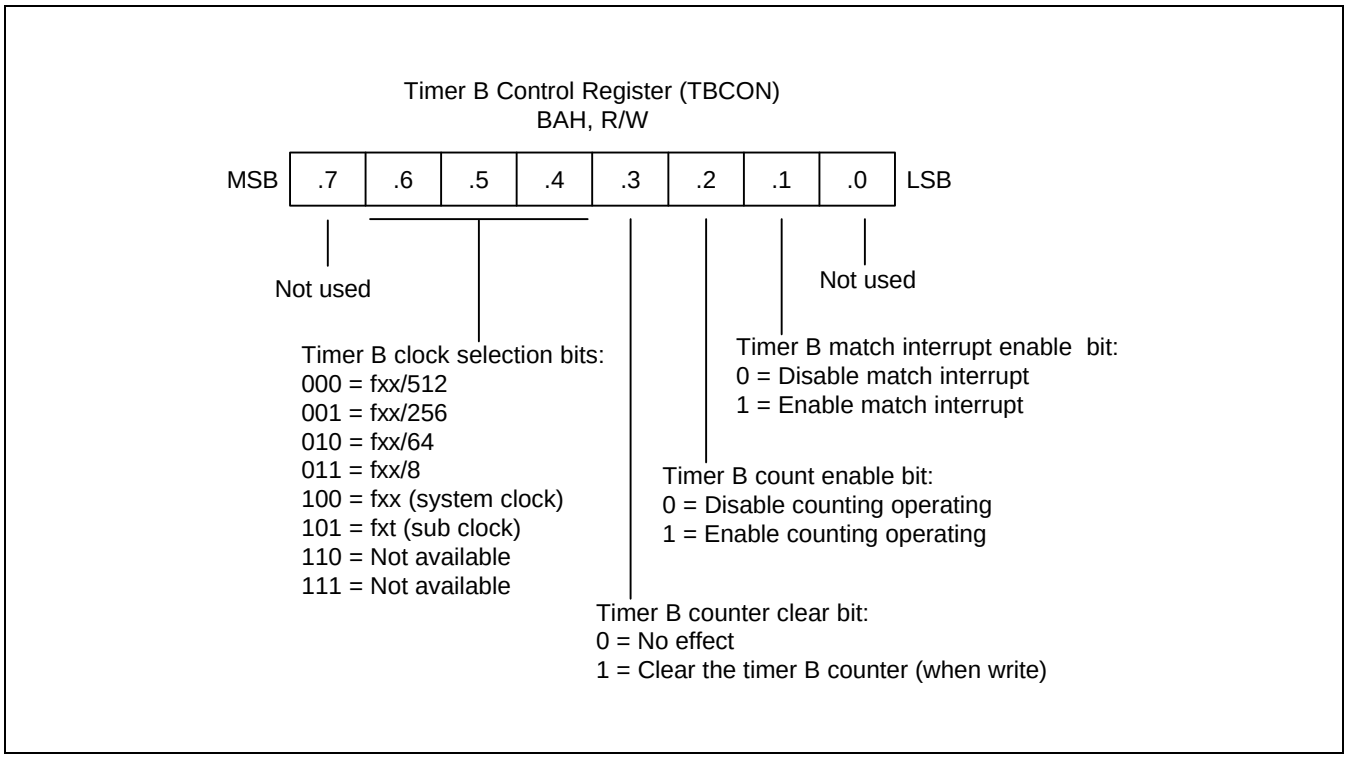


Figure 11-4. Timer B Control Register (TBCON)

FUNCTION DESCRIPTION

Interval Timer Function (Timer A and Timer B)

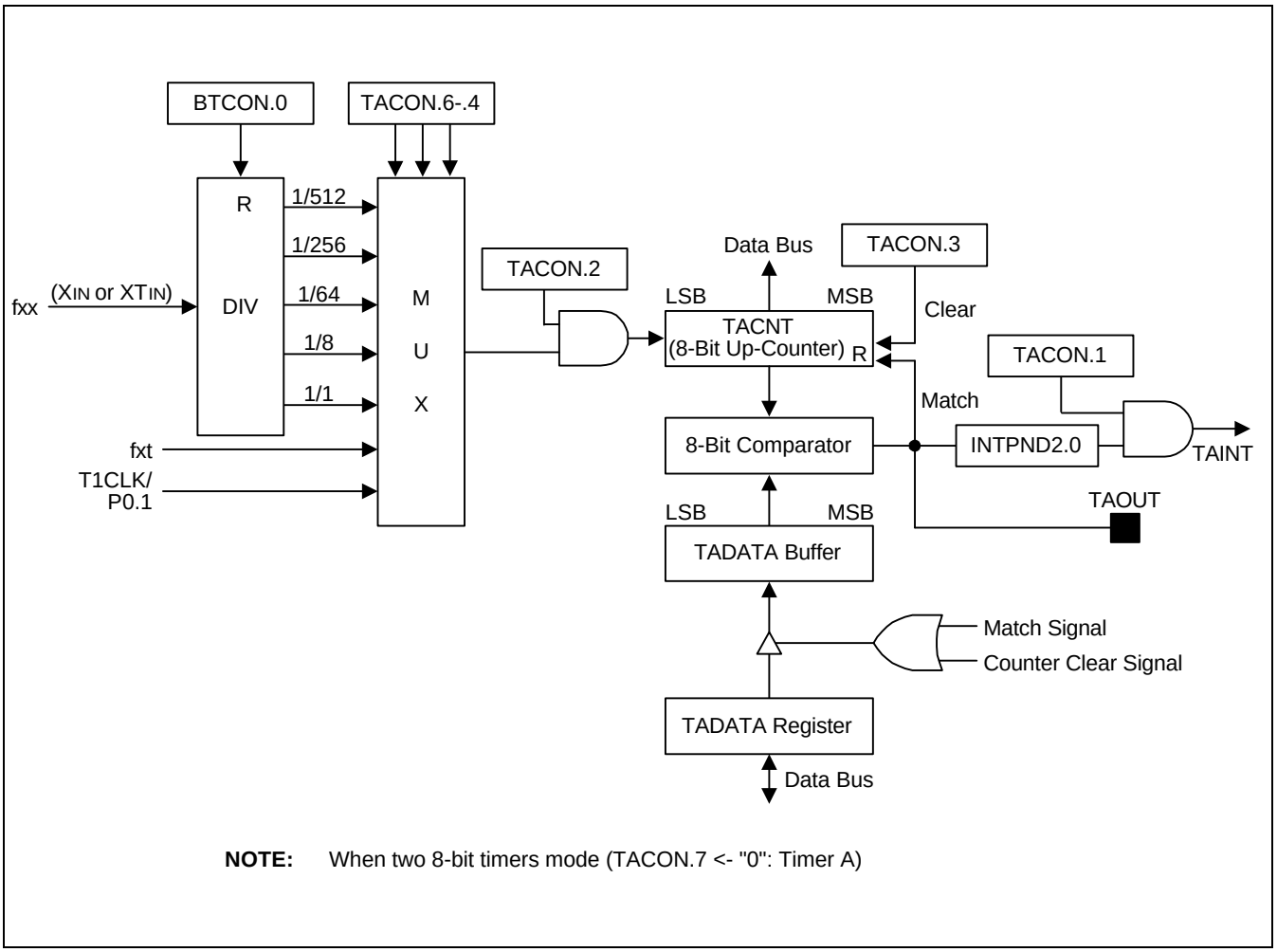
The timer A and B module can generate an interrupt: the timer A match interrupt (TAINT) and the timer B match interrupt (TBINT).

The timer A match interrupt pending condition (INTPND2.0) and the timer B match interrupt pending condition (INTPND2.1) must be cleared by software in the application's interrupt service by means of writing a "0" to the INTPND2.0 and INTPND2.1 interrupt pending bit.

Even though TAINT and TBINT are disabled, the application's service routine can detect a pending condition of TAINT and TBINT by the software and execute it's sub-routine. When this case is used, the TAINT and TBINT pending bit must be cleared by the application sub-routine by writing a "0" to the corresponding pending bit INTPND2.0 and INTPND2.1.

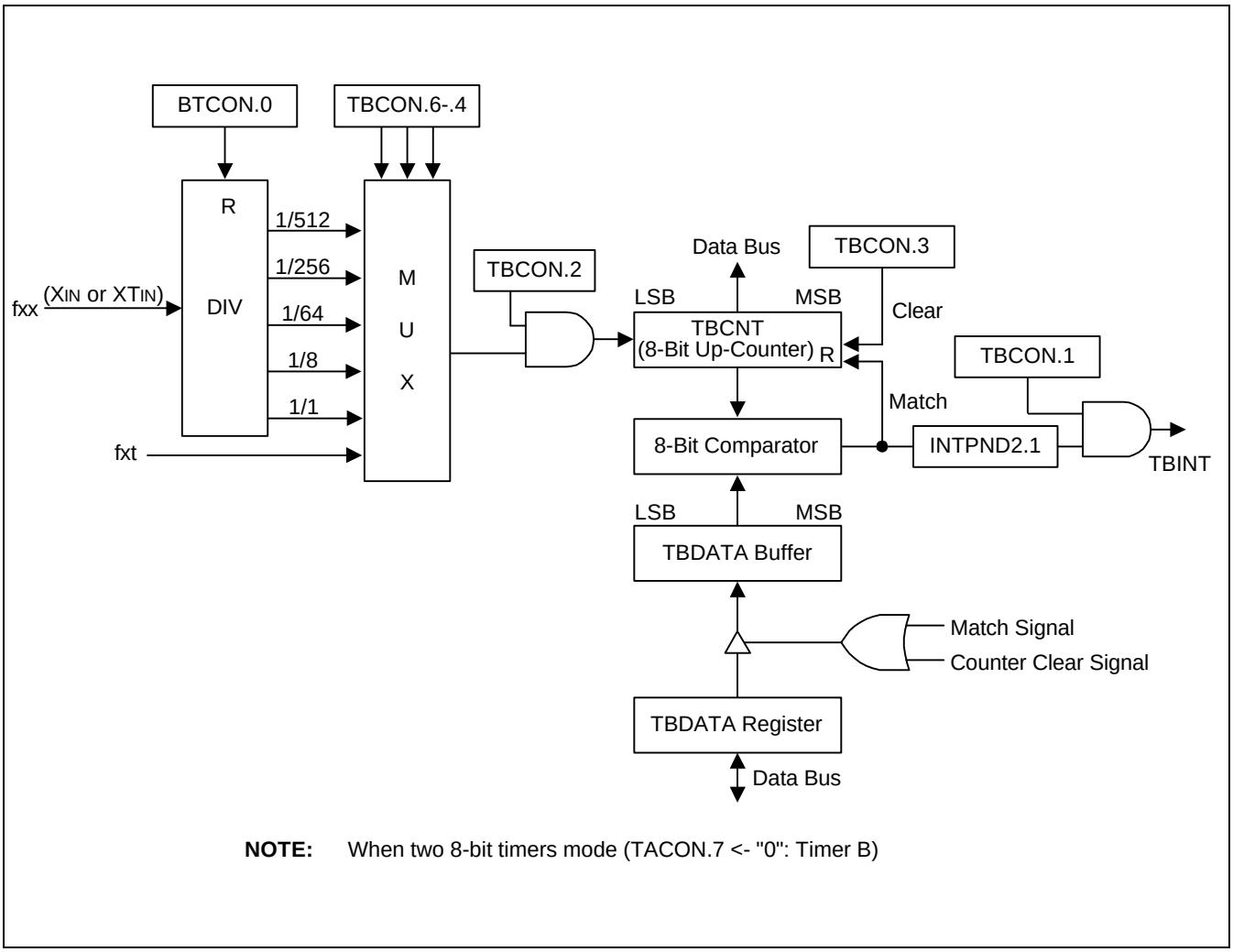
In interval timer mode, a match signal is generated when the counter value is identical to the values written to the timer A or timer B reference data registers, TADATA or TBDATA. The match signal generates corresponding match interrupt and clears the counter.

If, for example, you write the value 20H to TADATA and 0EH to TACON, the counter will increment until it reaches 20H. At this point, the timer A interrupt request is generated, the counter value is cleared, and counting resumes and you write the value 10H to TBDATA, "0" to TACON.7, and 0EH to TBCON, the counter will increment until it reaches 10H. At this point, TB interrupt request is generated, the counter value is cleared and counting resumes.



NOTE: When two 8-bit timers mode (TACON.7 <- "0": Timer A)

Figure 11-5. Timer A Block Diagram(Two 8-bit Timers Mode)



NOTE: When two 8-bit timers mode (TACON.7 <- "0": Timer B)

Figure 11-6. Timer B Block Diagram (Two 8-bit Timers Mode)

NOTES

12 WATCH TIMER

OVERVIEW

Watch timer functions include real-time and watch-time measurement and interval timing for the system clock. To start watch timer operation, set bit 1 of the watch timer control register, WTCON.1 to "1".
And if you want to service watch timer overflow interrupt, then set the WTCON.6 to “1”.
The watch timer overflow interrupt pending condition (INTPND2.3) must be cleared by software in the application's interrupt service routine by means of writing a "0" to the INTPND2.3 interrupt pending bit.
After the watch timer starts and elapses a time, the watch timer interrupt pending bit (INTPND2.3) is automatically set to "1", and interrupt requests commence in 3.91ms, 0.25, 0.5 and 1-second intervals by setting Watch timer speed selection bits (WTCON.3 – .2).

The watch timer can generate a steady 0.5 kHz, 1 kHz, 2 kHz, or 4 kHz signal to BUZ output pin for Buzzer. By setting WTCON.3 and WTCON.2 to "11b", the watch timer will function in high-speed mode, generating an interrupt every 3.91 ms. High-speed mode is useful for timing events for program debugging sequences.

Also, you can select watch timer clock source by setting the WTCON.7 appropriately value.

The watch timer supplies the clock frequency for the LCD controller (f_{LCD}). Therefore, if the watch timer is disabled, the LCD controller does not operate.

Watch timer has the following functional components:

— Real Time and Watch-Time Measurement
— Using a Main or Sub Clock Source (Main clock divided by 2^7(fx/128) or Sub clock(fxt))
— Clock Source Generation for LCD Controller (f_{LCD})
— I/O pin for Buzzer Output Frequency Generator (P0.3, BUZ)
— Timing Tests in High-Speed Mode
— Watch timer overflow interrupt generation
— Watch timer control register, WTCON (page 0, DAH, read/write)

WATCH TIMER CONTROL REGISTER (WTCON)

The watch timer control register, WTCON is used to select the input clock source, the watch timer interrupt time and Buzzer signal, to enable or disable the watch timer function. It is located in page 0 at address DAH, and is read/write addressable using register addressing mode.
A reset clears WTCON to "00H". This disable the watch timer and select fx/128 as the watch timer clock.
So, if you want to use the watch timer, you must write appropriate value to WTCON.

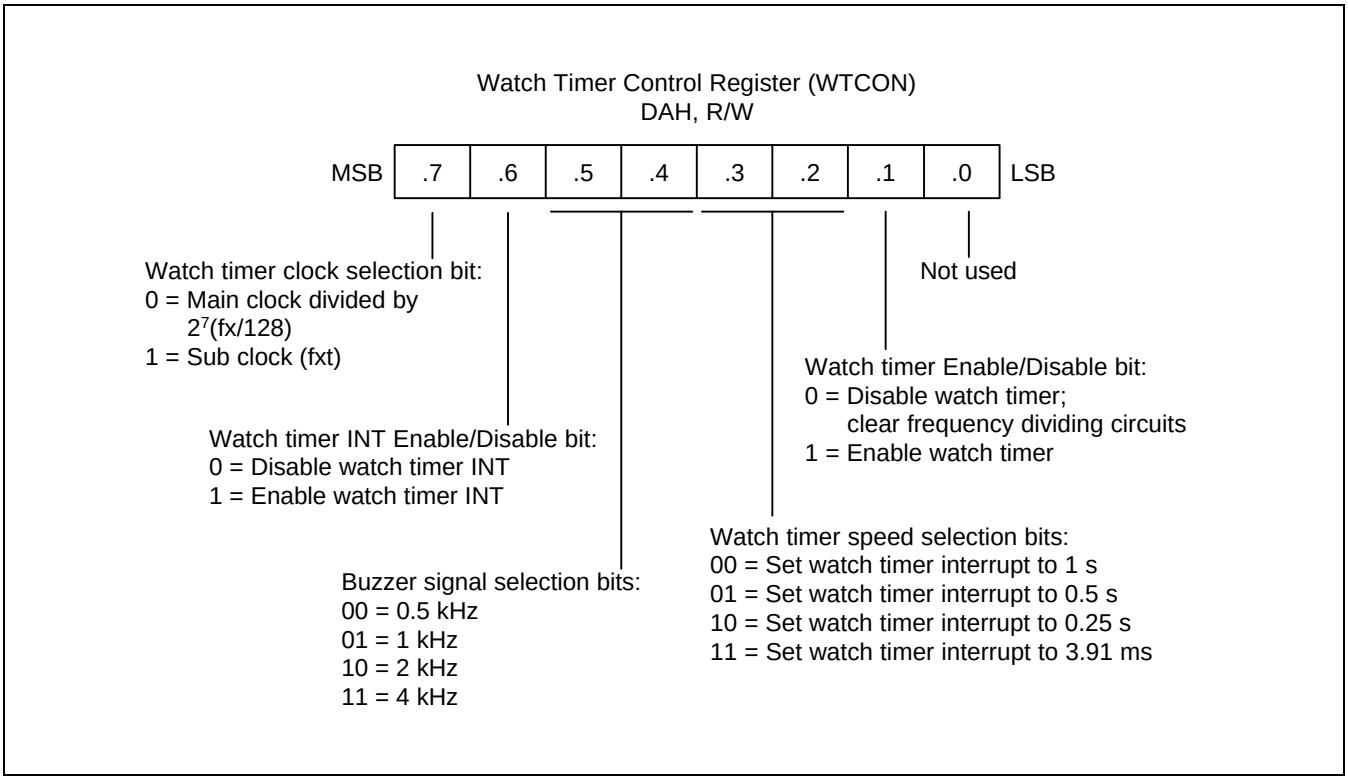


Figure 12-1. Watch Timer Control Register (WTCON)

WATCH TIMER CIRCUIT DIAGRAM

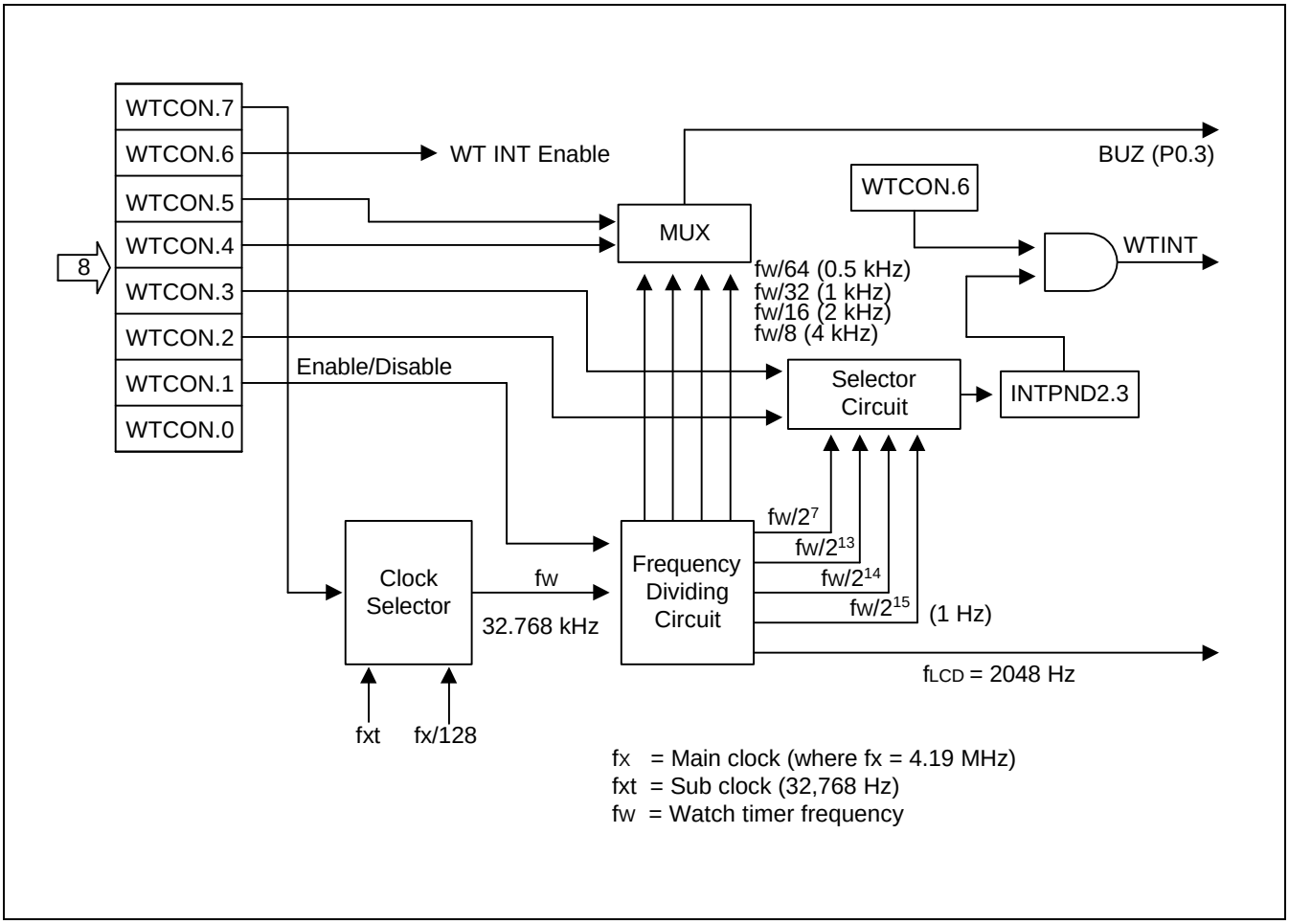


Figure 12-2. Watch Timer Circuit Diagram

NOTES

13 LCD CONTROLLER/DRIVER

OVERVIEW

The S3C9228/P9228 microcontroller can directly drive an up-to-128-dot (16segments x 8 commons) LCD panel. Its LCD block has the following components:

— LCD controller/driver
— Display RAM for storing display data
— 16 segment output pins (SEG0–SEG15)
— 8 common output pins (COM0–COM7)
— Internal resistor circuit for LCD bias

To use the LCD controller, bit 2 in the watch mode register WMOD must be set to 1 because LCDCK is supplied by the watch timer.

The LCD mode control register, LMOD, is used to turn the LCD display on or off, to select LCD clock frequency, to turn the COM signal output on or off, to select bias and duty, and to switch the port 3 high impedance or normal I/O port. Data written to the LCD display RAM can be transferred to the segment signal pins automatically without program control.

The LCD port control register, LPOT, is used to determine the LCD signal pins used for display output.

When a sub clock is selected as the LCD clock source, the LCD display is enabled even during main clock stop and idle modes.

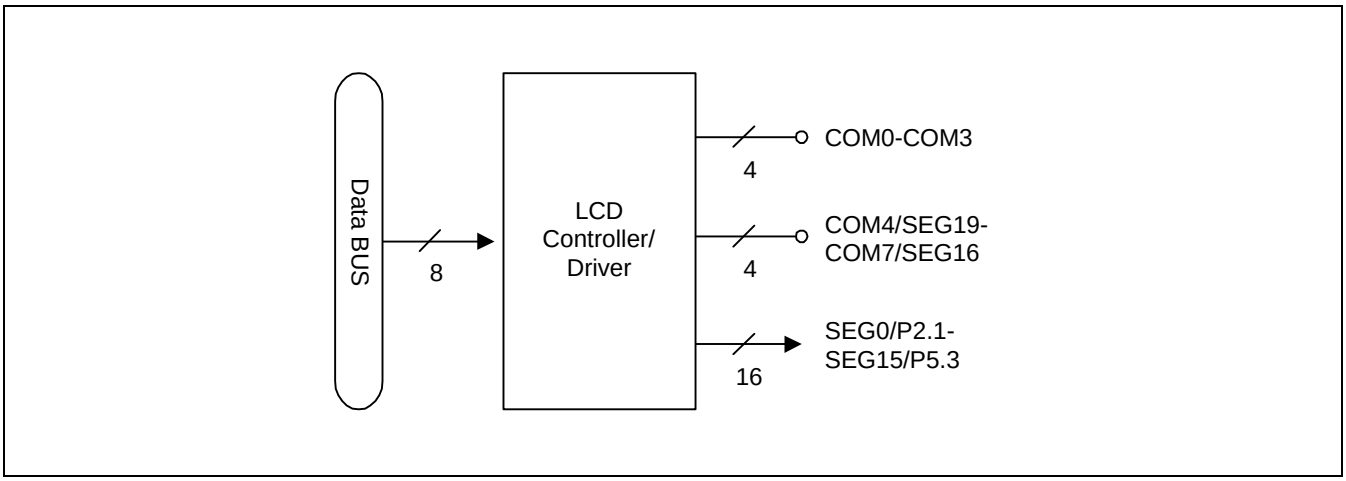


Figure 13-1. LCD Function Diagram

LCD CIRCUIT DIAGRAM

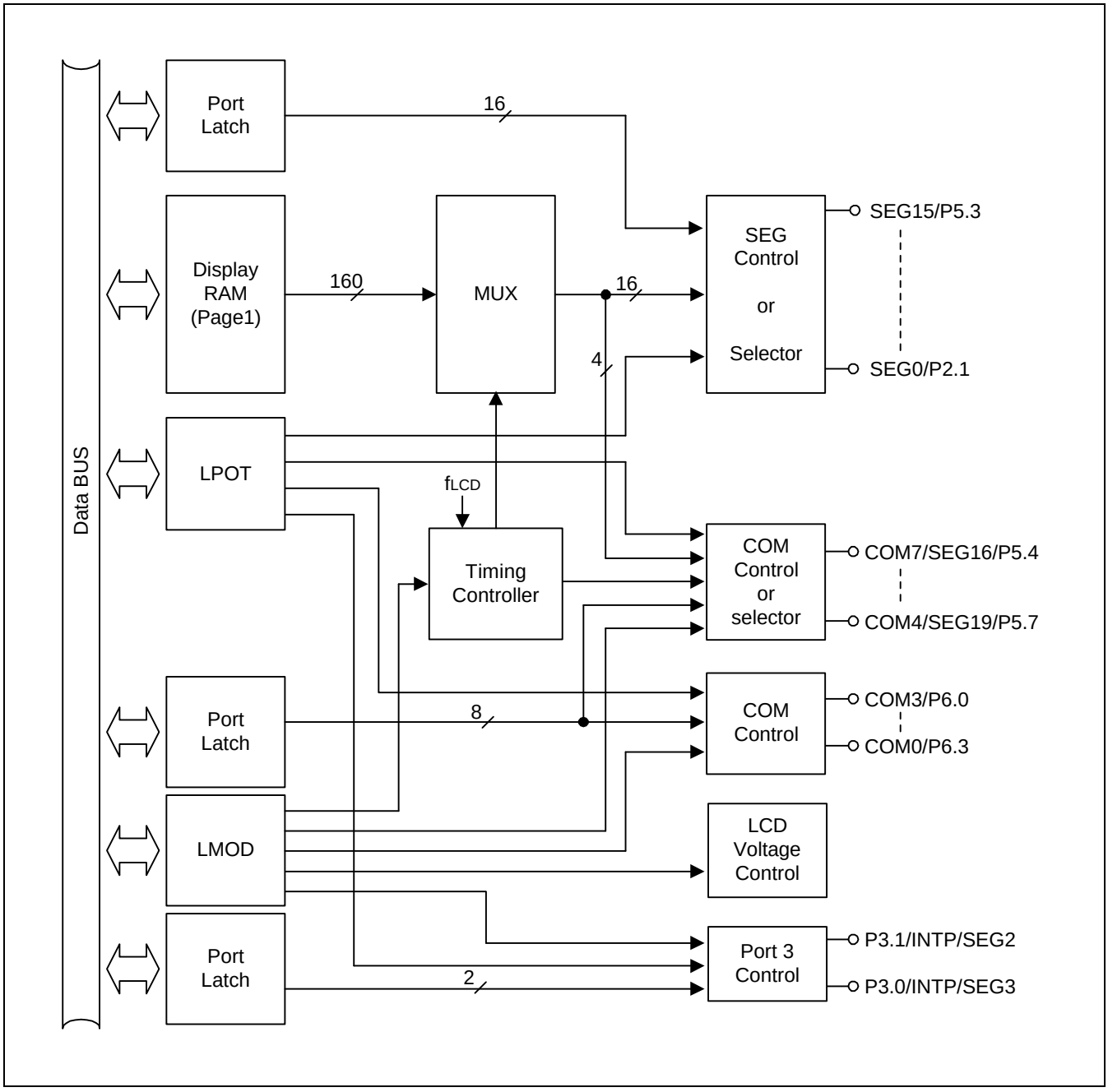


Figure 13-2. LCD Circuit Diagram

LCD RAM ADDRESS AREA

RAM addresses of page 1 are used as LCD data memory. When the bit value of a display segment is "1", the LCD display is turned on; when the bit value is "0", the display is turned off.

Display RAM data are sent out through segment pins SEG0–SEG19 using a direct memory access (DMA) method that is synchronized with the f_{LCD} signal. RAM addresses in this location that are not used for LCD display can be allocated to general-purpose use.

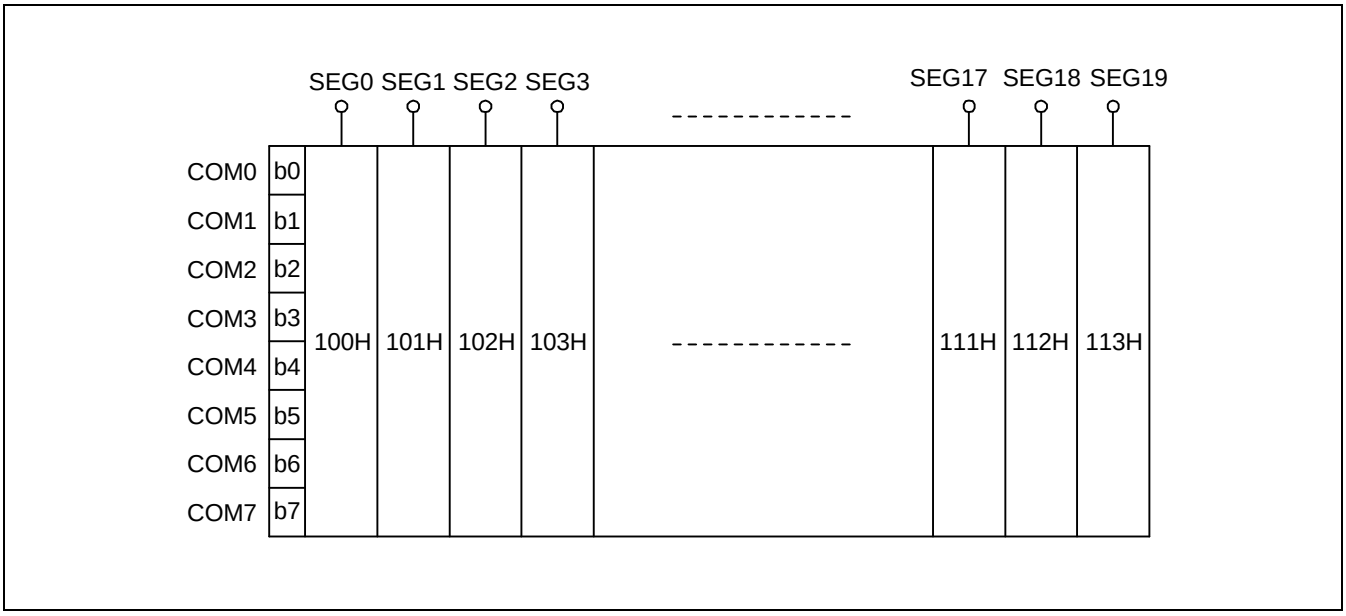


Figure 13-3. LCD Display Data RAM Organization

Table 13-1. Common and Segment Pins per Duty Cycle

Duty	Common Pins	Segment Pins	Dot Number
1/8	COM0–COM7	16 pins	128 dots
1/4	COM0–COM3	20 pins	80 dots
1/3	COM0–COM2	20 pins	60 dots

LCD MODE CONTROL REGISTER (LMOD)

A LMOD is located in page 0, at address FEH, and is read/write addressable using register addressing mode. It has the following control functions.

— LCD duty and bias selection
— LCD clock selection
— LCD display control
— COMs signal output control
— P3 high impedance control

The LMOD register is used to turn the LCD display on/off, to select duty and bias, to select LCD clock, to control port 3 high impedance/normal I/O port, and to turn the COM signal output on/off. Following a RESET, all LMOD values are cleared to "0". This turns off the LCD display, select 1/3 duty and 1/3 bias, and select 256Hz for LCD clock.

The LCD clock signal determines the frequency of COM signal scanning of each segment output. This is also referred as the LCD frame frequency. Since the LCD clock is generated by watch timer clock (fw). The watch timer should be enabled when the LCD display is turned on.

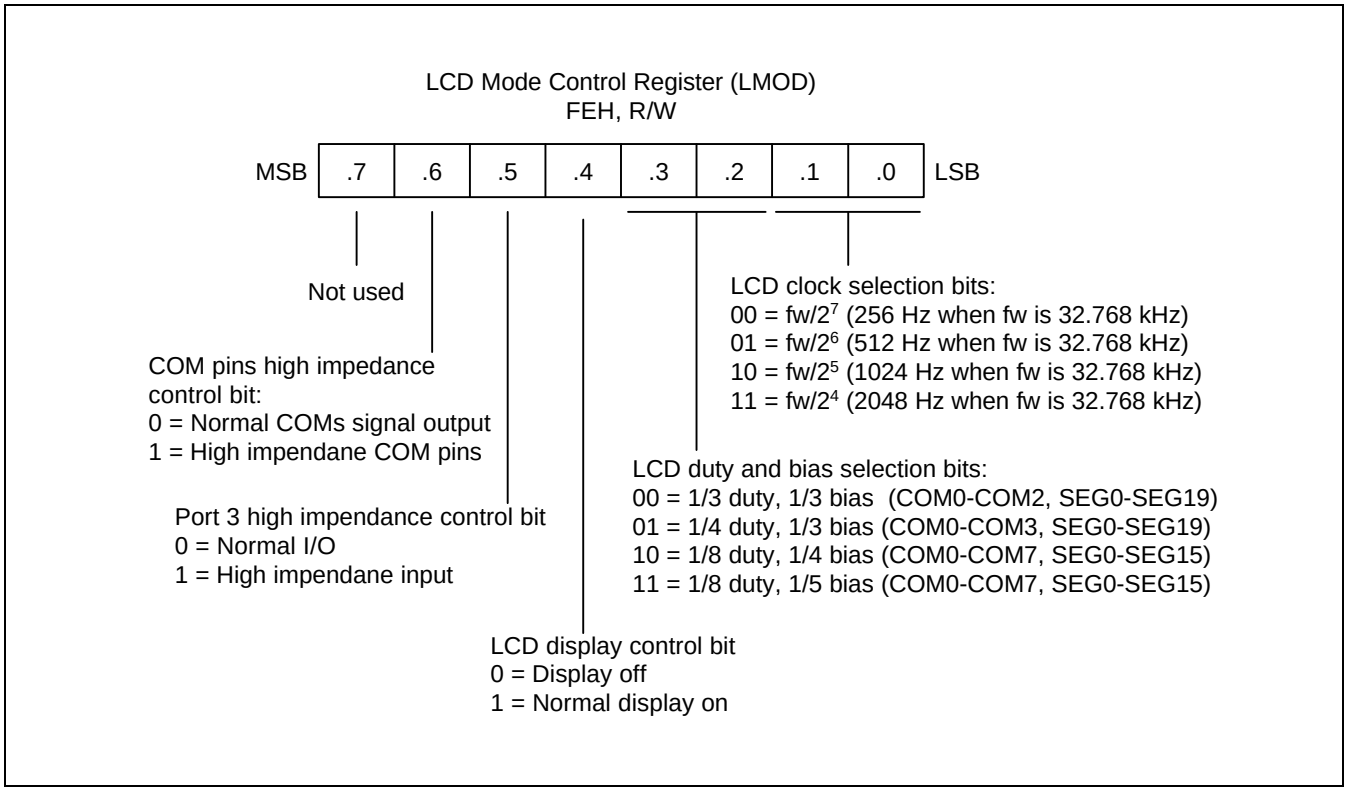


Figure 13-4. LCD Mode Control Register (LMOD)

LCD PORT CONTROL REGISTER

The LCD port control register LPOT is used to control LCD signal pins or normal I/O pins. Following a RESET, a LPOT values are cleared to "0".

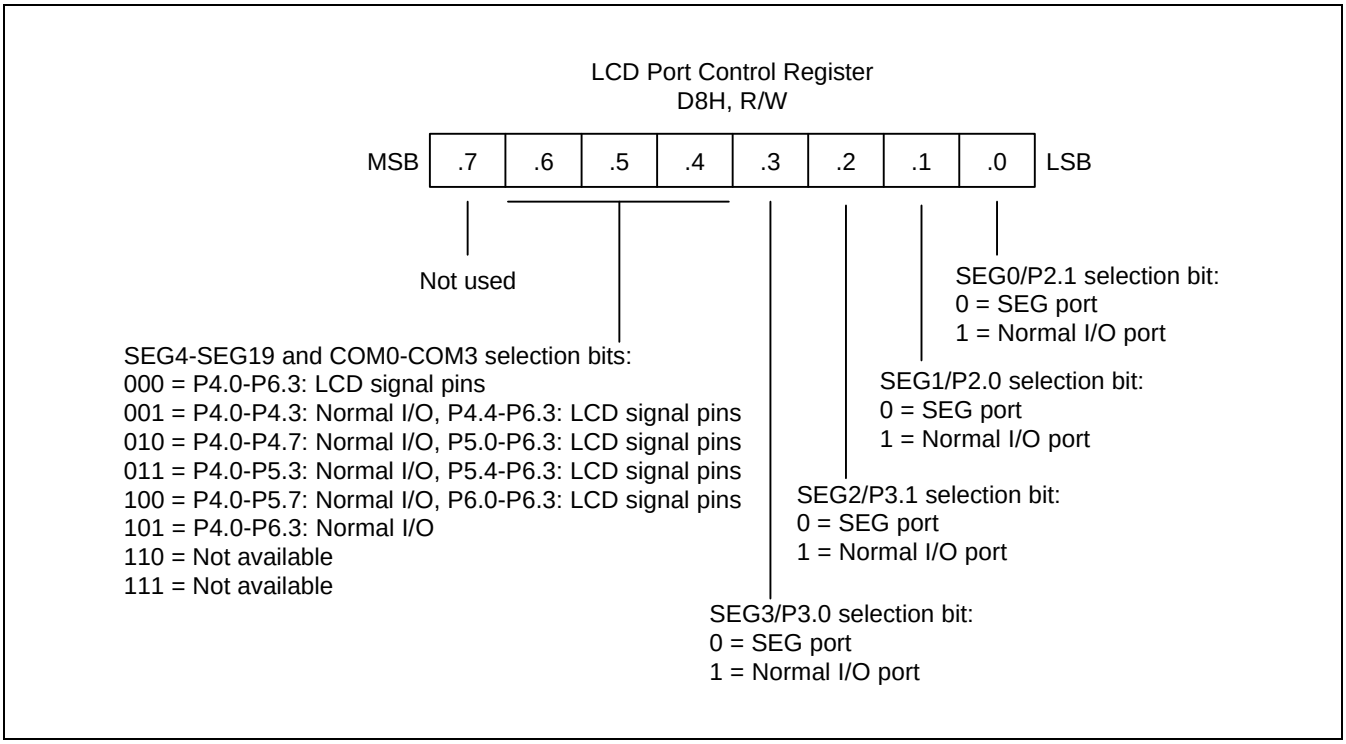


Figure 13-5. LCD Port Control Register

LCD VOLTAGE DIVIDING RESISTORS

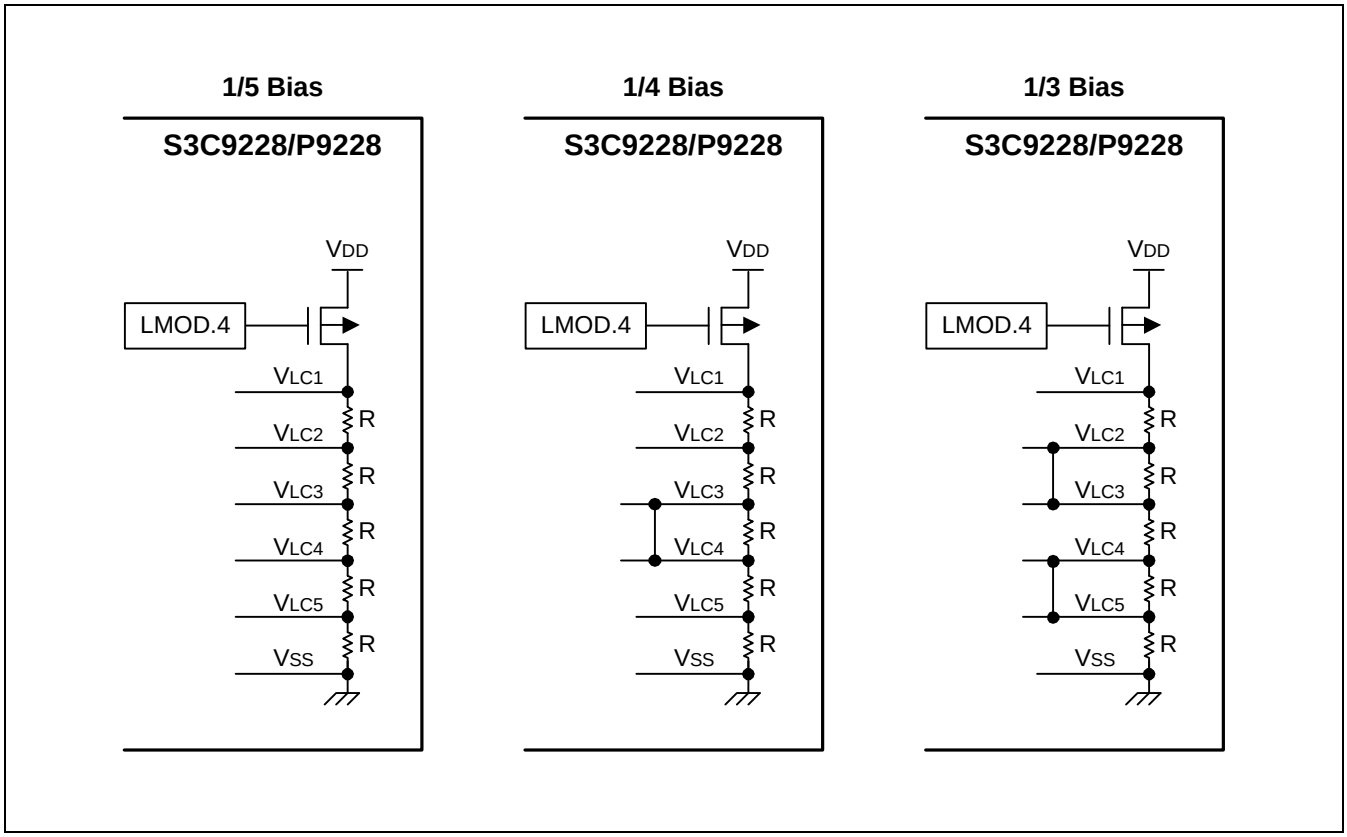


Figure 13-6. Internal Voltage Dividing Resistor Connection

COMMON (COM) SIGNALS

The common signal output pin selection (COM pin selection) varies according to the selected duty cycle.

— In 1/3 duty mode, COM0-COM2 pins are selected
— In 1/4 duty mode, COM0-COM3 pins are selected
— In 1/8 duty mode, COM0-COM7 pins are selected

SEGMENT (SEG) SIGNALS

The 19 LCD segment signal pins are connected to corresponding display RAM locations at page 1. Bits of the display RAM are synchronized with the common signal output pins.

When the bit value of a display RAM location is "1", a select signal is sent to the corresponding segment pin. When the display bit is "0", a 'no-select' signal to the corresponding segment pin.

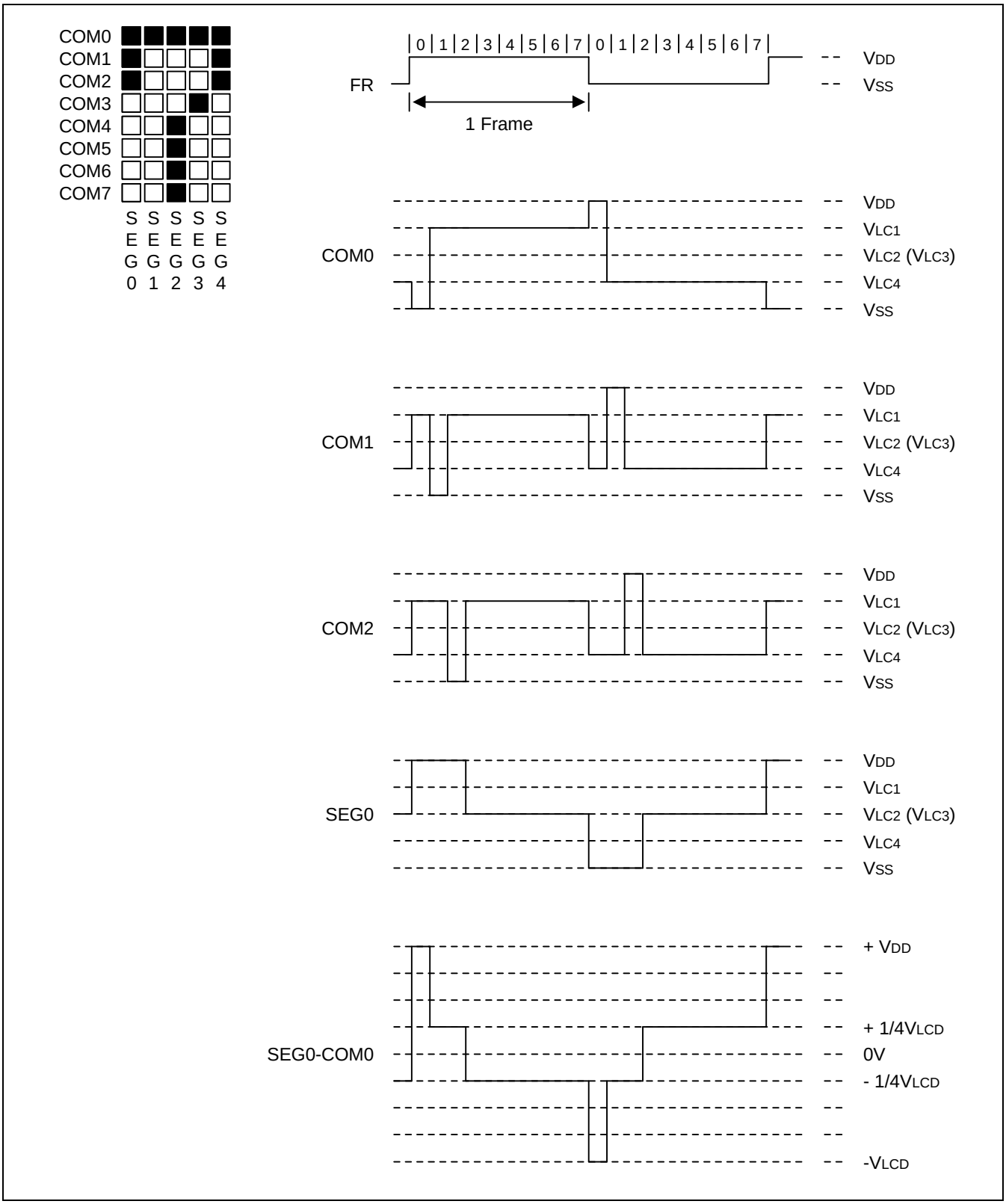


Figure 13-7. LCD Signal Waveforms (1/8 Duty, 1/4 Bias)

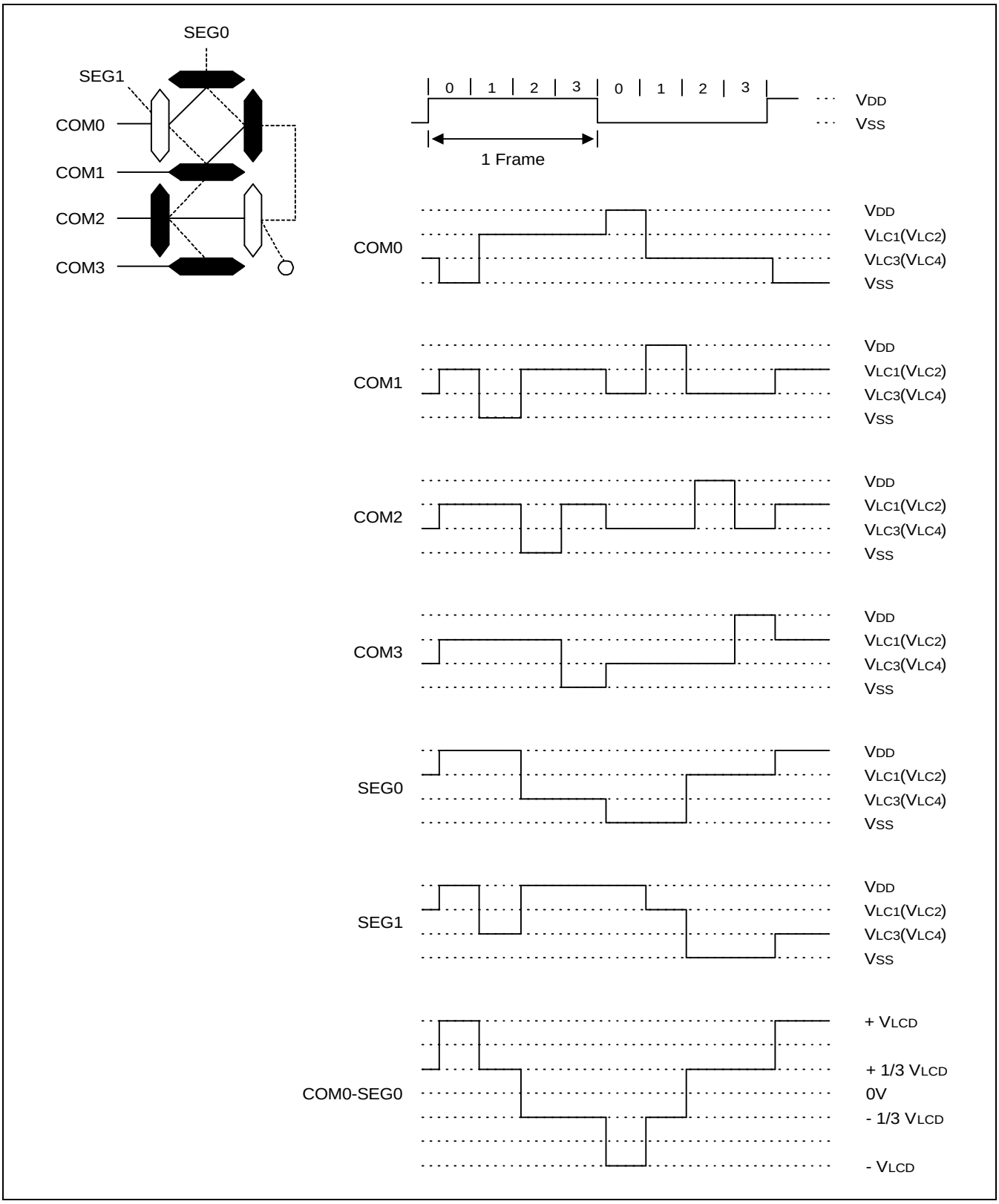


Figure 13-8. LCD Signal Waveforms (1/4 Duty, 1/3 Bias)

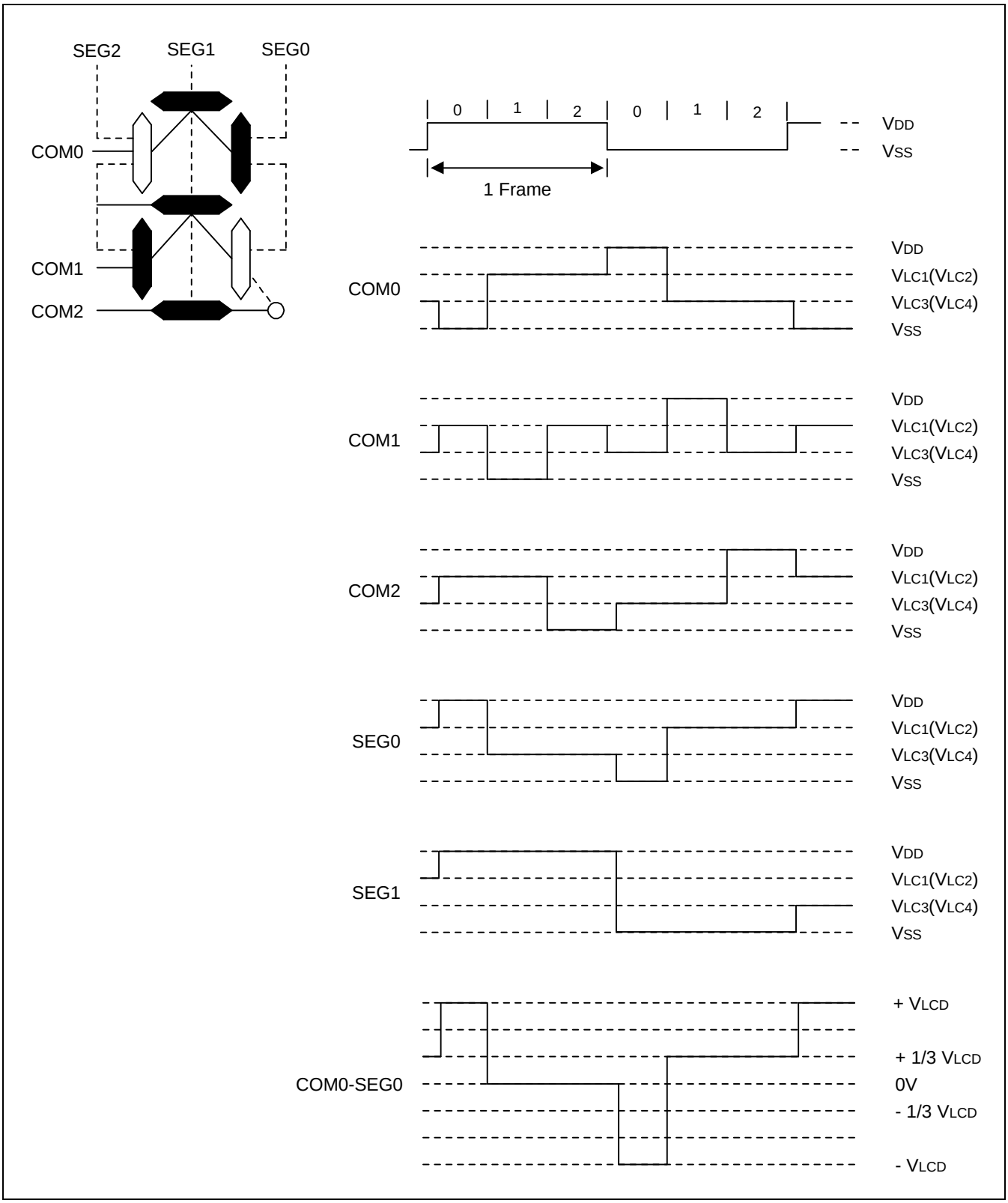


Figure 13-9. LCD Signal Waveforms (1/3 Duty, 1/3 Bias)

NOTES

14 10-BIT ANALOG-TO-DIGITAL CONVERTER

OVERVIEW

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

— Analog comparator with successive approximation logic

— D/A converter logic (resistor string type)

— ADC control register (ADCON)

— Four multiplexed analog data input pins (AD0–AD3)

— 10-bit A/D conversion data output register (ADDATAH/ADDATAL)

— 4-bit digital input port (Alternately, I/O port)

FUNCTION DESCRIPTION

To initiate an analog-to-digital conversion procedure, at first you must set with alternative function for ADC input enable at port 1, the pin set with alternative function can be used for ADC analog input. And you write the channel selection data in the A/D converter control register ADCON.4–.5 to select one of the four analog input pins (AD0–3) and set the conversion start or enable bit, ADCON.0. The read-write ADCON register is located in page 0, at address D0H. The pins which are not used for ADC can be used for normal I/O.

During a normal conversion, ADC logic initially sets the successive approximation register to 800H (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.5–4) 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 ADDATAH/ADDATAL register where it can be read. The A/D converter then enters an idle state. Remember to read the contents of ADDATAH/ADDATAL before another conversion starts. Otherwise, the previous result will be overwritten by the next conversion result.

NOTE

Because the A/D converter has no sample-and-hold circuitry, it is very important that fluctuation in the analog level at the AD0–AD3 input pins during a conversion procedure be kept to an absolute minimum. Any change in the input level, perhaps due to noise, will invalidate the result. If the chip enters to STOP or IDLE mode in conversion process, there will be a leakage current path in A/D block. You must use STOP or IDLE mode after ADC operation is finished.

CONVERSION TIMING

The A/D conversion process requires 4 steps (4 clock edges) to convert each bit and 10 clocks to set-up A/D conversion. Therefore, total of 50 clocks are required to complete an 10-bit conversion: When fxx/8 is selected for conversion clock with an 4.5 MHz fxx clock frequency, one clock cycle is 1.78 us. Each bit conversion requires 4 clocks, the conversion rate is calculated as follows:

4 clocks/bit × 10-bit + set-up time = 50 clocks, 50 clock × 1.78 us = 89 us at 0.56 MHz (4.5 MHz/8)

Note that A/D converter needs at least 25μs for conversion time.

A/D CONVERTER CONTROL REGISTER (ADCON)

The A/D converter control register, ADCON, is located at address D0H in page 0. It has three functions:

— Analog input pin selection (bits 4 and 5)

— End-of-conversion status detection (bit 3)

— ADC clock selection (bits 2 and 1)

— A/D operation start or enable (bit 0)

After a reset, the start bit is turned off. You can select only one analog input channel at a time. Other analog input pins (AD0–AD3) can be selected dynamically by manipulating the ADCON.4–5 bits. And the pins not used for analog input can be used for normal I/O function.

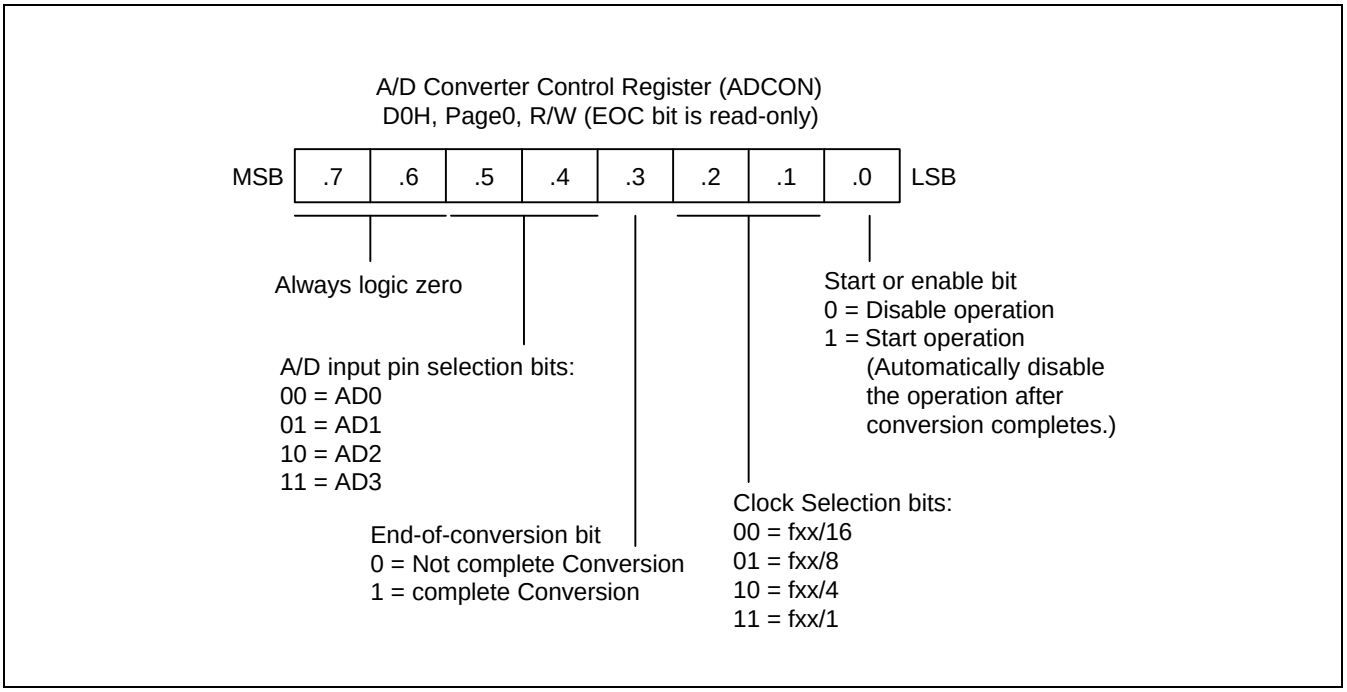


Figure 14-1. A/D Converter Control Register (ADCON)

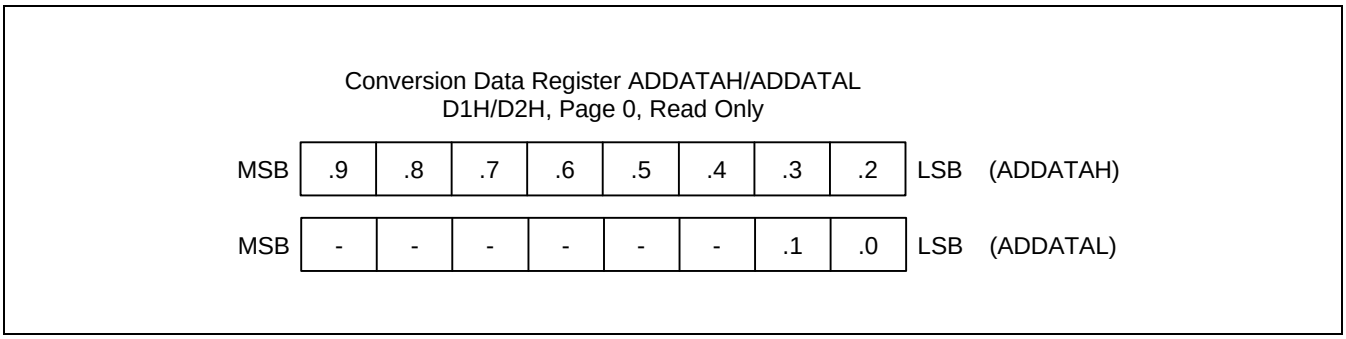


Figure 14-2. A/D Converter Data Register (ADDATAH/ADDATAL)

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 V_{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 conversion bit is always 1/2 V_{DD}.

BLOCK DIAGRAM

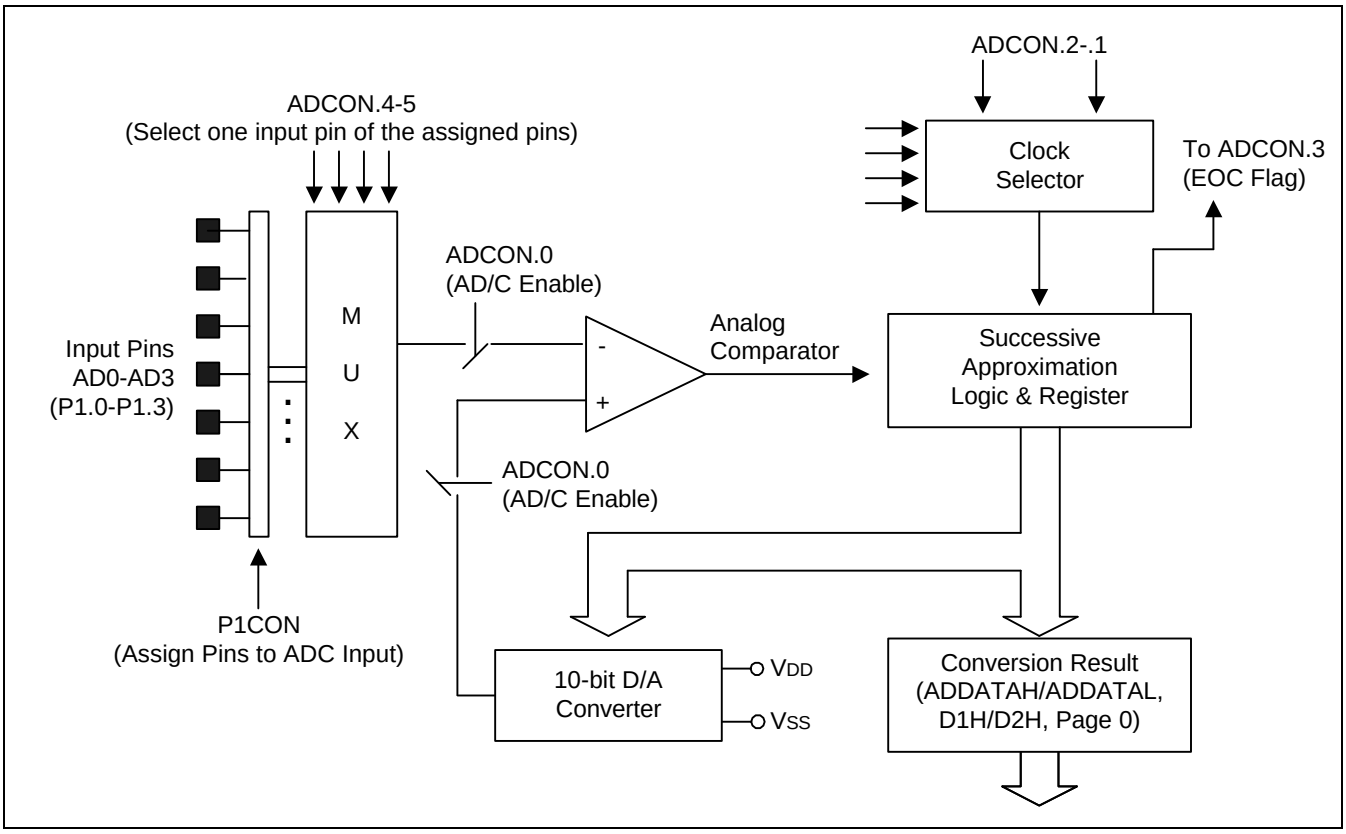


Figure 14-3. A/D Converter Functional Block Diagram

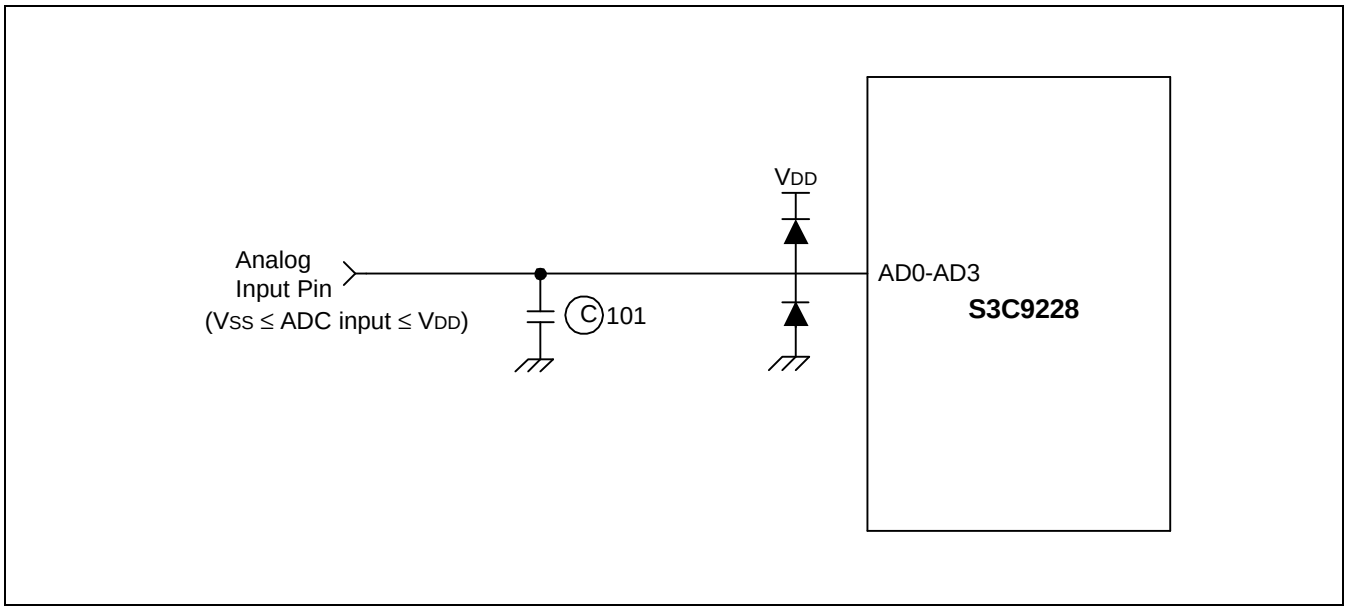


Figure 14-4. Recommended A/D Converter Circuit for Highest Absolute Accuracy

15 SERIAL I/O INTERFACE

OVERVIEW

Serial I/O modules, SIO can interface with various types of external device that require serial data transfer. The components of SIO function block are:

— 8-bit control register (SIOCON)
— Clock selector logic
— 8-bit data buffer (SIODATA)
— 8-bit prescaler (SIOPS)
— 3-bit serial clock counter
— Serial data I/O pins (SI, SO)
— Serial clock input/output pin (SCK)

The SIO module can transmit or receive 8-bit serial data at a frequency determined by its corresponding control register settings. To ensure flexible data transmission rates, you can select an internal or external clock source.

PROGRAMMING PROCEDURE

To program the SIO module, follow these basic steps:

1. Configure the I/O pins at port (SCK/SI/SO) by loading the appropriate value to the P2CON register if necessary.
2. Load an 8-bit value to the SIOCON control register to properly configure the serial I/O module. In this operation, SIOCON.2 must be set to "1" to enable the data shifter.
3. For interrupt generation, set the serial I/O interrupt enable bit (SIOCON) to "1".
4. When you transmit data to the serial buffer, write data to SIODATA and set SIOCON.3 to 1, the shift operation starts.
5. When the shift operation (transmit/receive) is completed, the SIO pending bit (INTPND2.2) are set to "1" and SIO interrupt request is generated.

SIO CONTROL REGISTERS (SIOCON)

The control register for serial I/O interface module, SIOCON, is located at E1H in page 0. It has the control setting for SIO module.

— Clock source selection (internal or external) for shift clock
— Interrupt enable
— Edge selection for shift operation
— Clear 3-bit counter and start shift operation
— Shift operation (transmit) enable
— Mode selection (transmit/receive or receive-only)
— Data direction selection (MSB first or LSB first)

A reset clears the SIOCON value to "00H". This configures the corresponding module with an internal clock source at the SCK, selects receive-only operating mode, and clears the 3-bit counter. The data shift operation and the interrupt are disabled. The selected data direction is MSB-first.

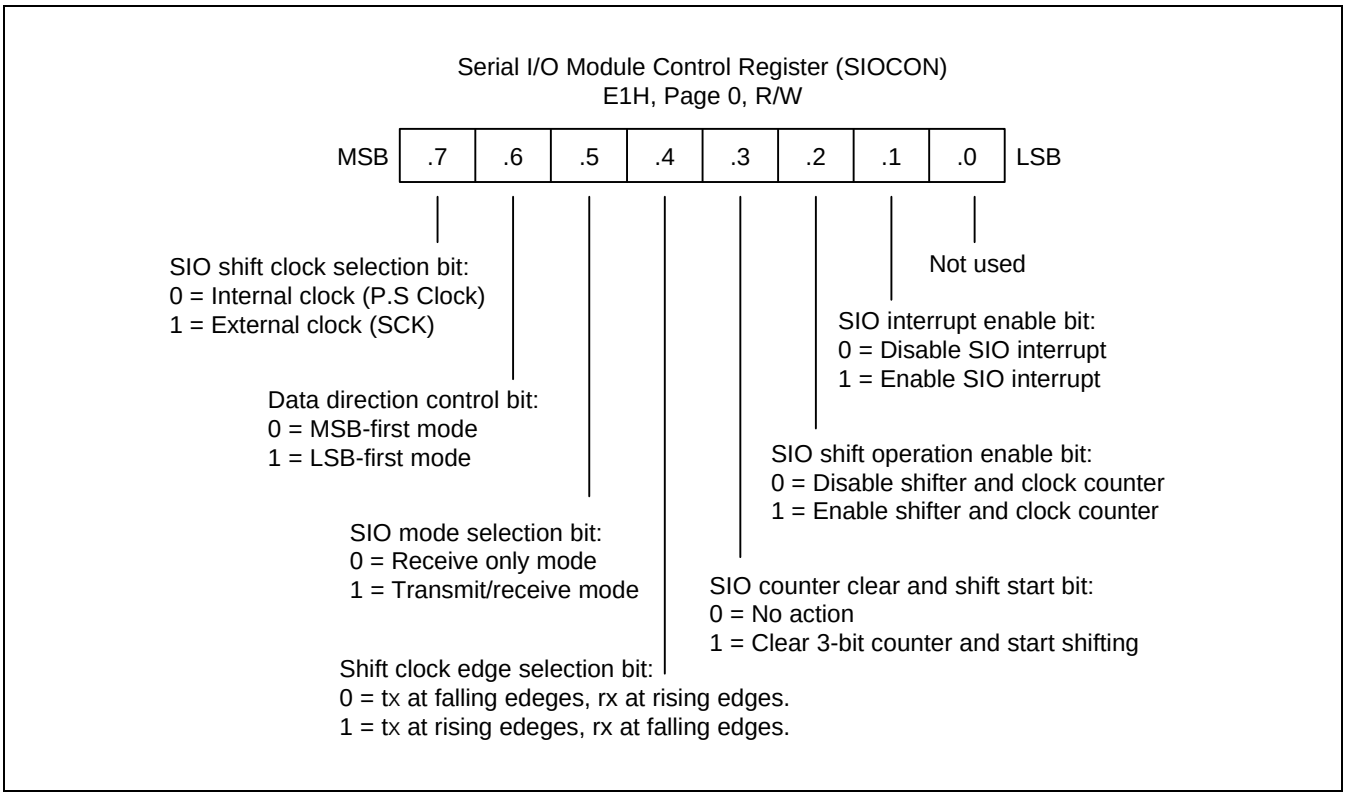


Figure 15-1. Serial I/O Module Control Register (SIOCON)

SIO PRE-SCALER REGISTER (SIOPS)

The prescaler register for serial I/O interface module, SIOPS, are located at E3H in page 0.
The value stored in the SIO pre-scale register, SIOPS, lets you determine the SIO clock rate (baud rate) as follows:

Baud rate = Input clock (fxx/4)/(Prescaler value + 1), or SCK input clock.

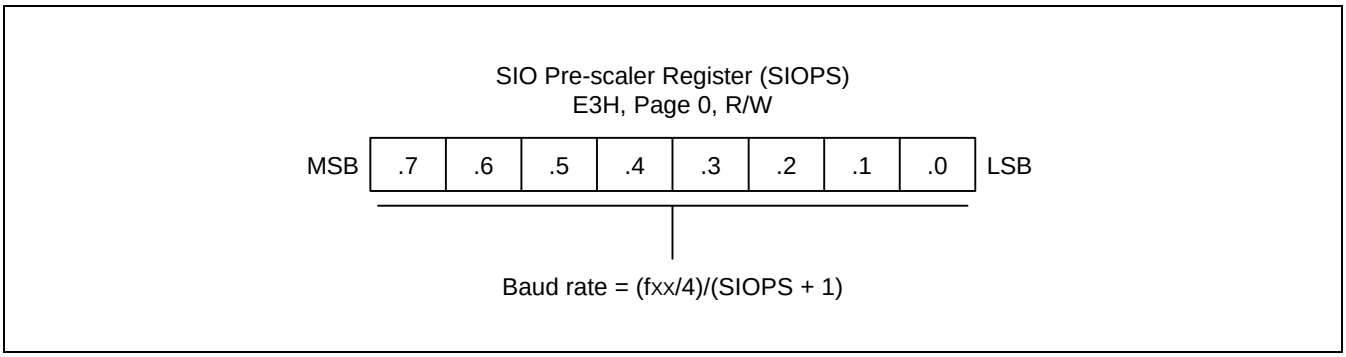


Figure 15-2. SIO Prescaler Register (SIOPS)

SIO BLOCK DIAGRAM

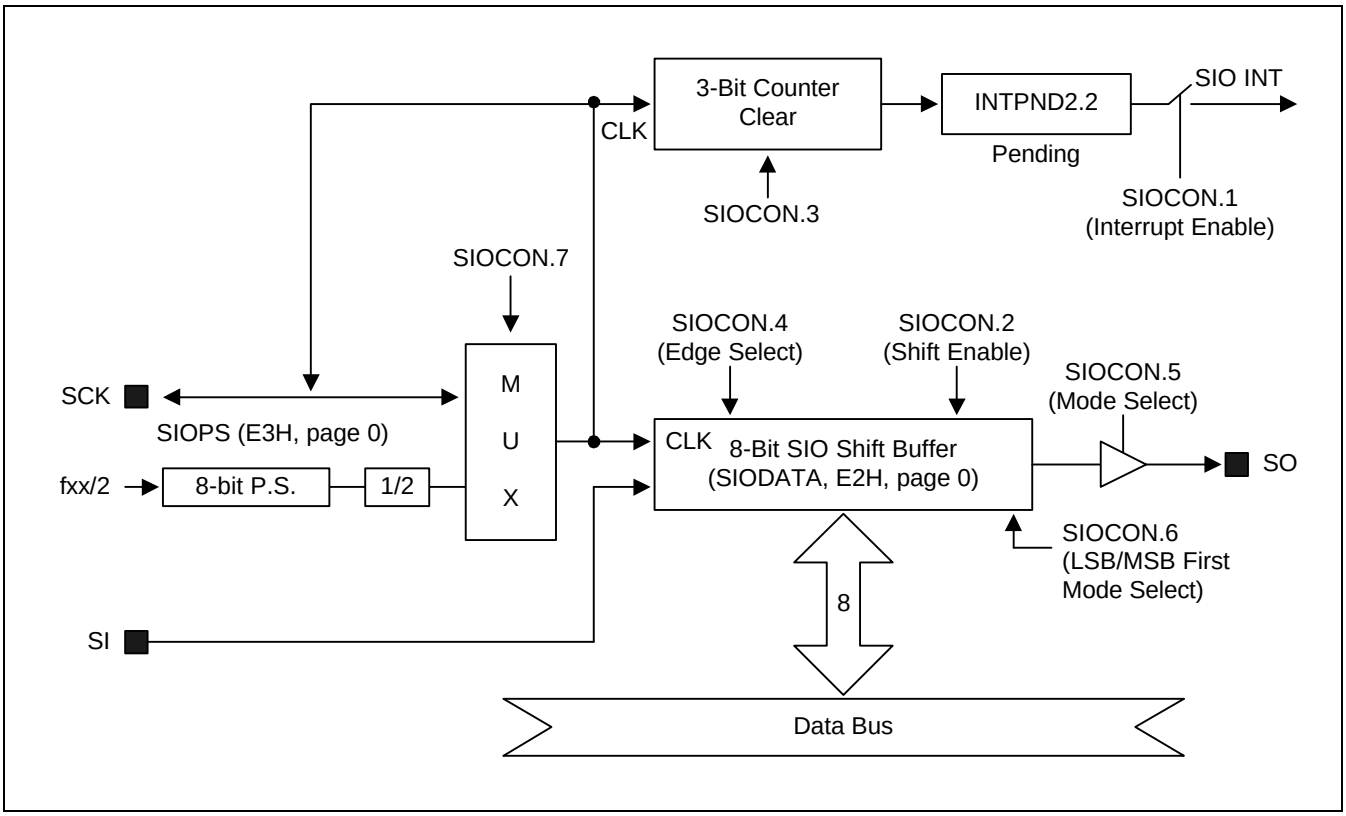


Figure 15-3. SIO Functional Block Diagram

SERIAL I/O TIMING DIAGRAM (SIO)

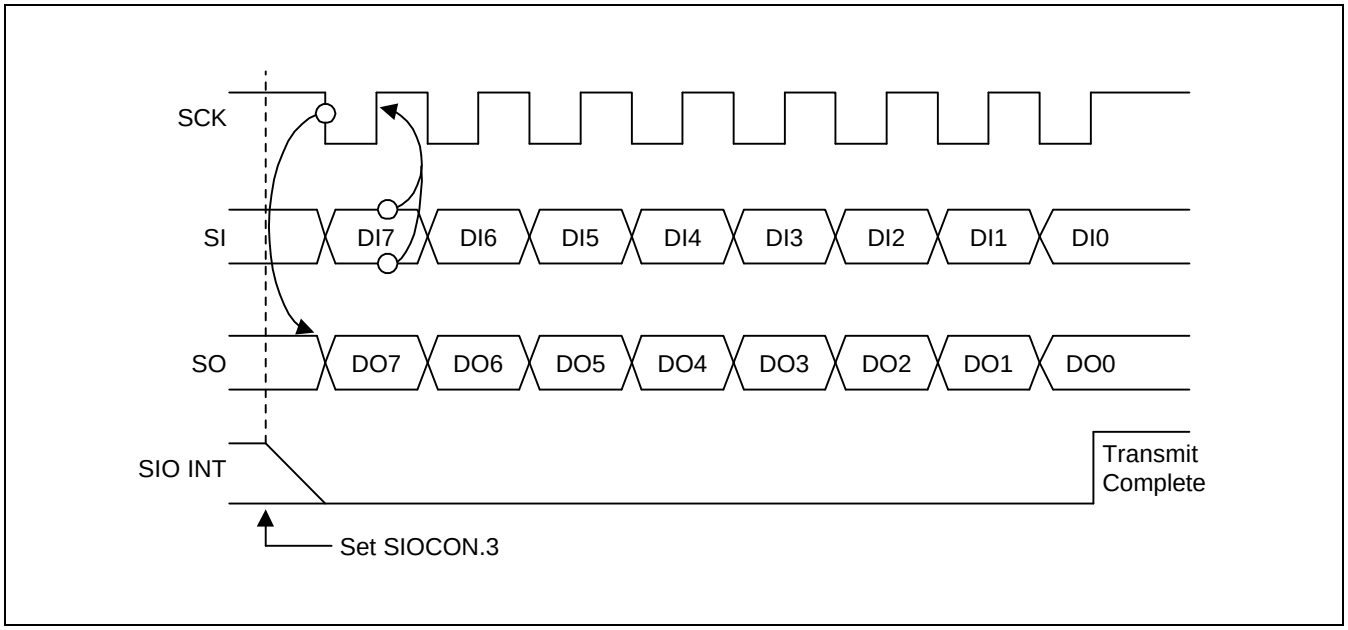


Figure 15-4. Serial I/O Timing in Transmit/Receive Mode (Tx at falling, SIOCON.4 = 0)

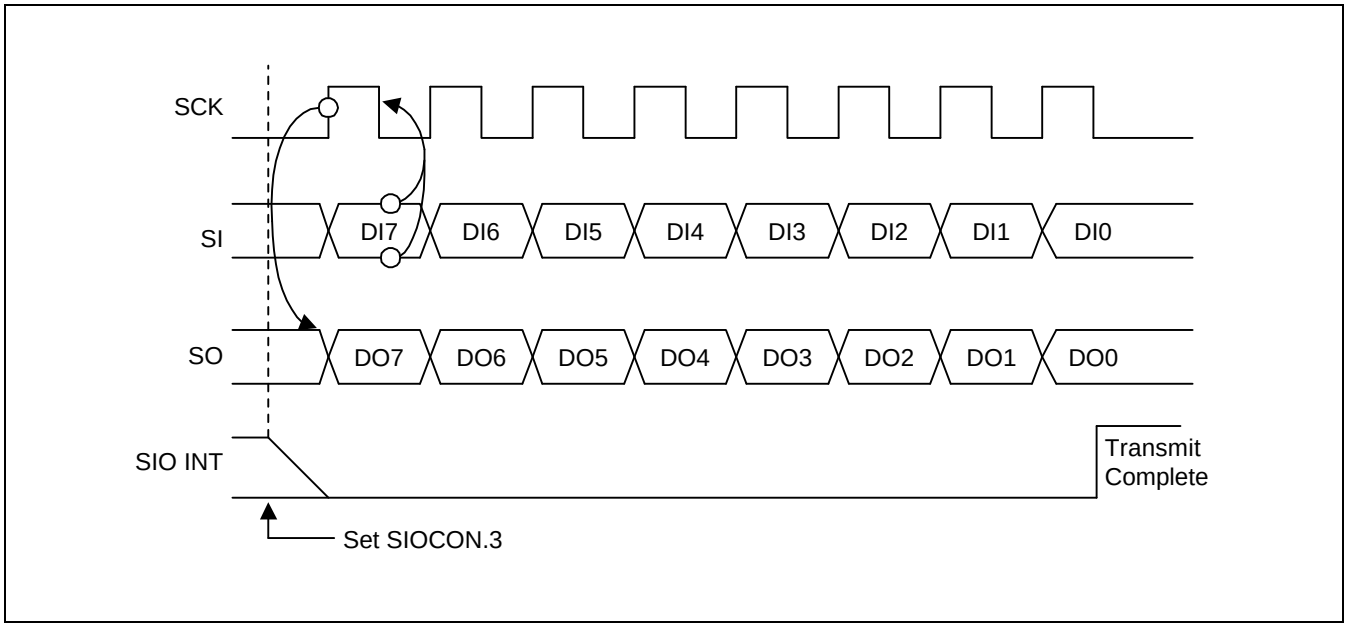


Figure 15-5. Serial I/O Timing in Transmit/Receive Mode (Tx at rising, SIOCON.4 = 1)

16 ELECTRICAL DATA

OVERVIEW

In this chapter, S3C9228/P9228 electrical characteristics are presented in tables and graphs. The information is arranged in the following order:

— Absolute maximum ratings
— D.C. electrical characteristics
— Data retention supply voltage in Stop mode
— Stop mode release timing when initiated by an external interrupt
— Stop mode release timing when initiated by a Reset
— I/O capacitance
— A.C. electrical characteristics
— A/D converter electrical characteristics
— Input timing for external interrupt
— Input timing for RESET
— Serial data transfer timing
— Oscillation characteristics
— Oscillation stabilization time
— Operating voltage range

Table 16-1. Absolute Maximum Ratings

(T_A = 25°C)

Parameter	Symbol	Conditions	Rating	Unit
Supply voltage	V_{DD}	–	– 0.3 to + 6.5	V
Input voltage	V_{IN}	Ports 0–6	– 0.3 to V_{DD} + 0.3	V
Output voltage	V_O	–	– 0.3 to V_{DD} + 0.3	V
Output current High	I_{OH}	One I/O pin active	– 15	mA
		All I/O pins active	– 60	
Output current Low	I_{OL}	One I/O pin active	+ 30	mA
		Total pin current for ports	+ 100	
Operating temperature	T_A	–	– 25 to + 85	°C
Storage temperature	T_{STG}	–	– 65 to + 150	°C

Table 16-2. D.C. Electrical Characteristics

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Operating Voltage	V_{DD}	fx = 0.4–4MHz, fxt = 32.8kHz	2.0	–	5.5	V
		fx = 0.4–8MHz	2.7	–	5.5	
Input High voltage	V_{IH1}	Ports 4–6	0.7 V_{DD}	–	V_{DD}	V
	V_{IH2}	Ports 0–3, RESET	0.8 V_{DD}		V_{DD}	
	V_{IH3}	X_{IN}, X_{OUT} and XT_{IN}, XT_{OUT}	V_{DD} – 0.1		V_{DD}	
Input Low voltage	V_{IL1}	Ports 4–6	–	–	0.3 V_{DD}	V
	V_{IL2}	Ports 0–3, RESET			0.2 V_{DD}	
	V_{IL3}	X_{IN}, X_{OUT}, XT_{IN}, XT_{OUT}			0.1	
Output High voltage	V_{OH}	V_{DD} = 4.5 to 5.5 V; All output ports; I_{OH} = –1 mA	V_{DD} – 1.0	–	V_{DD}	V
Output Low voltage	V_{OL}	V_{DD} = 4.5 to 5.5 V; All output ports; I_{OL} = 10 mA	–	–	2.0	V
Input High leakage current	I_{LIH1}	V_I = V_{DD}; All input pins except X_{IN}, X_{OUT}, XT_{IN}, XT_{OUT}	–	–	3	µA
	I_{LIH2}	V_I = V_{DD}; X_{IN}, X_{OUT}, XT_{IN}, XT_{OUT}			20	

Table 16-2. D.C. Electrical Characteristics (Continued)

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input Low leakage current	I_{LIL1}	V_I = 0 V; All input pins except RESET, X_{OUT}, XT_{IN}, XT_{OUT}	–	–	–3	µA
	I_{LIL2}	V_I = 0 V; X_{IN}, X_{OUT}, XT_{IN}, XT_{OUT}			–20	
Output High leakage current	I_{LOH}	$V_O = V_{DD}$ All output pins	–	–	3	
Output Low leakage current	I_{LOL}	V_O = 0 V All output pins	–	–	–3	
Pull-Up Resistor	R_{L1}	V_I = 0 V; V_{DD} = 5V, T_A = 25°C Ports 0–6	25	47	100	kΩ
		V_{DD} = 3V, T_A = 25°C	50	90	150	
	R_{L2}	V_I = 0 V; V_{DD} = 5V, T_A = 25°C RESET	150	250	400	
		V_{DD} = 3V, T_A = 25°C	300	500	700	
Oscillator Feed back Resistors	R_{OSC1}	V_{DD} = 5 V, T_A = 25 °C $X_{IN} = V_{DD}$, X_{OUT} = 0V	300	600	1500	kΩ
	R_{OSC2}	V_{DD} = 5 V, T_A = 25 °C $XT_{IN} = V_{DD}$, XT_{OUT} = 0 V	1500	3000	4500	
LCD Voltage Dividing Resistor	R_{LCD}	T_A = 25 °C	50	70	90	kΩ
$\lvert V_{LCD}\text{-COMi} \rvert$ Voltage Drop (i = 0-7)	V_{DC}	V_{DD} = 2.7 V to 5.5 V - 15 µA per common pin	–	–	120	mV
$\lvert V_{LCD}\text{-SEGx} \rvert$ Voltage Drop (x = 0–19)	V_{DS}	V_{DD} = 2.7 V to 5.5 V - 15 µA per common pin	–	–	120	
Middle Output Voltage	V_{LC2}	V_{DD} = 2.7 V to 5.5 V, LCD clock = 0Hz, $V_{LC1} = V_{DD}$	$0.8V_{DD}$–0.2	$0.8V_{DD}$	$0.8V_{DD}$+ 0.2	V
	V_{LC3}		$0.6V_{DD}$–0.2	$0.6V_{DD}$	$0.6V_{DD}$+ 0.2	
	V_{LC4}		$0.4V_{DD}$–0.2	$0.4V_{DD}$	$0.4V_{DD}$+ 0.2	
	V_{LC5}		$0.2V_{DD}$–0.2	$0.2V_{DD}$	$0.2V_{DD}$+ 0.2	

NOTE: Low leakage current is absolute value.

Table 16-2. D.C. Electrical Characteristics (Concluded)

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Supply current (1)	I_{DD1}	Run mode: V_{DD} = 5 V ± 10%	8.0 MHz	–	6.0	12.0	mA
		Crystal oscillator C1 = C2 = 22pF	4.19 MHz		3.0	6.0	
		V_{DD} = 3 V ± 10%	8.0 MHz		2.5	5.0	
			4.19 MHz		1.5	3.0	
	I_{DD2}	Idle mode: V_{DD} = 5 V ± 10%	8.0 MHz		1.3	3.0	
		Crystal oscillator C1 = C2 = 22pF	4.19 MHz		1.0	2.0	
		V_{DD} = 3 V ± 10%	8.0 MHz		0.8	1.6	
			4.19 MHz		0.4	0.8	
	I_{DD3}	Run mode: V_{DD} = 3 V ± 10%, 32 kHz crystal oscillator			15	30	µA
	I_{DD4}	Idle mode: V_{DD} = 3 V ± 10%, 32 kHz crystal oscillator			6	15	
	I_{DD5}	Stop mode; V_{DD} = 5 V ± 10%, T_A = 25 °C			0.5	3	
		Stop mode; V_{DD} = 3 V ± 10%, T_A = 25 °C			0.3	2	

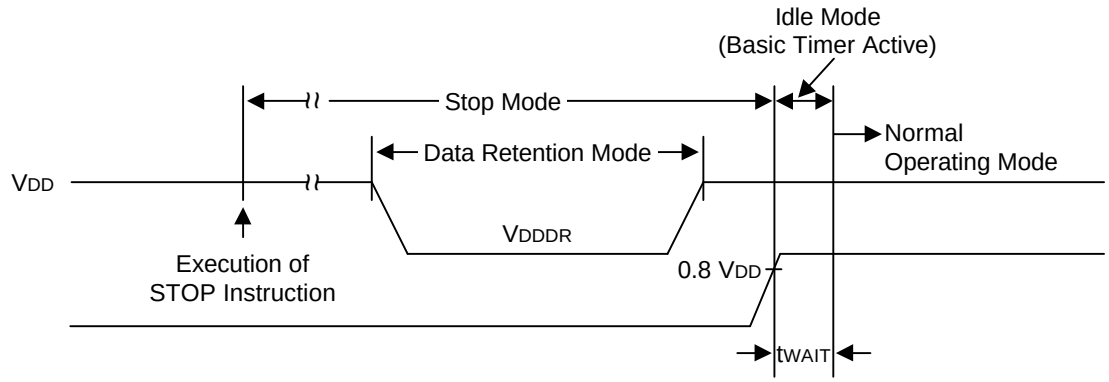
NOTES:

1. Supply current does not include current drawn through internal pull-up resistors, LCD voltage dividing resistors, and ADC.
2. I_{DD1} and I_{DD2} include power consumption for subsystem clock oscillation.
3. I_{DD3} and I_{DD4} are current when main system clock oscillation stops and the subsystem clock is used.
4. I_{DD5} is current when main system clock and subsystem clock oscillation stops.

Table 16-3. Data Retention Supply Voltage in Stop Mode

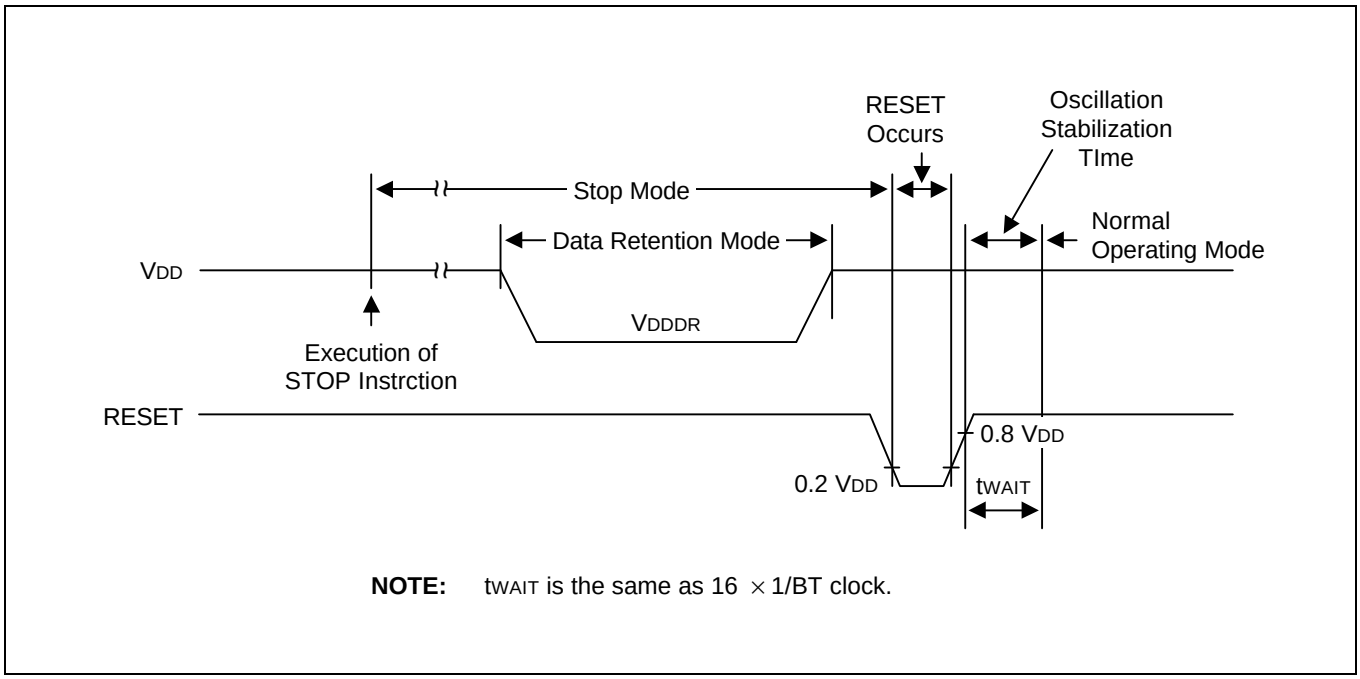
(T_A = – 25 °C to + 85 °C)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V_{DDDR}	–	2.0	–	5.5	V
Data retention supply current	I_{DDDR}	Stop mode, T_A = 25 °C V_{DDDR} = 2.0 V	–	–	1	µA



NOTE: tWAIT is the same as 16 x 1/BT clock.

Figure 16-1. Stop Mode Release Timing When Initiated by an External Interrupt



NOTE: tWAIT is the same as 16 × 1/BT clock.

Figure 16-2. Stop Mode Release Timing When Initiated by a RESET

Table 16-4. Input/Output Capacitance

(T_A = 25 °C, V_{DD} = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input capacitance	C_{IN}	f = 1 MHz; unmeasured pins are connected to V_{SS}	–	–	10	pF
Output capacitance	C_{OUT}					
I/O capacitance	C_{IO}					

Table 16-5. A.C. Electrical Characteristics

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
SCK cycle time	t_{KCY}	External SCK source	1,000	–	–	ns
		Internal SCK source	1,000			
SCK high, low width	t_{KH}, t_{KL}	External SCK source	500			
		Internal SCK source	t_{KCY}/2–50			
SI setup time to SCK high	t_{SIK}	External SCK source	250			
		Internal SCK source	250			
SI hold time to SCK high	t_{KSI}	External SCK source	400			
		Internal SCK source	400			
Output delay for SCK to SO	t_{KSO}	External SCK source	–	–	300	ns
		Internal SCK source			250	
Interrupt input, High, Low width	t_{INTH}, t_{INTL}	All interrupt V_{DD} = 3 V	500	700	–	ns
RESET input Low width	t_{RSL}	Input V_{DD} = 3 V	10	–	–	µs

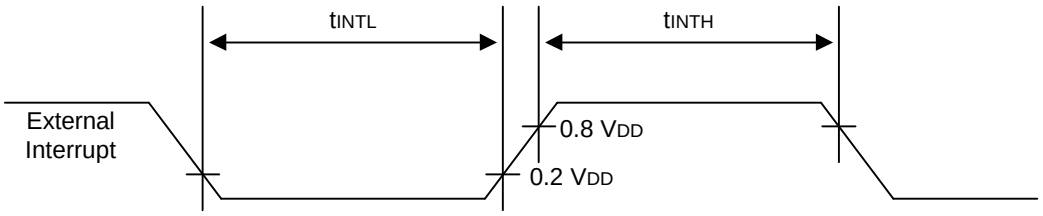
Table 16-6. A/D Converter Electrical Characteristics

(T_A = – 25°C to + 85°C, V_{DD} = 2.7 V to 5.5 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Resolution			–	10	–	bit
Total accuracy		VDD = 5.12 V	–	–	±3	LSB
Integral linearity error	ILE	fxx = 8 MHz	–	–	±2	
Differential linearity error	DLE	f_{CON} = fxx/4	–	–	±1	
Offset error of top	EOT		–	±1	±3	
Offset error of bottom	EOB		–	±0.5	±2	
Conversion time (1)	T_{CON}	10-bit resolution 50 × fxx/4, fxx = 8MHz	25	–	–	µS
Analog input voltage	V_{IAN}	–	V_{SS}	–	V_{DD}	V
Analog input impedance	R_{AN}	–	2	1000	–	MΩ
Analog input current	I_{ADIN}	V_{DD} = 5 V	–	–	10	µA
	I_{ADC}	V_{DD} = 5 V	–	1	3	mA
		V_{DD} = 3 V		0.5	1.5	
		V_{DD} = 5 V When 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 an operating current during A/D conversion.



NOTE: The unit tCPU means one CPU clock period.

Figure 16-3. Input Timing for External Interrupts

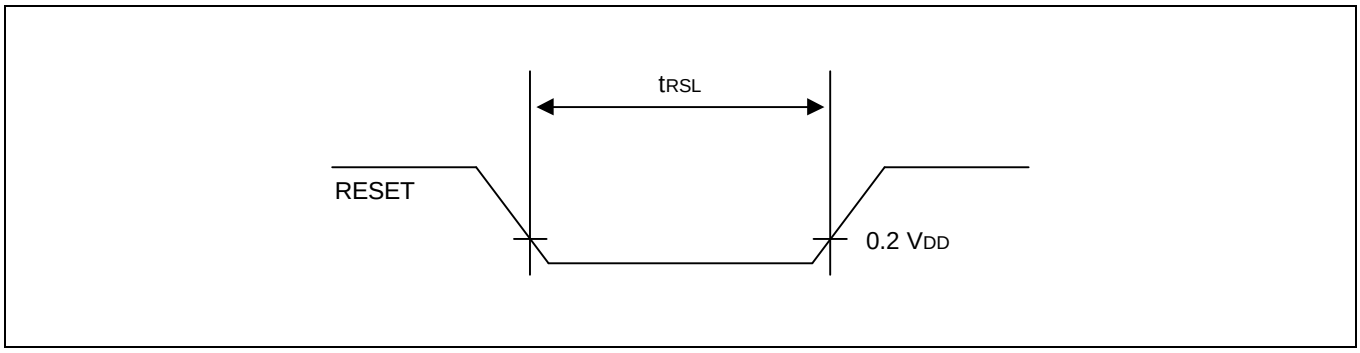


Figure 16-4. Input Timing for RESET

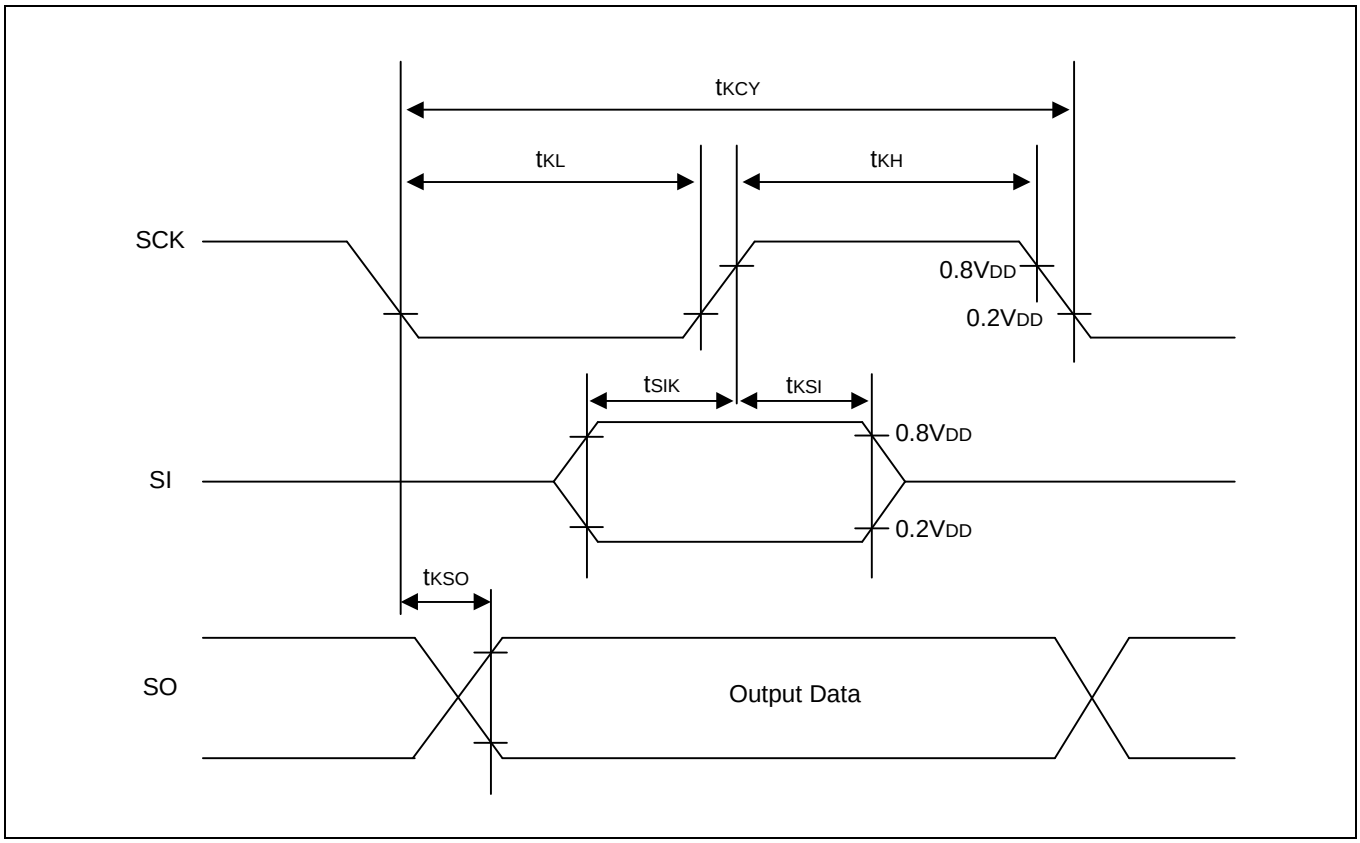


Figure 16-5. Serial Data Transfer Timing

Table 16-7. Main Oscillation Characteristics

(T_A = – 25°C to + 85°C)

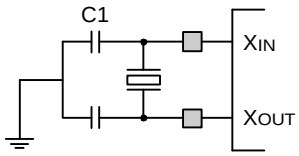
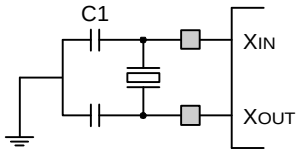
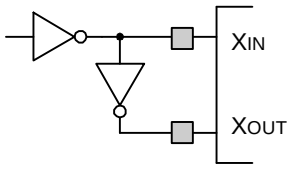
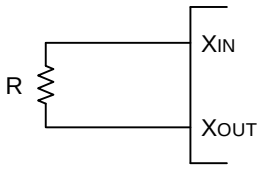
Oscillator	Clock Configuration	Parameter	Test Condition	Min	Typ	Max	Units
Crystal		Main oscillation frequency	2.7 V – 5.5 V	0.4	–	8	MHz
			2.0 V – 5.5 V	0.4	–	4	
Ceramic Oscillator		Main oscillation frequency	2.7 V – 5.5 V	0.4	–	8	
			2.0 V – 5.5 V	0.4	–	4	
External Clock		X_{IN} input frequency	2.7 V – 5.5 V	0.4	–	8	
			2.0 V – 5.5 V	0.4	–	4	
RC Oscillator		Frequency	5.0 V	0.4	–	2	MHz
		Frequency	3.0 V	0.4	–	1	

Table 16-8. Sub Oscillation Characteristics

(T_A = – 25°C to + 85°C)

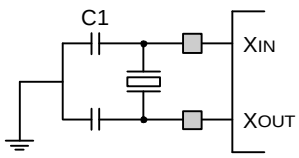
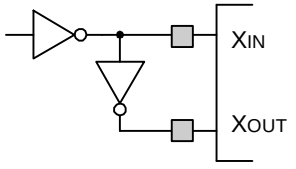
Oscillator	Clock Configuration	Parameter	Test Condition	Min	Typ	Max	Units
Crystal		Sub oscillation frequency	2.0 V – 5.5 V	32	32.768	35	kHz
External clock		XT_{IN} input frequency	2.0 V – 5.5 V	32	–	100	

Table 16-9. Main Oscillation Stabilization Time

(T_A = – 25 °C to + 85 °C, V_{DD} = 2.0 V to 5.5 V)

Oscillator	Test Condition	Min	Typ	Max	Unit
Crystal	fx > 1 MHz	–	–	30	ms
Ceramic	Oscillation stabilization occurs when VDD is equal to the minimum oscillator voltage ranage.	–	–	10	ms
External clock	X_{IN} input high and low width (t_{XH}, t_{XL})	62.5	–	1250	ns

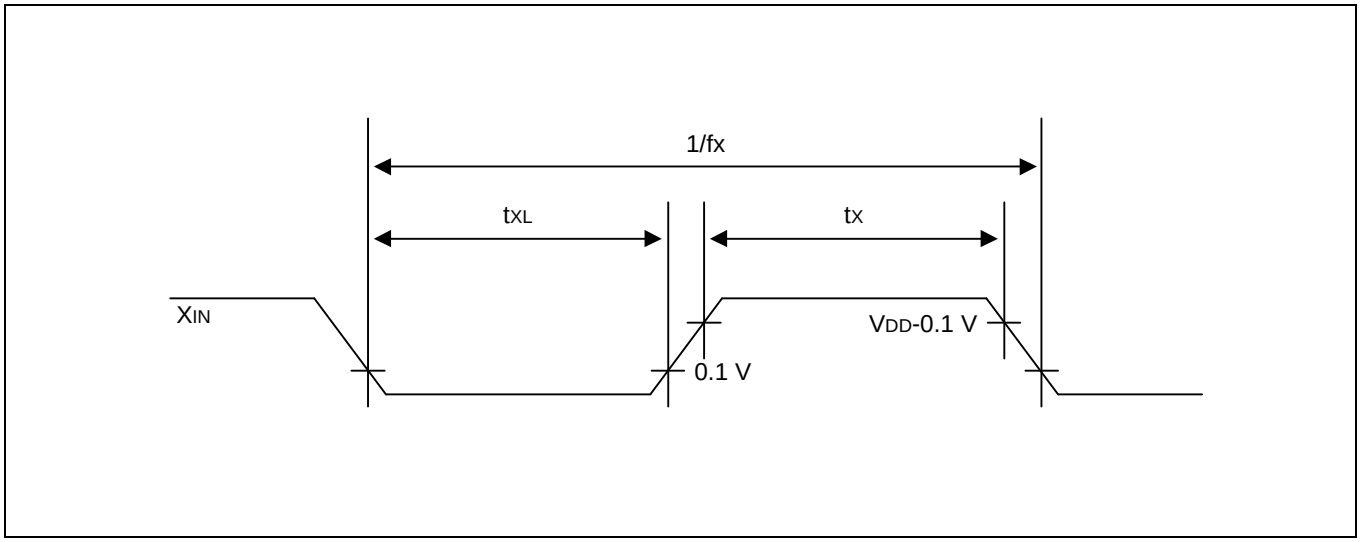


Figure 16-6. Clock Timing Measurement at X_{IN}

Table 16-10. Sub Oscillation Stabilization Time

(T_A = – 25 °C to + 85 °C, V_{DD} = 2.0 V to 5.5 V)

Oscillator	Test Condition	Min	Typ	Max	Unit
Crystal	–	–	–	10	s
External clock	XT_{IN} input high and low width (t_{XH}, t_{XL})	5	–	15	µs

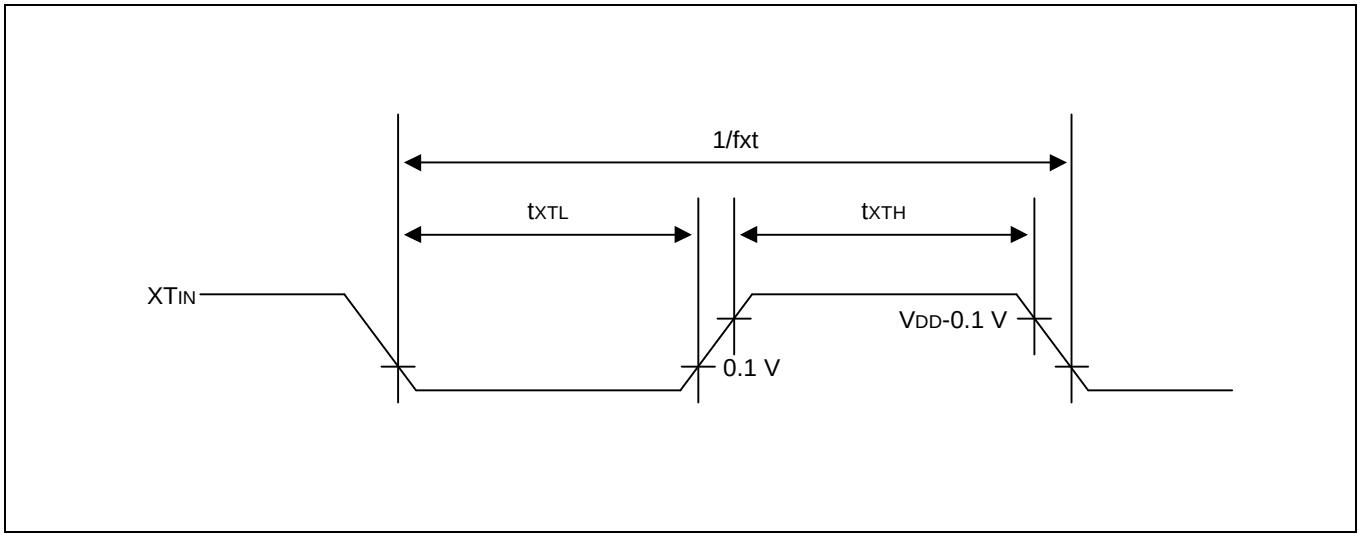
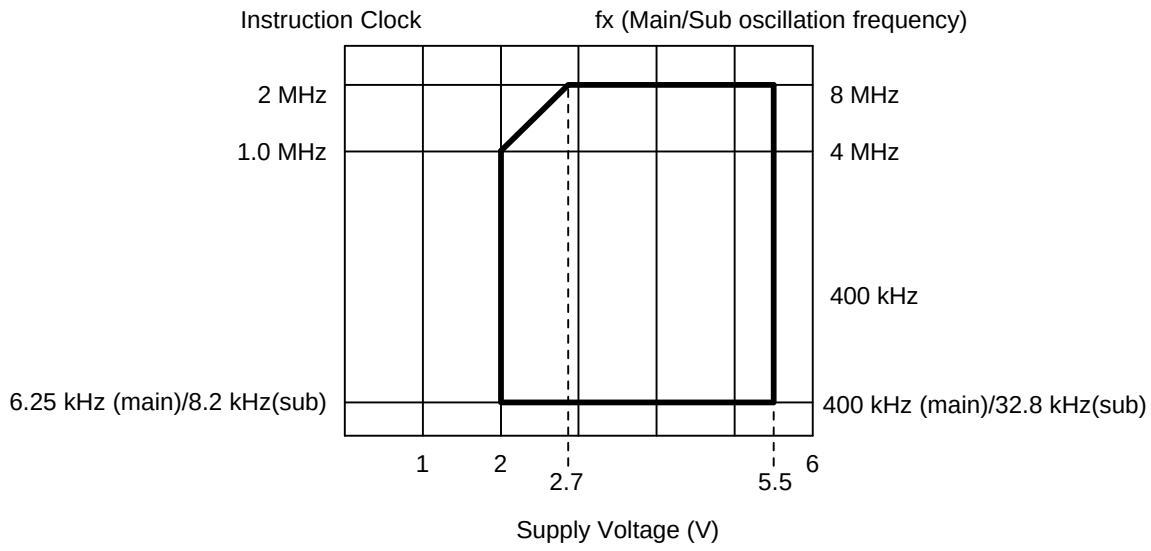


Figure 16-7. Clock Timing Measurement at XT_{IN}



Instruction Clock = 1/4n x oscillator frequency (n = 1, 2, 8, 16)

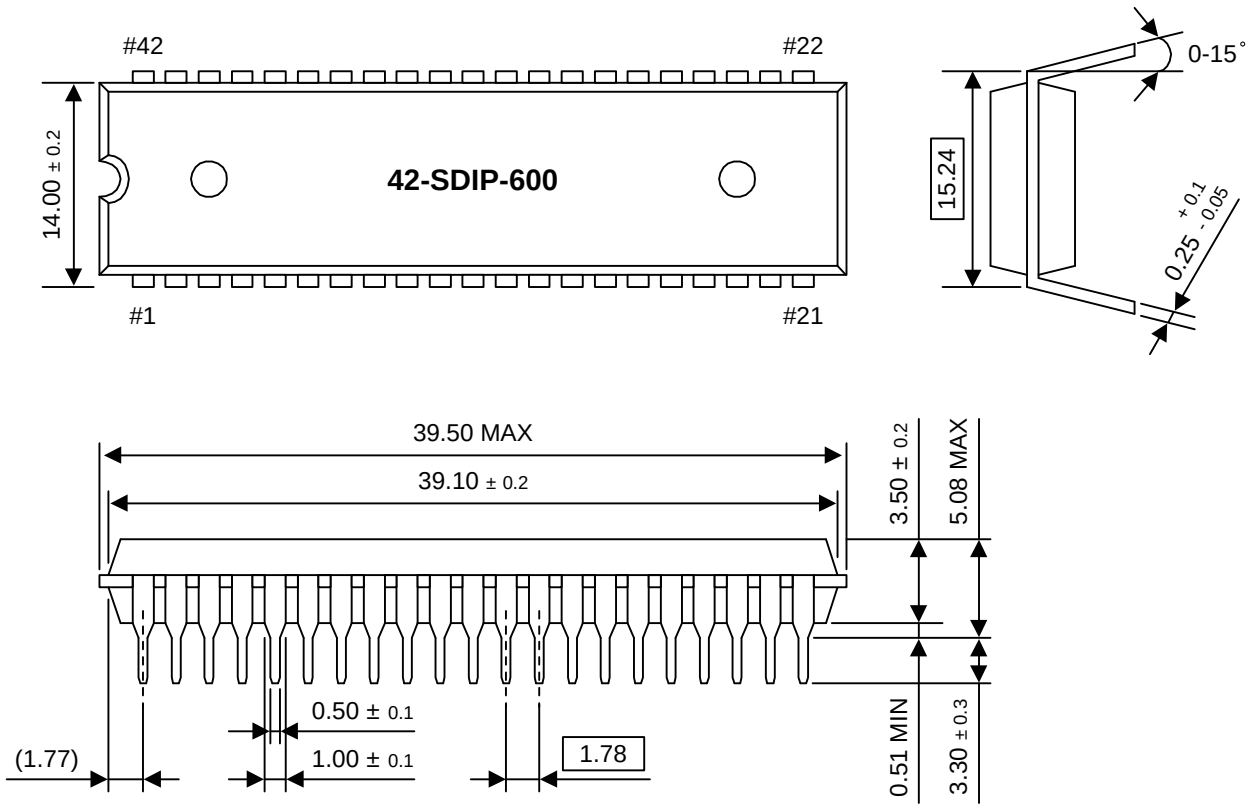
Figure 16-8. Operating Voltage Range

NOTES

17 MECHANICAL DATA

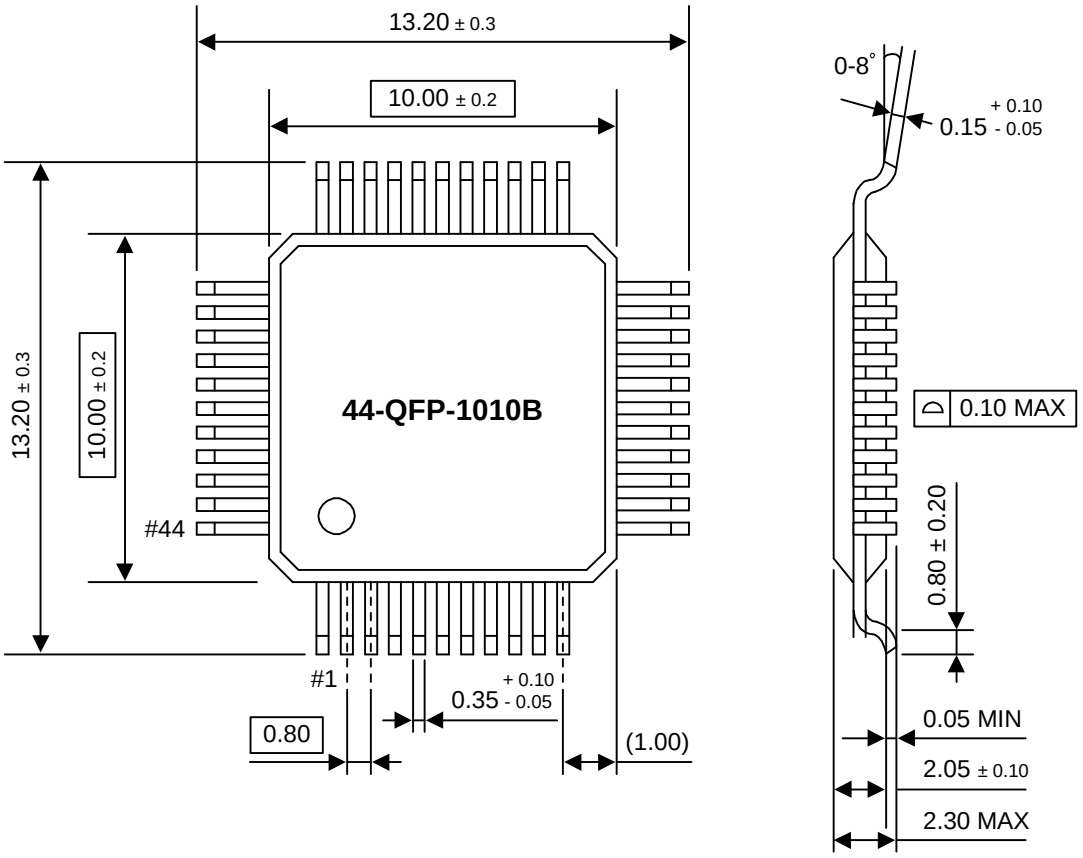
OVERVIEW

The S3C9228/P9228 microcontroller is currently available in a 42-pin SDIP and 44-pin QFP package.



NOTE: Dimensions are in millimeters.

Figure 17-1. 42-SDIP-600 Package Dimensions



NOTE: Dimensions are in millimeters.

Figure 17-2. 44-QFP-1010B Package Dimensions

18 S3P9228 OTP

OVERVIEW

The S3P9228 single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the S3C9228 microcontroller. It has an on-chip OTP ROM instead of masked ROM. The EPROM is accessed by serial data format.

The S3P9228 is fully compatible with the S3C9228, both in function and in pin configuration. Because of its simple programming requirements, the S3P9228 is ideal for use as an evaluation chip for the S3C9228.

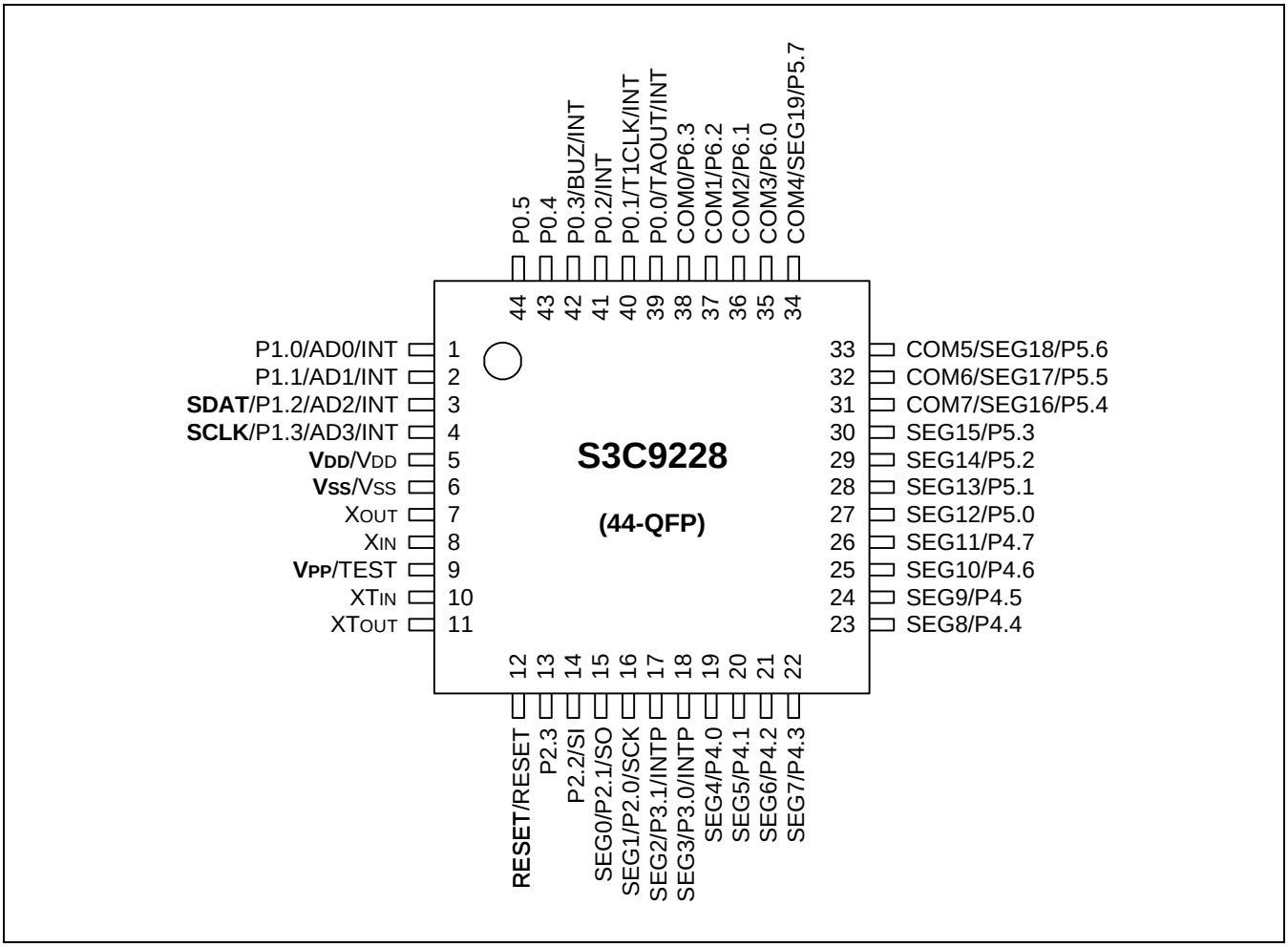


Figure 18-1. S3P9228 44-QFP Pin Assignments

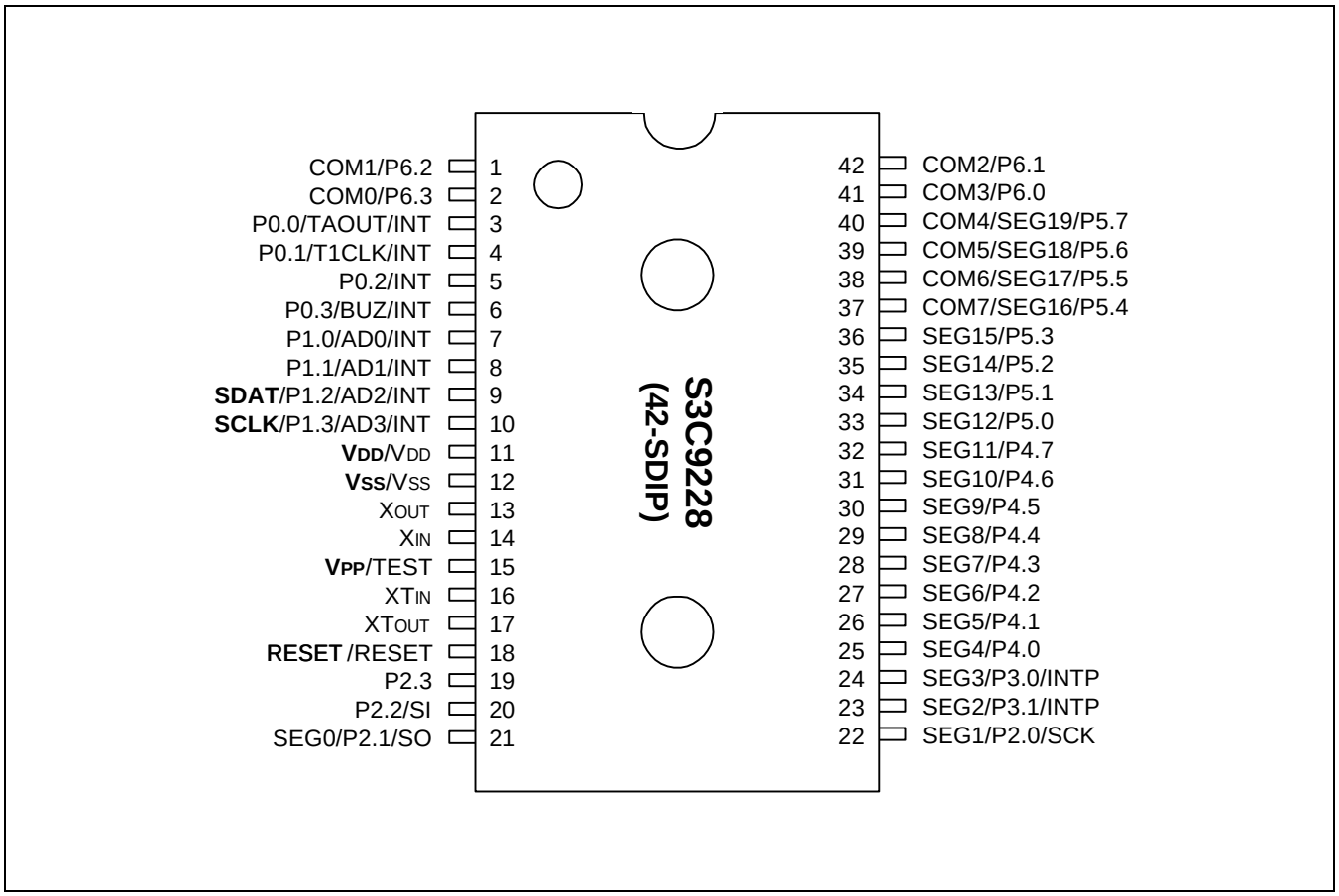


Figure 18-2. S3P9228 42-SDIP Pin Assignments

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

Main Chip	During Programming			
Pin Name	Pin Name	Pin No.	I/O	Function
P1.2	SDAT	3 (9)	I/O	Serial data pin. Output port when reading and input port when writing. Can be assigned as a Input/push-pull output port.
P1.3	SCLK	4 (10)	I/O	Serial clock pin. Input only pin.
TEST	V_{PP}(TEST)	9 (15)	I	Power supply pin for EPROM cell writing (indicates that OTP enters into the writing mode). When 12.5 V is applied, OTP is in writing mode and when 5 V is applied, OTP is in reading mode. (Option)
RESET	RESET	12 (18)	I	Chip initialization
V_{DD}/V_{SS}	V_{DD}/V_{SS}	5/6 (11/12)	I	Logic power supply pin. V_{DD} should be tied to + 5 V during programming.

NOTE: Parentheses indicate pin number for 42-SDIP package.

Table 18-2. Comparison of S3P9228 and S3C9228 Features

Characteristic	S3P9228	S3C9228
Program Memory	8 Kbyte EPROM	8 Kbyte mask ROM
Operating Voltage (V_{DD})	2.0 V to 5.5 V	2.0 V to 5.5 V
OTP Programming Mode	V_{DD} = 5 V, V_{PP}(TEST)=12.5V	
Pin Configuration	44-QFP, 42-SDIP	44-QFP, 42-SDIP
EPROM Programmability	User Program 1 time	Programmed at the factory

OPERATING MODE CHARACTERISTICS

When 12.5 V is supplied to the V_{PP}(TEST) pin of the S3P72C8, the EPROM programming mode is entered. The operating mode (read, write, or read protection) is selected according to the input signals to the pins listed in Table 17-3 below.

Table 18-3. Operating Mode Selection Criteria

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

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

Table 18-4. D.C. Electrical Characteristics

(T_A = – 25°C to + 85°C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Supply current (1)	I_{DD1}	Run mode: V_{DD} = 5 V ± 10%	8.0 MHz	–	6.0	12.0	mA
		Crystal oscillator C1 = C2 = 22pF	4.19 MHz		3.0	6.0	
		V_{DD} = 3 V ± 10%	8.0 MHz		2.5	5.0	
			4.19 MHz		1.5	3.0	
	I_{DD2}	Idle mode: V_{DD} = 5 V ± 10%	8.0 MHz		1.3	3.0	
		Crystal oscillator C1 = C2 = 22pF	4.19 MHz		1.0	2.0	
		V_{DD} = 3 V ± 10%	8.0 MHz		0.8	1.6	
			4.19 MHz		0.4	0.8	
	I_{DD3}	Run mode: V_{DD} = 3 V ± 10%, 32 kHz crystal oscillator			15	30	µA
	I_{DD4}	Idle mode: V_{DD} = 3 V ± 10%, 32 kHz crystal oscillator			6	15	
	I_{DD5}	Stop mode; V_{DD} = 5 V ± 10%, T_A = 25 °C			0.5	3	
		Stop mode; V_{DD} = 3 V ± 10%, T_A = 25 °C			0.3	2	

NOTES:

1. Supply current does not include current drawn through internal pull-up resistors, LCD voltage dividing resistors, and ADC.
2. I_{DD1} and I_{DD2} include power consumption for subsystem clock oscillation.
3. I_{DD3} and I_{DD4} are current when main system clock oscillation stops and the subsystem clock is used.
4. I_{DD5} is current when main system clock and subsystem clock oscillation stops.

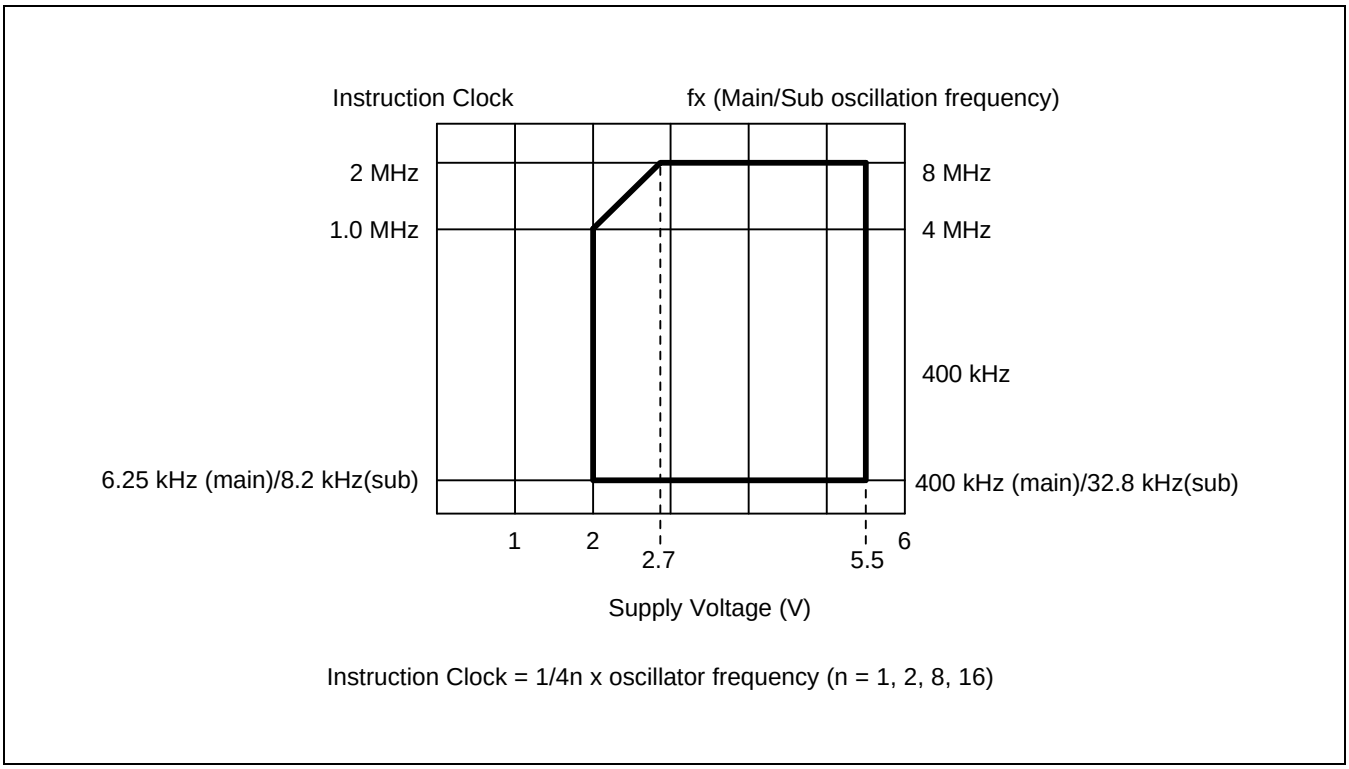


Figure 18-3. Standard Operating Voltage Range

NOTES

19 DEVELOPMENT TOOLS

OVERVIEW

Samsung provides a powerful and easy-to-use development support system in turn key 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, S3C8, S3C9 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.

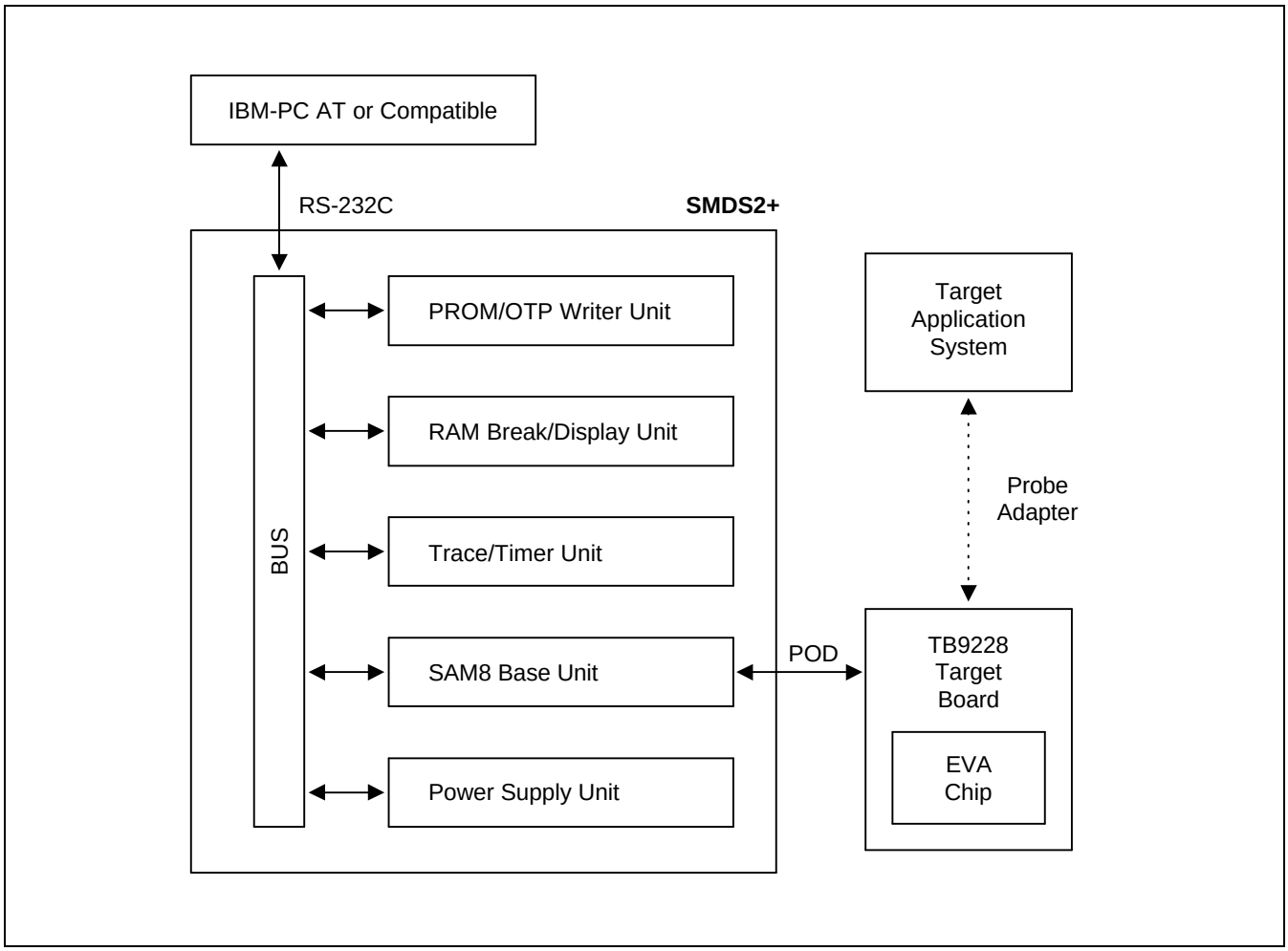


Figure 19-1. SMDS Product Configuration (SMDS2+)

TB9228 TARGET BOARD

The TB9228 target board is used for the S3C9228 microcontroller. It is supported by the SMDS2+ development system.

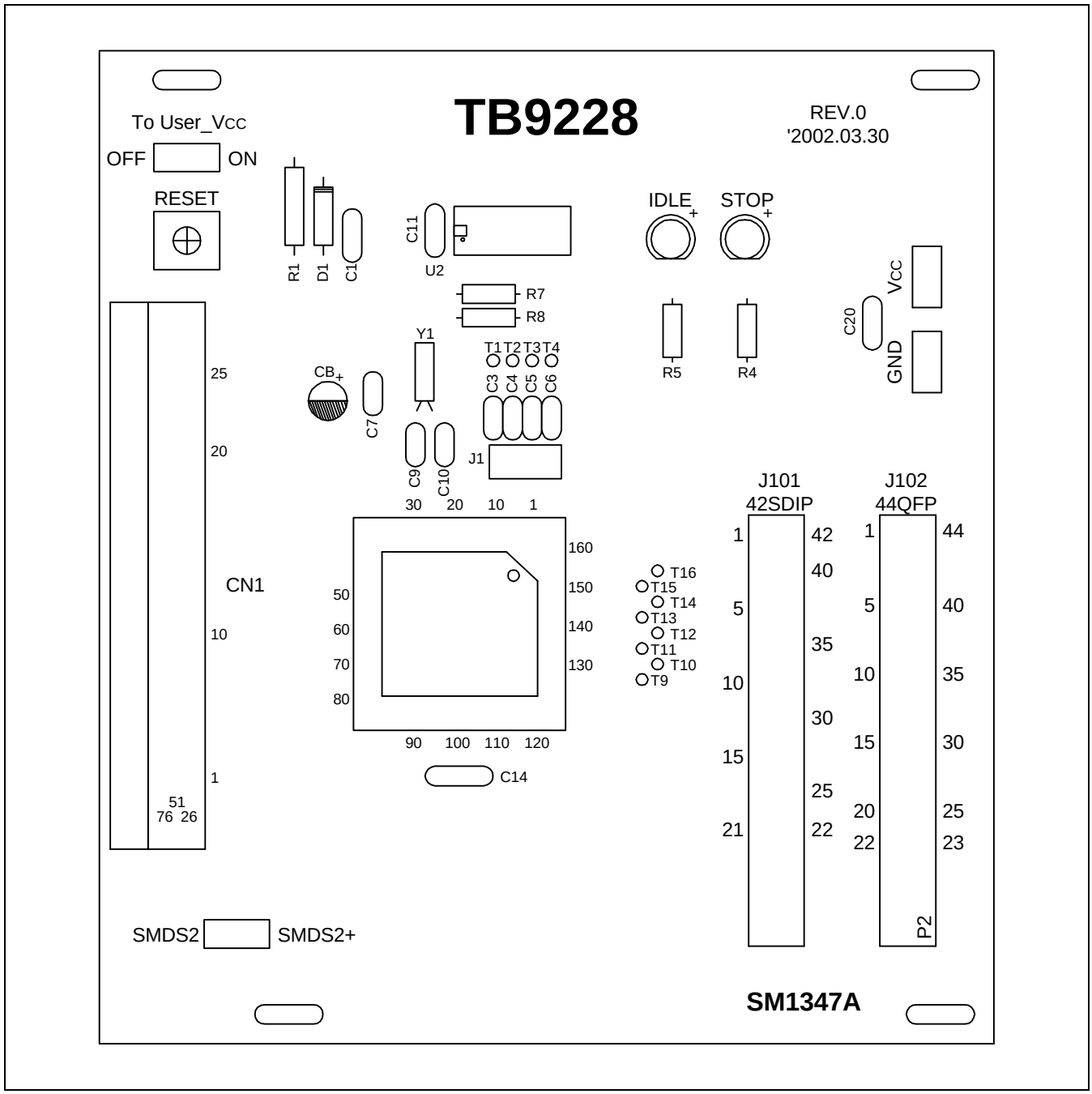
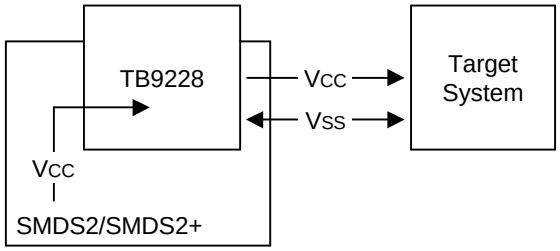
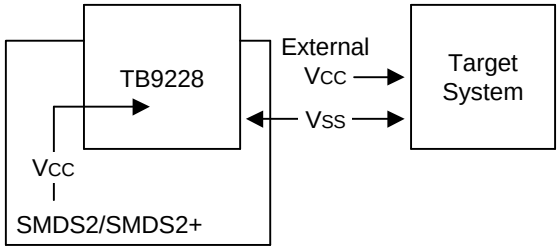


Figure 19-2. TB9228 Target Board Configuration

Table 19-1. Power Selection Settings for TB9228

"To User_V_{CC}" Settings	Operating Mode	Comments
To User_Vcc Off On		The SMDS2/SMDS2+ supplies V_{CC} to the target board (evaluation chip) and the target system.
To User_Vcc Off On		The SMDS2/SMDS2+ supplies V_{CC} only to the target board (evaluation chip). The target system must have its own power supply.

NOTE: The following symbol in the "To User_V_{CC}" Setting column indicates the electrical short (off) configuration:

SMDS2+ Selection (SAM8)

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

Table 19-2. The SMDS2+ Tool Selection Setting

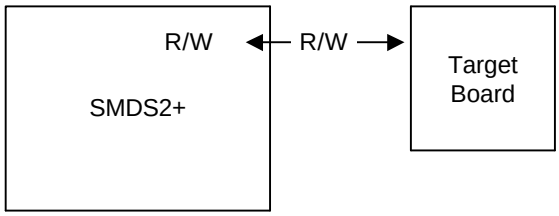
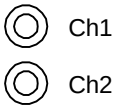
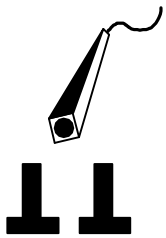
"SW1" Setting	Operating Mode
SMDS2 SMDS2+	

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

Target Board Part	Comments
External Triggers 	 Connector from External Trigger Sources of the Application System 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.

IDLE LED

The Green LED is ON when the evaluation chip (S3E9220) is in idle mode.

STOP LED

The Red LED is ON when the evaluation chip (S3E9220) is in stop mode.

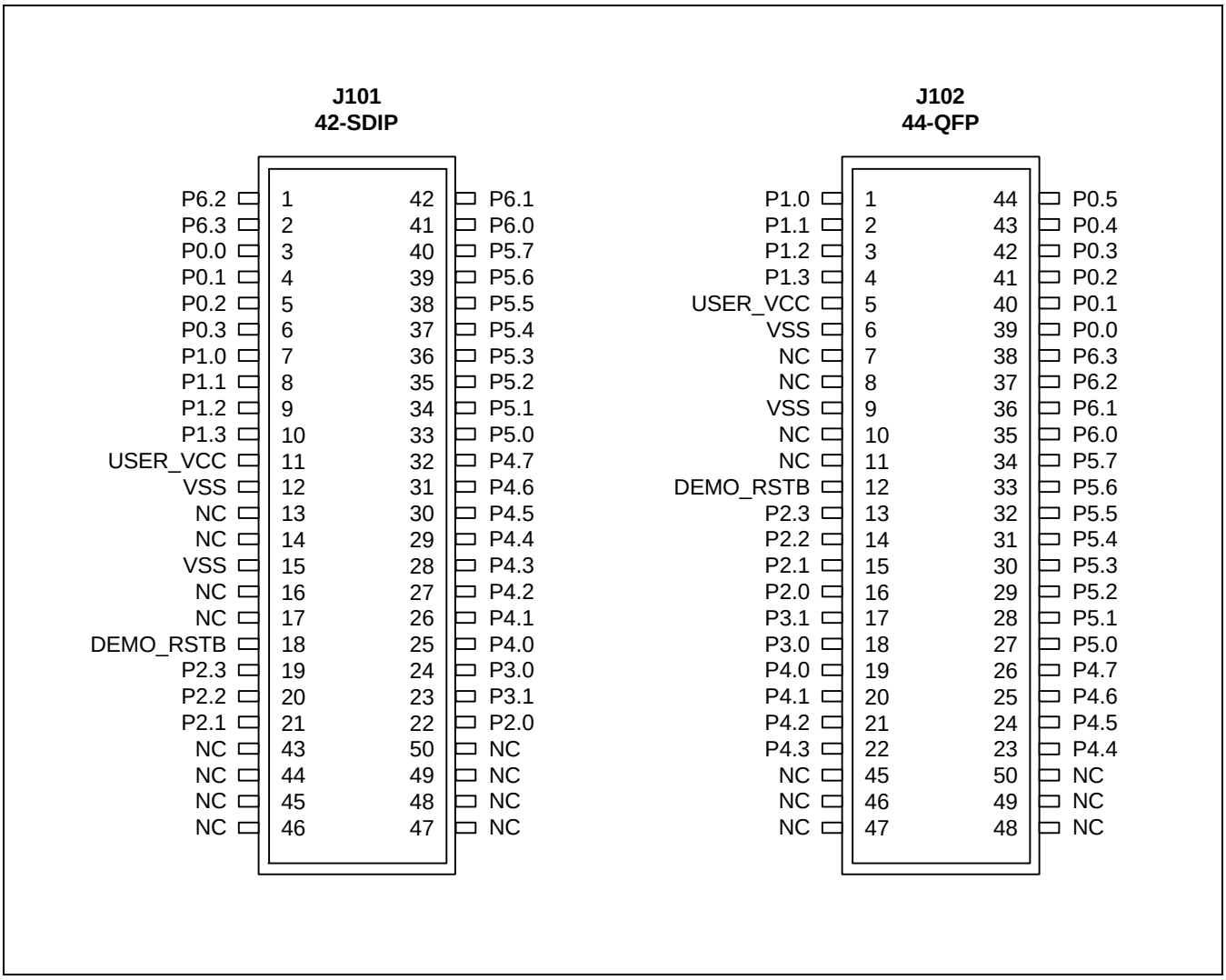


Figure 19-3. Connectors (J101, J102) for TB9228

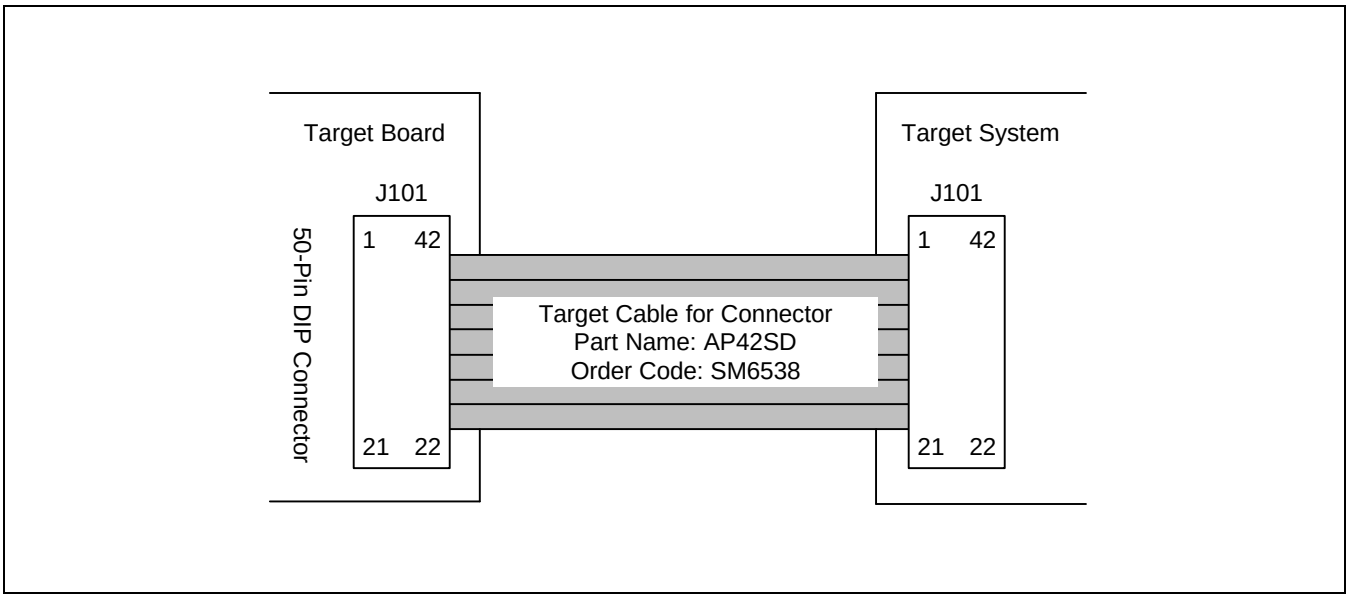


Figure 19-4. S3C9228 Probe Adapter for 42-SDIP Package

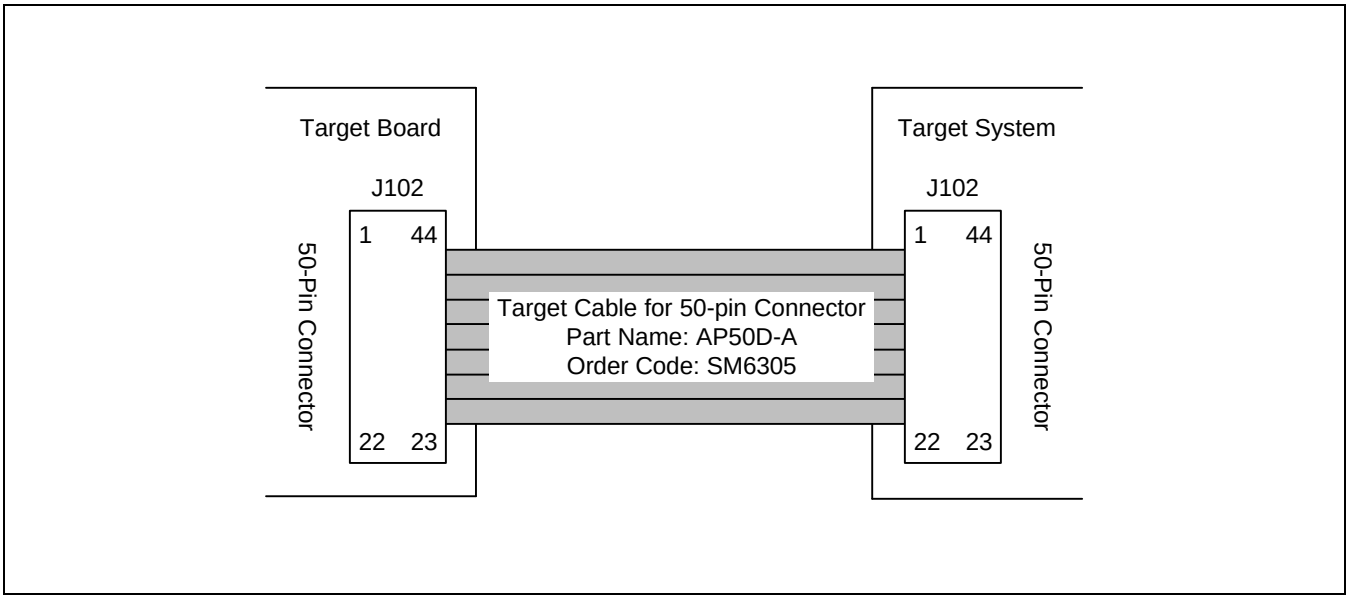


Figure 19-5. S3C9228 Probe Adapter for 44-QFP Package

NOTES